ADDIS ABABA INSTITUTE OF TECHNOLOGY



School of Information Technology and Engineering (SiTE)

Modular Curriculum

for

Degree of Bachelor of Science in Software Engineering and Computing Technology (Software Engineering, Cyber Security, Information Technology, Artificial Intelligence Streams)

2021 G.C

Acronyms 8
Introduction 1
Rationale of the Program 3
Structure of the Program 4
Objectives of the Program 5
General Objective 5
Specific Objectives 6
Program Outcomes 6
Graduate Profile 6
Cognitive knowledge 6
Practical skills 7
Attitude (Transferable skill) 7
Professional Profile 8
EDUCATIONAL PROCESS 9
Entry and Admission Requirements 9
Exam and Attendance Requirement 9
Graduation Requirement 9
Duration of the Study 9
Degree Nomenclature 10
Course Coding 11
Modularization 11
Modules 12
Common and Supportive Modules - Year I 12
Core Modules - Year I, II, and III 13

Core Modules - Year IV and V 15
Core Modules - Year IV and V 16
Core Modules - Year IV and V 17
Core Modules - Year IV and V 18
Common Modules for Year IV and V 19
Electives 20
Teaching-Learning Methods and Materials 22
Assessment Methods 23
Grading Scheme 23
Quality Assurance 24
Resource Profile of the School 26
Permanent Staff - Professors 26
Permanent Staff - Lecturers 27
Permanent Staff - Assistant Lecturers 28
Visiting Professors 28
Technical Assistants 29
Appendix 1A: Course Schedule 30
Software Engineering and Computing Technology Program 30
Software Engineering Stream 32
Summary of SE Stream ECTS 35
Information Technology Stream 36
Summary of Information Technology Stream ECTS 38
Artificial Intelligence Stream 39
Summary of Artificial Intelligence Stream ECTS 41
Cyber Security Stream 42

Summary of Cyber Security Stream ECTS 44
Appendix 1B-Software Engineering and Computing Technology Syllabuses 45
Year I - Year III 45
SEM I 45
Communication Skill (English) 45
Engineering Drawing 46
Engineering Professions * 49
Engineering Mechanics (stat) 50
Applied Mathematics I 51
Civics and Ethical Education 53
SEM II 55
Fundamental of Computer Systems 55
Fundamental of Computer Science 60
Basic Writing Skills 64
Linear Algebra 66
Applied Math II 70
SEM III 74
Fundamentals of Database 74
Probability and Statistics 81
Discrete Mathematics 86
Fundamentals of Data structures 90
Fundamental of Networking 93
SEM IV 97
System Programming 97
Fundamental of Electrical Circuits and Electronics 103

Fundamental of Web Design and Programming 106
Object Oriented Programming 109
Fundamental of Software Engineering 113
SEM V 117
Algorithm Analysis 117
Mobile Application Development 121
Operating Systems 124
Digital Logic Design 129
Computer Graphics 132
SEM VI 136
Fundamental of IT security 136
Fundamental of AI 141
HCI 145
Design Patterns 150
Computer Organizations and Architecture 153
Programming Paradigms 156
Appendix 1C- Syllabuses per Stream 160
Software Engineering Stream 160
SEM VII 160
Industrial Internship Guideline 160
SEM VIII 160
Software Requirement Engineering 160
Software Architecture and Design 168
Fundamentals of Parallel Computing 172
Advanced Computer Networking 176

Enterprise Application Development 180
SEM IX 184
Advanced Database Systems 184
Non Engineering Elective II 189
Software Project Management 190
Fundamentals of Distributed Systems 194
Project I 198
SEM X 199
Technology Entrepreneurship 199
Ethics and Professionalism 199
Software Testing, Verification and Quality Assurance 204
Selected Topics in Software Engineering 209
Project II 209
Information Technology Stream 210
SEM VII 210
Industry Internship Guideline 210
SEM VIII 210
Database Administration and Security 210
E-business Strategy and Development 219
Enterprise Systems and Network Administration 223
Internet of Things 229
Data Mining and Analytics 235
SEM IX 235
Fundamentals of Distributed Systems 235
Cloud Computing and Security 240

Non Engineering Elective II 245
Enterprise Application Development 245
Project I, 245
SEM X 246
Ethical Hacking and Penetration Testing 246
Non Engineering Elective II 246
Project II 246
Selected Topics in Information Technology 246
Technology Entrepreneurship 246
Artificial Intelligence Stream 247
SEM VII 247
Numerical Optimization and ODE 247
Social Network Analysis 252
Software Project Management 255
Machine Learning 255
Fundamentals of Parallel Computing 259
SEM VIII 262
SEM IX 262
Non Engineering Elective II Project - I 262
Deep Learning 262
Data Mining and Analytics 266
Machine Intelligence 272
Selected Topics in AI 275
SEM X 276
Computer Vision 276

Reinforcement Learning 280
Project II 284
Non Engineering Elective II 284
Technology Entrepreneurship 284
Cyber Security Engineering 285
SEM VII 285
Enterprise Systems and Network Security 285
Cryptographic Concepts and Applications * 289
Systems and Network Security Administration 293
Machine Learning 294
Software Project Management 294
SEM VIII 295
SEM IX 295
Non Engineering Elective II 295
Project I 295
Ethical Hacking and Penetration Testing 296
Computer Crime and Digital Forensics 299
Web Security 303
Electives 306
Natural Language Processing 306
Introduction to Cognitive Science 315
Appendix 1C: Summary of SECT Curricula 319
Software Engineering Stream 319
Information Technology Stream 319
Artificial Intelligence Stream 320

Acronyms

AAiT	Addis Ababa Institute of Technology
AAU	Addis Ababa University
AC	Academic Council
AI	Artificial Intelligence
BSc	Bachelor of Science
CGPA	Cumulative Grade Point Average
ECTS	European Credit Transfer System
SiTE	School Information Technology and Engineering
MSc	Master of Science
SGS	School of Graduate Studies

Introduction

Computing is the process of using computer technology to complete a given goal-oriented task. Computing may encompass the design and development of software and hardware systems for a broad range of purposes - often structuring, processing and managing any kind of information - to aid in the pursuit of scientific studies, making intelligent systems, and creating and using different media for entertainment and communication. Specifically, Software Engineering and Computing Technology (SECT) is the study of mathematical principles, methods, and technological knowledge to design, implement, test and maintain computing systems.

The Association for Computing Machinery (ACM) states Computer Science, Information Technology, Cyber Security, Software Engineering, and Information Science as sub disciplines of Computing. Similarly, this stream based curriculum (SECT) encloses the aforementioned soft computing sub disciplines namely: Software Engineering, Information Technology, Cyber Security, and Artificial Intelligence in turn which incorporates the aforementioned sub disciplines of computing.

However, it is common to observe many standards being attached to these disciplines. Some organizations diminish the body of knowledge and others misinterpreting them in a way one could not plot clear boundary among them. Therefore, this document gives a clear interpretation of all the above sub disciplines under software engineering and computing technology program.

Hence, in this document software engineering as a discipline is defined as *the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, documentation and maintenance of software system.* This definition is derived from various reputable institutions, scholars, and their articles written on specification of the software engineering as a discipline such as:

> "*Software engineering is that form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems.*" *[CMU/SEI-90-TR-003]*
>
> "*Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software.*" *[IEEE 1990]*
>
> "*The establishment and use of sound engineering principles (methods) in order to obtain economically software that is reliable and works on real machines.*" *[Bauer 1972]*

As it is stated in the above standardization, software engineering is about creating high-quality software in a systematic, controlled, and efficient manner. It also gives a strong emphasis to analysis and specification, design, implementation, and management of the evolution of a software product.

The other sub discipline that has been misinterpreted as software engineering since its existence is Information Technology. This document defines Information Technology as an academic discipline which is concerned with issues related to advocating for users and meeting their needs within an organizational and societal context through the selection, creation, application, integration and administration of computing technologies [ACM, 2018]. Hence, Information Technology refers to a program that prepares students to meet the computer technology needs of business, government, healthcare, schools, and other kinds of organizations. Rather than Information system programs which emphasize on the information, the major emphasis in IT is on the technology itself. Hence, IT and most of its specialization are market oriented and are supposed to fill the technology gaps observed in institutions. Similarly, AAiT will also use this program to equip students who can address specific demands of stakeholders.

Cyber Security is also one of the multidisciplinary streams of the SECT curriculum. Its definition is standardized as a *computing-based discipline involving technology, people, information, and processes to analyze create, and test secured computer systems. [ACM, 2017]* It is an interdisciplinary course of study, including aspects of law, policy, human factors, ethics, and risk management. Cyber Security draws from the foundational fields of information security and information assurance; and began with more narrowly focused field of computer security. Hence, this program is in general more to Information Technology as it focuses on the technologies to secure systems and Software Engineering as it intends to create secure computing system in a strategic manner.

Artificial Intelligence is also one of the latest computing disciplines that came out to solve exponentially growing problems of the world. Artificial Intelligence is "The science and engineering of making intelligent machines, especially intelligent computer

programs". It is accomplished by studying how human brain thinks and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software systems. Today, it is known as narrow AI or weak AI, in that it is designed to perform narrow tasks like facial recognition or internet searches and etc. However, the long-term goal of many researchers is to create Artificial General Intelligence (AGI).

Generally, the program entitled Software Engineering and Computing Technology produces students with the ability to develop and efficiently utilize high quality, secured, and autonomic computing systems. Launching such problem solving programs contributes a lot to developing countries who intend to grow its economy by using equipped ICT infrastructures. Ethiopia is a good example for this. Hence, Software Engineering and Computing Technology are considered as an enabler of socio-economic transformation. It is believed to facilitate and transform Ethiopia's predominantly subsistence-agriculture economy and society into an information-and knowledge-based economy and society, effectively integrated into the global economy.

Rationale of the Program

The dual challenges of society's critical dependence on Computing Technologies, and the relative immaturity of related engineering practices, make attention to professional practice issues even more important to software engineering and computing technology programs than many other engineering programs. Graduates of SECT programs need to arrive in the workplace equipped to meet these challenges and to help evolve the discipline into a more professional and accepted state. Like other engineering professionals, when appropriate and feasible, software engineers and technologists need to seek quantitative data on which to base decisions, yet also be able to function effectively in an environment of ambiguity and avoid the limitations of oversimplified or unverified "formula-based" modeling.

Though, in the developed countries the demand pushes Software Engineering professionals to become as numerous as all types of engineers combined, developing countries like Ethiopia the ratio is very small. Hence, they are only a small fraction of the total number of engineers. AAiT is convinced to launch this program as a very good way of benefiting the nation from the Software Engineering and Computing market. Besides that, the direction being followed by AAiT goes hand in hand with the nations ICT

roadmap. The roadmap stresses on the principle of leapfrogging rather than begin reactive to the developed nation's technologies. To this end, AAiT launches Artificial Intelligence and Cyber Security programs in addition to the existing engineering programs.

As the center has conducted literature reviews and need assessment with all stakeholders and the university community, it is observed that enclosing the aforementioned four programs into a stream based curriculum is very important. Most reputable engineering schools have started to follow this trend. It is primarily conceived by the ACM. Moreover, the need analysis also indicated that all employers, students, staffs and other stakeholders would benefit from this curricula arrangement as it fulfills the critically low man power in the area of Artificial Intelligence, Cyber Security, Software Engineering and Information Technology with strong computing and engineering background.

Structure of the Program

The structure of undergraduate program in software engineering and computing technology could be defined with three hierarchical categories of academic semesters. The first one would be a pre engineering semester. In this level students will get to know the various fields of engineering, what engineers do, and some preliminary courses before getting to know the core modules; this includes language and writing skills, and other social and humanities courses. Moreover, it's intended to consolidate the students' educational background for higher learning and further help them to adapt to the system.

The second category of the program extends to five semesters and it's where students take all the core courses of Software Engineering and Computing Technology. Hence, students will take courses under Computer Science, Mathematical Computer Science, Information System, Software Engineering, and Electrical Engineering.

In the last two academic years of the program students will have join one of the specialization streams namely: Software Engineering, Information Technology, Artificial Intelligence, and Cyber Security. At the graduation year of each specialization programs, students will undertake thesis. Thesis is the final element of the BSc. study program. Its main goal is to develop graduates creative and problem-solving ability. Students will

receive an individual project specification from various areas and work them out individually under the supervision of faculty advisor or/and professional advisor from the industry. The candidate uses all the theoretical knowledge and practical skill s/he obtained during his/her study to solve the engineering problem in a scientific way which includes problem analysis, solution, verification, implementation, documentation, and presentation. BSc. Thesis is defended in front of the official committee at the end of the study.

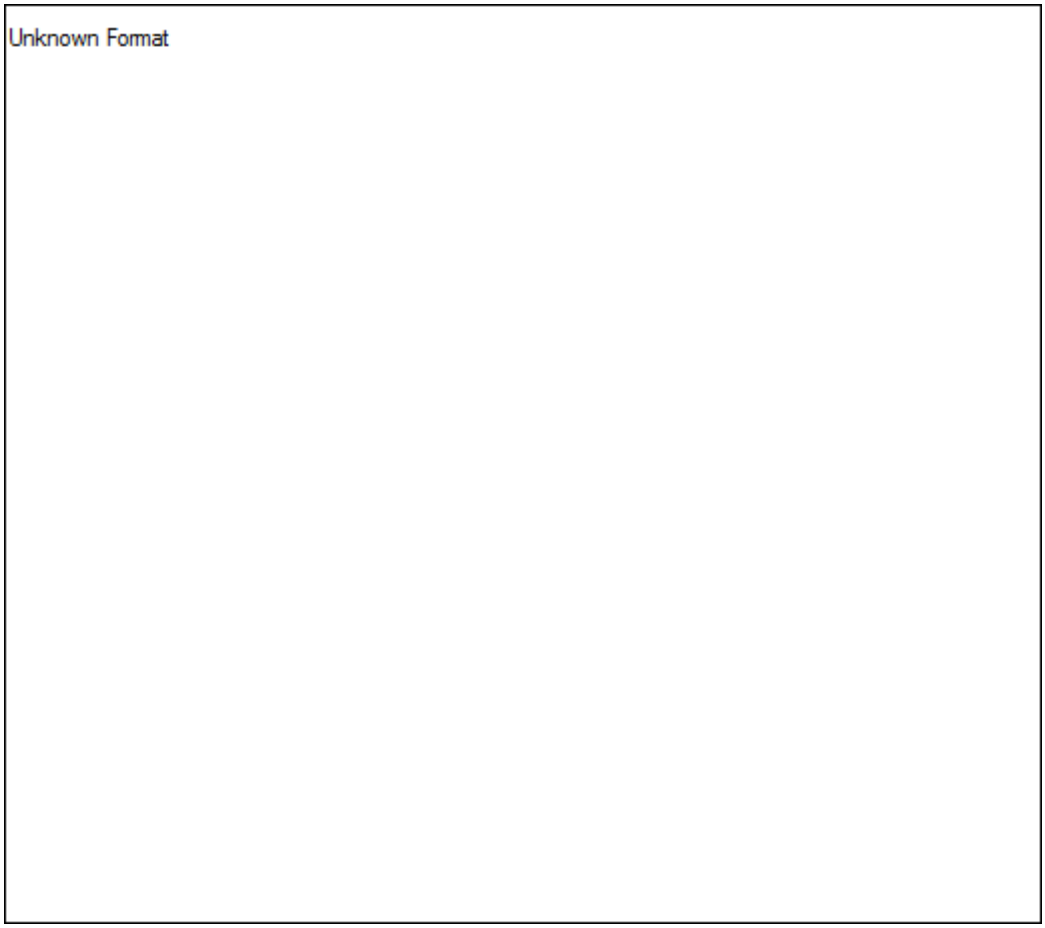


Figure 1: Software Engineering and Computing Technology Curriculum Structure

Objectives of the Program

General Objective

The general objective of the Software Engineering and Computing Technology program is to provide a quality software engineering and computing education with significant

hands-on and laboratory experience that will enable graduates to practice their profession with proficiency and integrity.

Specific Objectives

The following list of salient activities will be undertaken to achieve the general objective of the program:

- To train students to be effective team members, aware of cultural diversity, who conduct themselves ethically and professionally.
- To enhance students communication and technical skills to assure production of quality computer system, on time and within budget.
- Teach students build upon and adapt knowledge of science, mathematics, engineering, and new computing technologies to take on more expansive tasks that require an increased level of self-reliance, technical expertise, and leadership.
- Equip students with the necessary methodologies, skill, and tools to develop and deploy high standard computer systems.
- Teach students to design, develop, test, document and maintain computer systems of an industrial standard for private, non-governmental and governmental clients.
- Train students to conduct research in computing domain such as High Performance Computing, Natural Language Processing, Computer Vision, Dialogue Systems, OCR, Learning Systems, digital forensics, Security policy and risk analysis principles, technologies emerging every day for socio economical social problems and etc.
- Train students with entrepreneurship mindset in technology so that they can create new jobs, or expand existing business.
- Engage students in global online competition so that their understanding meets the global scale.

Program Outcomes

Graduate Profile

The graduate will have the following knowledge, skill and attitude at the end of the program.

Cognitive knowledge

The graduate will:

a) Have an understanding of professional, ethical, legal, security and social issues and responsibilities.

b) Have ability to apply knowledge of computing and mathematics to resolve on-hand and upcoming problems.

c) Have recognition of the need for, and an ability to engage in, continuing professional development and the knowledge and skills to act as research assistants or lecturers in higher education institutions

d) Use current techniques, skills, and tools necessary for software development, testing and maintenance.

e) Have the knowledge and innovative skills to plan, set up and run computer related businesses.

f) Have understanding of best practices and standards and their application;

Practical skills

The graduates will be able to demonstrate an understanding of and apply appropriate theories, models, and techniques that provide a basis for problem identification and analysis, design, development, implementation, verification, and documentation of computer systems.

Attitude (Transferable skill)

The graduate will:

a) Function effectively on teams to accomplish a common goal and work under stressed situation.

b) Communicate effectively with a range of audiences and give clear, good quality verbal presentations on scientific topics

c) Have the knowledge and innovative skills to plan, set up and run computer related businesses.

d) Demonstrate an understanding and appreciation of the importance of negotiation, effective work habits, leadership, and good communication with stakeholders in a typical software development environment.

e) Learn new models, techniques, and technologies as they emerge and appreciate the necessity of such continuing professional development.

Professional Profile

Software Engineers and Computing Technologists must have

- A high-level understanding of systems as a whole. This understanding must go beyond the implementation details of the various components to encompass an appreciation for the structure of computer systems and the processes involved in their construction and analysis.
- Thorough understanding of the balance between theory and practice and the essential link between them not only the theoretical underpinnings of the discipline but also how that theory influences practice
- A solid foundation that allows and encourages them to maintain their skills as the field evolves.

EDUCATIONAL PROCESS

Entry and Admission Requirements

Admission requirements would be similar to the general higher education entry requirement based on higher education of Ethiopia policies and procedures. That is, the student should complete preparatory level education and the entry point is determined accordingly by the Ministry of Education of Ethiopia for the regular students whereas the admission criteria for distance and continuing education program are as per the University's requirement.

Furthermore, a student who has successfully completed the first semester of the pre-engineering program from any of the AAiT departments can opt for Software Engineering and Computing Technology program, subject to further screening and selection.

Exam and Attendance Requirement

The exam requirement for a student to be promoted from one semester to the next is as set in the university legislation.

Graduation Requirement

No Grade "F" in any of the courses and the grade point average (GPA) should be.

- Grade point average requirement

  I/ Cumulative GPA: 2.00

  II/ Major GPA: 2.00
- At least C grades for Industrial Project I and II and Maximum of **310 ECTS and** a minimum of 305 ECTS throughout the course of the program.

Duration of the Study

The B.Sc. program in Software Engineering and Computing Technology extends to five academic years (ten academic semesters).

Degree Nomenclature

After successful completion of all the requirements, a student graduating from the School of Information Technology and Engineering will be entitled to earn a degree with one of the following nomenclatures:

"Bachelor of Science

In

Software Engineering and Computing Technology

(Software Engineering Stream)"

"□□□□□□ □□□□□ □□□

□□□□□□□ □□□□□□□ □□ □□□□□□□□ □□□□□□ (□□□□□□ □□□□□□□ □□□)"

"Bachelor of Science
In
Software Engineering and Computing Technology
(Information Technology Stream)"

"□□□□□□ □□□□□ □□□

□□□□□□□ □□□□□□□ □□ □□□□□□□□ □□□□□□ (□□□□□□□□ □□□□□□ □□□)"

"Bachelor of Science
In
Software Engineering and Computing Technology
(Artificial Intelligence Stream)"

"□□□□□□ □□□□□ □□□

□□□□□□□ □□□□□□□ □□ □□□□□□□□ □□□□□□ (□□□□□□□ □□□□□□□□ □□□)"

"Bachelor of Science
In
Software Engineering and Computing Technology
(Cyber Security Stream)"

"□□□□□□ □□□□□ □□□

Course Coding

All courses at Software Engineering and Computing Technology program should be categorized into Common, Supportive and Core. Designation is possibly applied using two to four letter abbreviations and four figures. The letters (two - four) show the discipline to which the course belongs. Consequently, all supportive and compulsory courses of the software engineering and computing technology program shall be designated by SECT, as an abbreviation for Software Engineering and Computing Technology. However, common courses shall be named according to their belongingness to specific discipline.

The first of the four digits indicates the year in which the course will normally be offered while the second and third digits indicate the module to which the course belongs to. The fourth digit is used to denote the course sequence in its module.

Moreover, the Modules are coded using Four digit number and prefixed with module home base and the letter M. Module that belong to SECT are prefixed with "SECT" and the number coding succeeds the letter "-M" stating that it's module. The first number in the module code states the year the module starts. The second and third digits state the module number in the curriculum. The last digit states the type of the module; 1 is used to denote Core modules, 2 are used to denote supportive and common courses are denoted by 3.

Modularization

Modularization is provided by collecting and grouping courses that can help to attain specific competence to accomplish a certain task at intermediate level after its completion. It is implemented as an alternative after piloting it with a few selected courses. The following Table shows list of modules and courses under each module.

Modules

Common and Supportive Modules - Year I

Category	Module Number	Module Name	Module Code	Module ECTS	Course No.	Course Code	Module Courses
Basic	01	Basic English Skills	*FLEn–M1013*	10	1	FLEn-1011	Communicative English Language Skill I
					2	FLEn-1012	Communicative English Language Skill II
	02	Moral and Civic Education	*MCiE-M1023*	4	1	MCiE-1012	Moral and Civic Education
	03	Logic and Critical Thinking	*LoCT–M1033*	5	1	LoCT-1011	Critical Thinking
	04	General Natural Science and Social Science	GNSS-M1043	38	1	Math 1011	Mathematics for Natural Sciences
					2	Psch 1011	General Psychology
					3	Phys 1011	General Physics
					4	Anth1012	Social Anthropology
					5	SpSc 1011	Physical fitness
					6	MGMT1012	Entrepreneurship
					7	EmTe 1012	Introduction to Emerging Technologies
					8	GeES 1011	Geography of Ethiopia and the Horn
					9	SNIE 1012	Inclusiveness
					10	Econ 1011	Economics
					11	GlTr 1012	Global Trends
					12	Stat-2171	Probability and Statistics for Engineers

	05	Applied Mathematics	*Math–M1053*		1	Math-1041	Applied Mathematics I
					2	Math-2041	Applied Mathematics II
					3	Math-2042	Applied Mathematics III
	06	Engineering Mechanics	CEng-M1053	5	1	CEng-2051	Engineering Mechanics I - Statics
	07	Basic Engineering Skills	GENG-M11063	10	1	Phil 3011	Philosophy of Science and Ethics
					2	MEng-2061	Engineering Drawing

Core Modules - Year I, II, and III

Category	Module Number	Module Name	Module Code	Module ECTS	Course No.	Course Code	Module Courses
Core	08	Computer Systems and Programming	SECT-M1081	12	1	SECT-1082	Computer Programming
					2	SECT-3082	Operating System and s System Programming
	09	Computer Science	SECT-M1091	12	1	SECT-3091	Fundamental of Data structure and analysis
					2	SECT-3092	Discrete Mathematics
	10	Programming	SECT-M2091	5	1	SECT-2101	Object Oriented Programming
	11	Software Development Fundamentals	SECT-M2101	19	1	SECT-2111	Fundamentals of Software Engineering
					2	SECT-3112	Web Design and Programming
					3	SECT-3113	Mobile Application Development
	12	Computer Architecture and Design	SECT-M2111	12	1	SECT-2121	Fundamental of Electrical circuits and Electronics
					2	SECT-3122	Computer Organizations and Architecture
	13	Graphics and HCI	SECT-M3121	10	1	SECT-3131	Human Computer Interaction

					2	SECT-3132	Computer Graphics
	14	Security Fundamentals	SECT-M3131	5	1	SECT-3141	Fundamentals of Cybersecurity
	15	Artificial Intelligence	SECT-M3111	5	1	SECT-3151	Fundamentals of Artificial Intelligence
	16	Database	SECT-2161	5	1	SECT-2161	Fundamentals of Database
	17	Networking	SECT -2171	5	1	SECT-2171	Fundamentals of Networking

Core Modules - Year IV and V

Software Engineering Stream

Category	Module Number	Module Name	Module Code	Module ECTS	Course No.	Course Code	Module Courses
Core	18	System Modeling	SECT-M4181	7	1	SECT-4181	Requirement Engineering, Architecture and Design
	19	Software Quality Assurance	SECT-M4191	7	1	SECT-5191	Software Testing, Verification and Quality Assurance
	20	Selected Topics in Software Engineering	SECT-M5201	5	1	SECT-5201	Selected Topics in Software Engineering
	21	High Performance Computing	SECT-M4141	12	1	SECT-4141	Fundamentals of Distributed Systems
					2	SECT-5192	Fundamentals of Parallel Computing
	22	Advanced Development	SECT-M5201	19	1	SECT-5221	Advanced Database Systems
					2	SECT-4221	Enterprise Application Development
	23	Software Project Management	SECT-M4231	5	1	SECT-4231	Software Project Management
	24	Pervasive Computing	SECT-M5241	5	1	SECT-M5241	Pervasive Computing

Core Modules - Year IV and V

Information Technology Stream

Category	Module Number	Module Name	Module Code	Module ECTS	Course No.	Course Code	Module Courses
Core	25	Enterprise Systems and Network Security	SECT-M4251	12	1	SECT-4211	Enterprise Systems and Network Administration
					2	SECT-5212	Software Security
	26	Selected Topics in Information Technology	SECT-M5261	4	1	SECT-5261	Selected Topics in Information Technology
	27	System and Database Administration	SECT-M4271	7	1	SECT-4271	Database Administration and Security
	28	eBusiness	SECT-M4251	5	1	SECT-4281	E-business Strategy and Development
	29	Data Analytics	SECT-M4291	5	1	SECT-4291	Data Mining and Analytics
					2	SECT-5291	Information Retrieval

Core Modules - Year IV and V

Artificial Intelligence Stream

Category	Module Number	Module Name	Module Code	Module ECTS	Course No.	Course Code	Module Courses

Core	30	Perception and NLP	SECT-M5261	17	1	SECT-4301	Introduction to Robotics
					2	SECT-5302	Computer Vision
					3	SECT-5303	Machine Intelligence
	31	Selected Topics in AI	SECT-M5311	4	1	SECT-5311	Selected Topics in AI
	32	Computational Analysis and Optimization	SECT-M4321	15	2	SECT-4321	Social Network Analysis
	33	Adaptive Computation and Machine Learning	SECT-M4331	37	1	SECT-4331	Machine Learning
					2	SECT-5332	Deep Learning
					3	SECT-5333	Reinforcement Learning
	34	Computing of AI	SECT-M4341	10	1	SECT-4341	Mathematics for AI
					2	SECT- 4342	Programming for AI

Core Modules - Year IV and V

Cyber Security Engineering Stream

Category	Module Number	Module Name	Module Code	Module ECTS	Course No.	Course Code	Module Courses
Core	35	Security and Risk Management	SECT-M5301	25	1	SECT-5351	Information Systems Security, Audit and Compliance
	36	Cyber Forensics	SECT-M5361	5	1	SECT-5361	Computer Crime and Digital Forensics
	37	Cryptography	SECT-M4371	5	1	SECT-4371	Cryptographic Concepts and Applications
	38	Security Engineering	SECT-M5381	25	1	SECT-4381	Enterprise Systems and Network Security
					2	SECT-5381	Ethical Hacking and Penetration Testing
	39	Software Security	SECT-M5391	5	1	SECT-5391	Software Security
	40	Selected Topics in Cyber security	SECT-M5401	5	1	SECT-5401	Selected Topics in Cyber security

Common Modules for Year IV and V

Category	Module Number	Module Name	Module Code	Module ECTS	Course No.	Course Code	Module Courses
	41	High performance Computing	SECT-M4141	10	1	SECT - 4141	Fundamentals of Distributed Systems
					2	SECT - 4192	Fundamentals of Parallel Computing
	42						
	43	Industry Internship	SECT-M4431	30	1	SECT-4431	Industry Internship
	44	Industrial Project	SECT-M4391		1	SECT - 5441	Project I
					2	SECT - 5442	Project II
	45	Machine Learning and big data	SECT-M4451	5	1	SECT-4451	Machine Learning and big data
	46	Cloud Computing Security	SECT-M4461	5	1	SECT-M4461	Cloud Computing Security

Electives

Category	Module No.	Module Name	Module Code	Course Code	Module Courses (Clustered under the module)
Electives	47	System and Database Administration	SECT-M5471	SECT - 5471	Website Administration
	48	Computational Analysis and Optimization	SECT-M5481	SECT - 5481	Statistical Methods and Data Analysis
	49	Adaptive Computation and Machine Learning	SECT-M5491	SECT– 5492	Computational Genomics
				SECT– 5491	Information Retrieval
				SECT - 5493	Introduction to Cognitive Science
	50	Perception and NLP	SECT-M5501	SECT-5501	Natural Language Processing
				SECT-5502	Introduction to Robotics
	51	Security Risk Management	SECT-M5511	SECT-5511	Mobile Application Security
				SECT-5512	Wireless Security
				SECT-5513	Firewall and Perimeter Security
	52	Security Analysis and Administration	SECT-M5521	SECT-5521	IT Governance, Risk & Information Security Management
				SECT-5522	Malware Analysis
				SECT-5523	Reverse Engineering
				SECT-5524	Data Analytics for Cyber Security
	53	Security Engineering	SECT-M5531	SECT-5531	Information Assurance Architectures
				SECT-5532	Secure System Design
	54	Cloud Computing and IOT	SECT-M5541	SECT-5541	Internet of Things
				SECT-5542	Cloud Computing and Security
	55	Compiler Design	SECT-M5551	SECT-5551	

	56	Wireless security	SECT-M5561	SECT-5561	Wireless security
		Firewall and Perimeter Security		SECT-5562	Firewall and Perimeter Security
		System and Network Security Administration		SECT-5563	System and Network Security Administration
		Data Analytics for Cyber Security		SECT-5564	Data Analytics for Cyber Security
	57	Basic Engineering Mechanics	GEng-M2053	MEng 2052	Engineering Mechanics II

Note:-

All Electives have 5 ECTS.

Teaching-Learning Methods and Materials

Learning should not be closed to one specific techniques or method; it should be integrative in nature which will allow teachers the freedom and learners the options to go through individually suited learning style that goes with the personality of the learner. So, for this Program suitable to use the integrated approach which include Student Center Learning, Parallel Curriculum Model (PCM), Accelerated Learning (AL) and Brain Target Teaching Model (BTT).

To implement the above hybrid teaching methodology the following teaching techniques will be employed:

- Course Lecture (Enhanced with active learning techniques such as think-pair-share, one minute paper)
- Brainstorming
- Case studies
- Presentations
- Debates
- Demonstrations
- Discussions/Panels
- Films/Video/TV Presentations
- Games/puzzles
- Mind mapping

- Role playing/Skits
- Structured and Unstructured laboratory classes
- Field trips and industrial visits
- Homework and Assignments
- Individual and group projects
- Interviews with IT professionals and/or job shadowing
- Preparation and presentation of a technical report
- Paper and article review
- Thesis

Assessment Methods

This program emphasizes on implementing continuous assessment during the teaching learning process. The following methods will be used to assess students' knowledge and ability. The instructor can extend his/her way of evaluating students while having the basic mechanisms in place.

- Tests
- Assignments
- Projects

- Laboratory exams
- Final Exams
- Paper Review/ Presentations

Grading Scheme

Raw Mark Interval [100%]	Corresponding Fixed Number Grade	Corresponding Letter Grade	Status Description	Class Description
[90, 100]	4.0	A+	Excellent	First class with Great Distinction
[83, 90)	4.0	A		
[80, 83)	3.75	A-		
[75, 80)	3.5	B+	Very Good	First class with Distinction
[68, 75)	3.0	B		
[65, 68)	2.75	B-	Good	First Class
[60, 65)	2.5	C+		
[50, 60)	2.0	C	Satisfactory	Second Class
[45, 50)	1.75	C-	Unsatisfactory	Lower Class
[40, 45)	1.0	D	Very Poor	Lower Class
[30, 40)	0	Fx	Fail	Lower Class
[<30)	0	F	Fail	Lower Class

Quality Assurance

Quality assurance can be used as an instrument to improve the system continuously, install accountability and enable compatibility with other higher education systems. The

quality of the program is assessed in terms of the instruction performance and impact of program on the quality of graduates looking for a job or further studies. The following factors help to ensure the quality of the education:

- In order to maintain the level of courses, course description are prepared in detail containing course objectives, learning outcome, course outline, textbooks, references and laboratory tasks.
- With regard to instruction performance, in line with the University policy, student evaluations are taken into account. The feedbacks from the students are used to improve the quality of instruction.
- Examinations are assessed by Departmental Examination Committee to verify the level of the courses instruction.
- Industry feedback will be an integral part of the process of quality assurance. The quality of graduates is measured with the feedback obtained from the employers and stakeholders who are the immediate beneficiary of the program and also the graduates who are able to rate their own confidence in meeting the challenges they encounter after graduation.

Resource Profile of the School

Permanent Staff - Professors

No.	Name	Specialization	Level	
1	Dr. Elefelious Getachew	Software Engineering	Assistant Professor	
2	Dr. G. Manjunath	Networks and Cloud Computing	Assistant Professor	
3	Dr. Mani Sarma Vittapu	Data Mining and Image Processing	Assistant Professor	
4	Dr. Venkateswarlu Sunkari	Software Engineering and Cloud Computing	Assistant Professor	
5.	Dr. Henok Mulugeta	Cybersecurity	Assistant Professor	
6.	Dr. Sileshe Demssie	Cybersecurity	Assistant Professor	
7.	Dr. Beakal Gizachew	AI	Assistant Professor	

Permanent Staff - Lecturers

No.	Name	Specialization	Level	
1.	Mr. Abreham Getachew	Information Systems	Lecturer	
2.	Mr. Betesgaw Lemma	Cloud Computing	Lecturer	
3	Mr. Daniel Abebe	Computer Science	Lecturer	PhD Candidates
4	Endrias Haile	Computer Science	Lecturer	
5	Mr. Eyob Gebretinsae	Computer Science	Lecturer	PhD Candidates
6	Mr. Eyob Wondimkun	Computer Science	Lecturer	
7	Mr. Fitsum Alemu	Computer Science	Lecturer	
8	Mr. Kassahun Abdissa	Information Science	Lecturer	PhD Candidates
9	Mr. Lisanu Tebikew	Computer Science	Lecturer	
10	Mr. Meareg Abreha	Computer Science	Lecturer	
11	Mr. Natnael Argaw	Computer Science	Lecturer	
12	Mr. Nebiat Fikru	Computer Science	Lecturer	
13	Mr. Tigabu Dagne	Computer Science	Lecturer	PhD Candidates
14	Mr. Wondemagegne Desta	Computer Science	Lecturer	PhD Candidates
15	Mr. Yoseph Abate	Computer	Lecturer	

		E n g i n e e r i n g / Software Engineering		

Permanent Staff - Assistant Lecturers

No.	Name	Specialization	Level
1	Minase Mekete	Computer Science	Assistant Lecturer
2	Eskinder Abebe	Computer Science	Assistant Lecturer
3	Sofanit Wubeshet	Computer Science	Assistant Lecturer
4	Yemane Teklay	Computer Science	Assistant Lecturer
5	Abreham Ararsa	Computer Engineering	Assistant Lecturer
6	Belete Tekle	Computer Science	Assistant Lecturer
7	Amanuel Negash	Software Engineering	Assistant Lecturer
8	Tewodros Wondifraw	Software Engineering	Assistant Lecturer
9	Michael Beyene	Information Technology	Graduate Assistant II

Visiting Professors

No.	Name	Specialization	Level
1.	Dr. Ben Goertzel	Mathematics, AGI and Cognitive Science	Assistant Professor
4	Dr.-Ing. Getahun Mekuria	Communication Engineering	Assistant Professor
5	Dr. Yaregal Asabe	NLP/ Image Processing	Assistant Professor
6	Dr. Fekade Getahun	Information Science	Assistant Professor
7	Dr. Million Meshesha	Information Science	Assistant Professor
8	Dr. Ruiting Lian	Computational Linguistics	Assistant Professor
9	Dr. Nil Geisweiller	Machine Learning, Evolutionary Learning, Probabilistic Reasoning	Assistant Professor
10	Dr. David Hanson	Robotics	Assistant Professor
11	Dr. Hugo de Garis	Neural Networks	Professor

12	Mr. Keyvan Sadeghi	Bayesian Networks, Temporal and Spatial Reasoning	Assistant Professor
13	Dr. Ted Goertzel	History and Future Implications of AI	Assistant Professor

Technical Assistants

No.	Name	Specialization	Level
1	Mr. Eliyas Dessie	Electronics Technician	Technical Assistant III
2	Mr. Seifesillase Hailu	Computer Science	Technical Assistant III
3	Mr. Adane Tegegne	Computer Science	Technical Assistant III

Appendix 1A: Course Schedule

Software Engineering and Computing Technology Program

Year I: Semester I

First Year - Semester I						
Course Code	**Course Title**	**ECTS/ Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
FLEn-1011	Communicative English Language Skill I	5	32	48	0	
Math-1011	Mathematics for Natural Sciences	5	32	48	0	
LoCT 1011	Critical Thinking	5	32	48	0	
Psch 1011	General Psychology	5	32	48	0	
Phys 1011	General Physics	5	32	16	48	
SpSc 1011	Physical Fitness	-	32	-	-	
GeES – 1011	Geography of Ethiopia and the Horn	5	32	48	0	
Total		**30**				

Year I: Semester II

First Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
FLEn-1012	Communicative English Language Skill II	5	32	48	0	
Anth 1012	Social Anthropology	4	32	16	0	
Math 1041	Applied Mathematic I	5	32	48	0	
MGMT1012	Entrepreneurship	5	32	48	0	
EmTe-1012	Introduction to Emerging Technologies	5	32	16	48	
MCiE 1012	Morale and Civic Education	4	32	16	0	

SECT 1082	Computer Programing	5	32	16	48	
Total		**33**				

Year II: Semester I

Second Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
MEng-2061	Engineering Drawing	5	32	16	48	
CEng- 2051	Engineering Mechanics I (Statics)	5	32	48	0	
Stat-2171	Probability and Statistics for Engineers	5	48	32	0	
Math-2041	Applied Mathematics II	7	48	48	0	
GITr-1012	Global Trends	4	32	16	0	
Econ-1011	Economics	5	32	48	0	
SNIE-1012	Inclusiveness	4	32	16	0	
Total		**35**				

Year II: Semester II

Second Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
Math-2042	Applied Mathematics III	7	48	48	0	
SECT-2171	Fundamental of Networking	*5*	32	16	48	
SECT-2161	Fundamental of Database	5	32	16	48	
SECT-2101	Object Oriented Programming	5	32	16	48	
SECT-3092	Discrete Mathematics	5	32	16	48	
SECT-3091	Fundamental of Data Structure and Algorithm Analysis	7	48	32	48	
		34				

Year III: Semester I

Third Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT-2121	Fundamentals of Electrical Circuits and Electronics	5	32	16	48	
SECT-2111	Fundamentals of Software Engineering	5	32	16	48	
SECT-3122	Computer Organizations and Architecture	7	48	32	48	
SECT-3131	Human Computer Interaction	5	32	16	48	
SECT-3112	Web Design and Programming	7	48	32	48	
		29				

Year III: Semester II

Third Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT-3151	Fundamentals of AI	5	32	16	48	
SECT-3141	Fundamentals of Cybersecurity	5	32	16	48	
SECT-3132	Computer Graphics	5	32	16	48	
SECT-3113	Mobile Application Development	7	32	16	48	
SECT-3082	Operating Systems and System Programming	7	48	32	48	
	Total	**29**				

Software Engineering Stream

Year IV: Semester I

Fourth Year - Semester II

Course Code	Course Title	ECTS /Week	Lec.	Tut.	Lab	Home Study
SECT -4181	Requirement Engineering, Architecture and Design	7	48	32	48	
SECT -4451	Machine Learning and Big data	5	32	16	48	
SECT - 4141	Fundamentals of Distributed Systems	7	48	32	48	
SECT - 4231	Software Project Management	5	32	16	32	
SECT - 4221	Enterprise Application Development	7	48	16	48	
	Total	**31**				

Year IV: Semester II

Fourth Year - Semester I						
Course Code	Course Name	**ECTS /Week**	Lec.	Tut	Lab	Home
SECT-4431	Industry Internship	30	0	0	400	350
Phil 3011	Philosophy of Science and Ethics	5	32	16	0	
	Total	35				

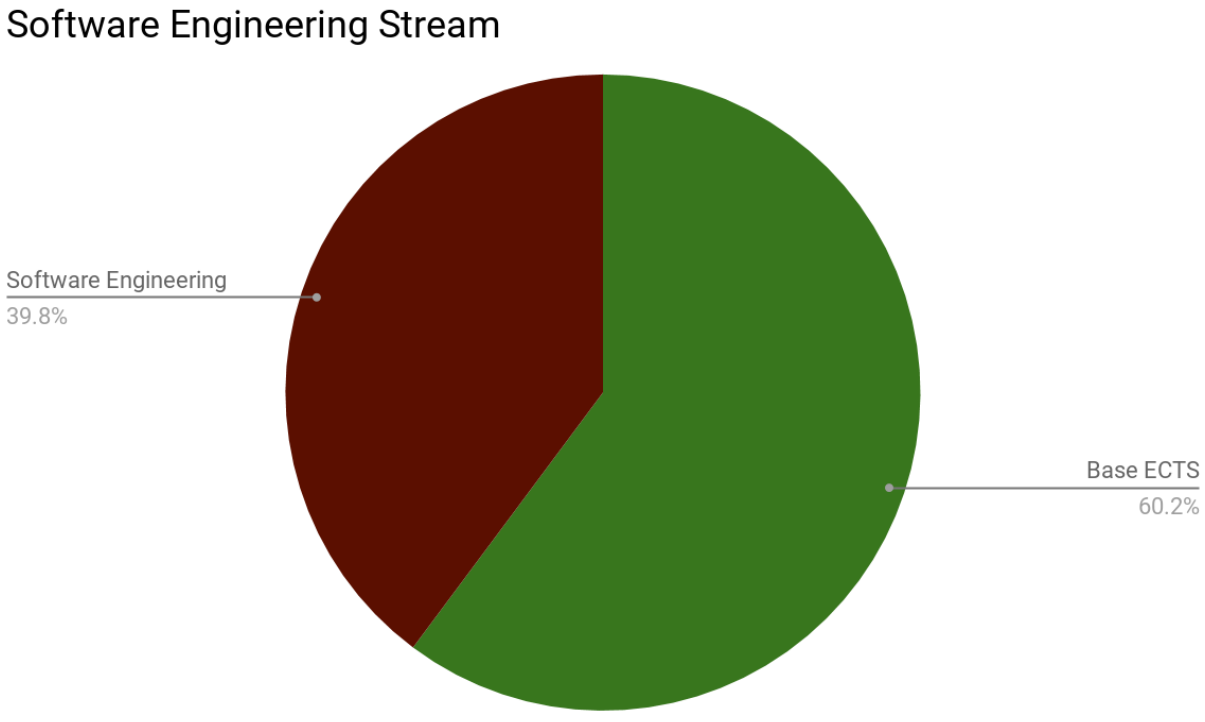
Year V: Semester I

Fifth Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT -5221	Advanced Database Systems	7	48	32	48	
SECT - 5XXX	Elective I	5				
SECT - 5241	Pervasive Computing	5	32	16	48	
SECT - 5441	Project I	7	0	0	128	
	Total	**24**				

Year V: Semester II

Fifth Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT-4461	Cloud Computing and Security	5	32	16	32	
SECT- 5191	Software Testing, Verification and Quality Assurance	7	48	16	48	
SECT - 5XXX	Elective II	5				
SECT- 5201	Selected Topics in Software Engineering	5	32	16	32	
SECT- 5442	Project II	7	0	0	128	
	Total	**29**				

Summary of SE Stream ECTS



Information Technology Stream

Year IV: Semester I

Fourth Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT- 4211	Enterprise Systems and Network Administration	7	48	32	48	
SECT - 4271	Database Administration and Security	7	32	16	48	
SECT - 4291	Data Mining and Analytics	5	32	16	48	
SECT - 4231	Software Project Management	5	32	16	48	
SECT - 4281	E-business Strategy and Development	5	32	16	32	
	Total	**29**				

Year IV: Semester II

Fourth Year - Semester I						
Course Code	**Course Name**	**ECTS /Week**	**Lec.**	**Tut**	**Lab**	**Home**
SECT-4431	Industry Internship	30	0	0	400	350
Phil 3011	Philosophy of Science and Ethics	5	32	16	0	
	Total	35				

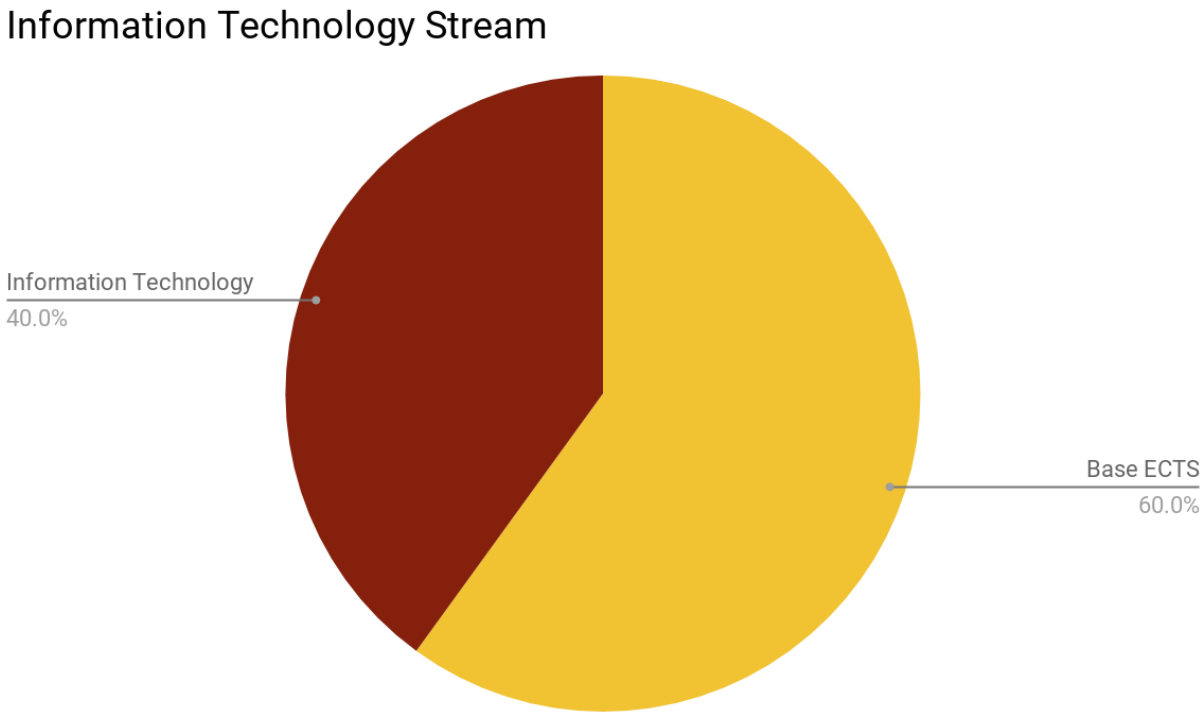
Year V: Semester I

Fifth Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT - 4141	Fundamentals of Distributed Systems	7	48	32	48	
SECT-4461	Cloud Computing and Security	5	32	16	32	
SECT - 5XXX	Elective I	5	32	16	48	
SECT - 5291	Information Retrieval	5	32	16	32	
SECT - 5441	Project I	7	0	0	128	
	Total	**29**				

Year V: Semester II

Fifth Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT- 5381	Ethical Hacking and Penetration Testing	7	32	16	32	
SECT - 5261	Selected Topics in Information Technology	5	32	16	32	
SECT - 5XXX	Elective II	5	32	16	48	
SECT - 5442	Project II	7	0	0	128	
	Total	**24**				

Summary of Information Technology Stream ECTS



Artificial Intelligence Stream

Year IV: Semester I

Fourth Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT - 4341	Mathematics for AI	5	32	16	48	
SECT - 4231	Software Project Management	5	32	16	48	
SECT - 4321	Social Network Analysis	5	32	16	48	
SECT - 4301	Introduction to Robotics and Intelligent Systems	5	32	16	48	
SECT - 4331	Machine Learning	7	48	32	48	
SECT - 4342	Programming for AI	5	32	16	32	
	Total	**32**				

Year IV: Semester II

Fourth Year - Semester II						
Course Code	Course Name	**ECTS /Week**	Lec.	Tut	Lab	Home
SECT-4431	Industry Internship	30	0	0	400	350
Phil 3011	Philosophy of Science and Ethics	5	32	16	0	
	Total	35				

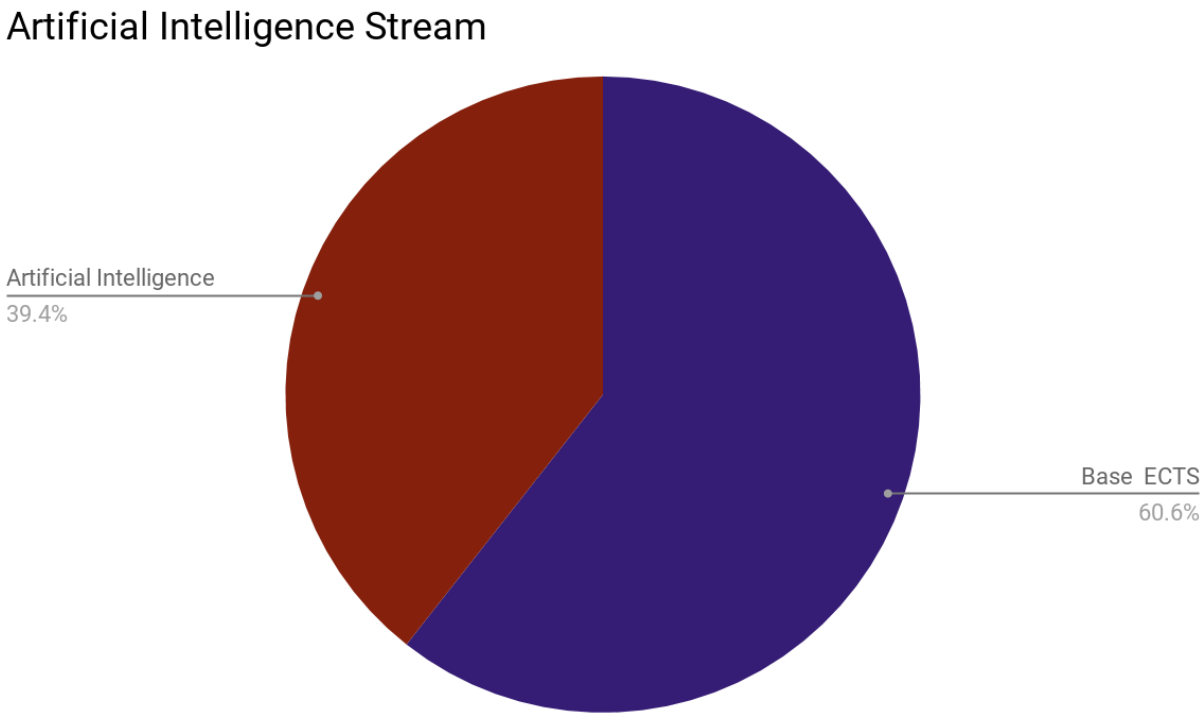
Year V: Semester

Fifth Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT - 5311	Selected Topics in AI	5	32	16	32	
SECT - 5XXX	Elective I	5	32	16	48	
SECT - 5441	Project - I	7	48	32	48	
SECT - 5332	Deep Learning	5	32	16	32	
SECT - 5303	Machine Intelligence	5	32	16	32	
	Total	**27**				

Year V: Semester II

Fifth Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT - 5302	Computer Vision	7	32	16	32	
SECT - 5333	Reinforcement Learning	5	32	16	48	
SECT - 5442	Project II	7	48	32	48	
SECT - 5XXX	Elective II	5	32	16	32	
	Total	**24**				

Summary of Artificial Intelligence Stream ECTS



Cyber Security Stream

Year IV: Semester I

Fourth Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT - 4381	Enterprise Systems and Network Security	7	48	32	48	
SECT - 4371	Cryptographic Concepts and Applications	5	32	16	48	
SECT-4451	Machine Learning and big data	5	32	16	32	
SECT - 4351	Software Project Management	5	32	16	48	
SECT-4461	Cloud Computing Security	5	32	16	48	
	Total	**27**				

Year IV: Semester II

Fourth Year - Semester I						
Course Code	Course Name	**ECTS /Week**	Lec.	Tut	Lab	Home
SECT-4431	Industry Internship	30	0	0	400	350
Phil 3011	Philosophy of Science and Ethics	5	32	16	0	
	Total	35				

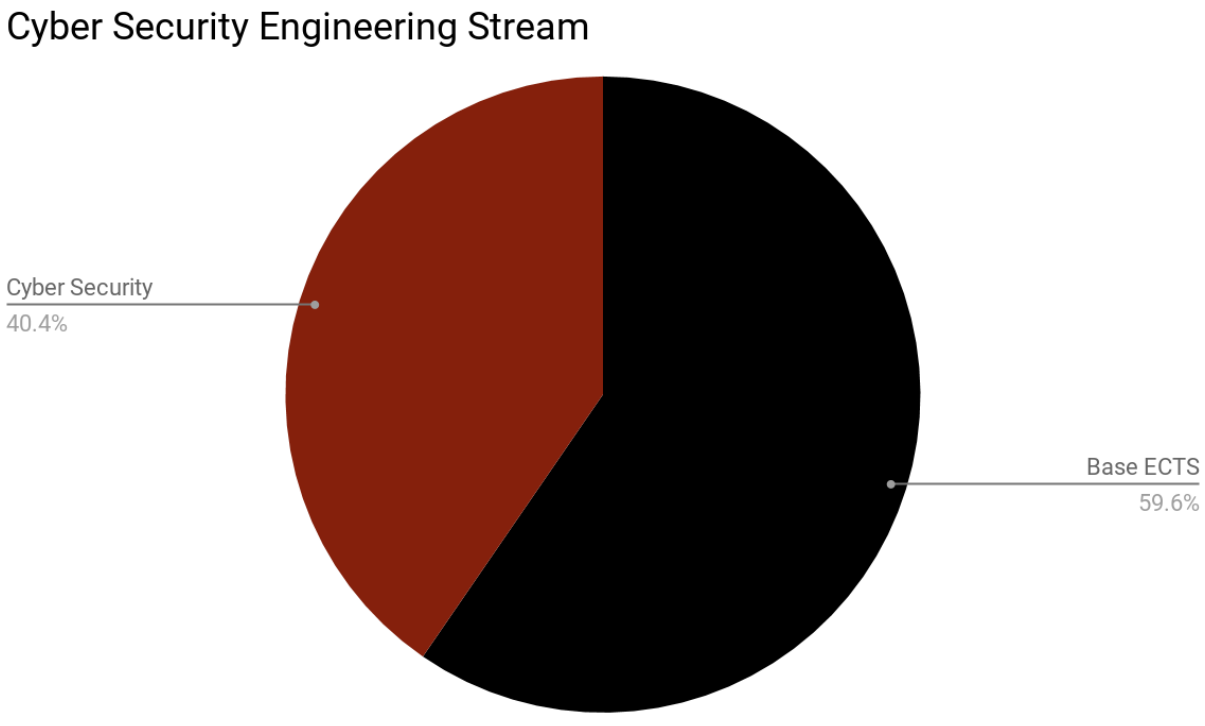
Year V: Semester I

Fifth Year - Semester I						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT- 5381	Ethical Hacking and Penetration Testing	7	48	32	48	
SECT - 5361	Computer Crime and Digital Forensics	5	32	16	48	
SECT - 5391	Software Security	5	32	16	48	
SECT - 5XXX	Elective I	5	32	16	48	
SECT - 5441	Project I	7	48	32	48	
	Total	**29**				

Year V: Semester II

Fifth Year - Semester II						
Course Code	**Course Title**	**ECTS /Week**	**Lec.**	**Tut.**	**Lab**	**Home Study**
SECT - 5351	Information Systems Security, Audit and Compliance	7	32	16	48	
SECT - 5XXX	Elective II	5	32	16	48	
SECT - 5401	Selected Topics in Cyber security	5	32	16	48	
SECT - 5442	Project II	7	48	32	48	
	Total	**24**				

Summary of Cyber Security Stream ECTS



Appendix 1B-Software Engineering and Computing Technology Syllabuses

Year I - Year III

SEM I

Communication Skill (English)

Program: Software Engineering and Computing Technology	
Module Title: **Basic English Skills**	Module Code: **EnLa-M11013**
	Module ECTS: 10
Course Title: **Communication Skill (English)**	Course Code: **EnLa-1011**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description
The aim of this course is to develop students' basic communication skills in the context that they will most need those skills: graduate school. Within the context of going abroad to present a paper on their graduate research, students will learn skills needed for traveling (e.g. asking for/giving directions, making reservations), negotiations, survey taking, and problem solving, as well as be introduced to skills involved in making a presentation at a conference. Additionally, students will learn to start and continue a conversation naturally, using a number of communication strategies such as asking follow-up questions and giving extended answers. They will also learn about turn taking and how to control the flow of a conversation by adding information. Finally, writing skills will be practiced with a short essay using the Online Homework Submission and Evaluation System.
Course Objectives

The course prepares the inexperienced writer for college level reading and academic writing by focusing on critical thinking, reading, writing, and study skills necessary to succeed at the academic level

Learning Outcomes

The program provides a knowledge and understanding of the following:

After completing this course students will:

- understand and use different skills and strategies to read
- understand the meaning of what is read
- read different materials for a variety of purposes
- write clearly and effectively
- understand and use the steps of the writing process
- analyze and evaluate the effectiveness of written work
- analyze and evaluate the effectiveness of formal and informal communication
- communicate ideas clearly and effectively

Prerequisites:

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study: 61 hrs.**	**Laboratory:**

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

- Continuous Assessment with Assignment, Tests and Projects (50%),
- Final examination (50%)

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

References:

- Writing with Contemporary Readings by Baker, Barbara and Catherine Baker
- Text & Thought: An Integrated Approach to College Reading and Writing by Lester, Lanny, and Judith Resnick
- The Writer's Response: A Reading-Based Approach to College Writing by McDonald, Stephen, and William Salomone

Program: Software Engineering and Computing Technology	
Module Title: **Basic Engineering Skills**	Module Code: **MEng-1062**
	Module ECTS: 10
Course Title: **Engineering Drawing**	Course Code: **MEng-1062**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

Students are introduced to fundamental knowledge and skills such as line work, lettering, scale use, and sketching, multiview drawings, sectional views, with the basics of manual drafting techniques and the use of drafting equipment, auxiliary views, descriptive geometry, patterns and developments

Course Objectives

At the end of the course, students shall master:

- Different types of projection techniques.
- Sketching multiview drawings.
- Pictorial drawings of given multi-view drawings.
- Sketching auxiliary and sectional views.
- Finding intersection lines of different geometries & development of surfaces.
- Theory of projections.
- Multi-view drawings.
- Pictorial drawings.
- Auxiliary views.
- Sectional views.
- Intersection and development.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Knowledge on the fundamentals of Engineering drawing; ability to develop and/or comprehend a simple engineering drawing in both First and Third angle orthographic projections, international standards in engineering drawing practice and engineering graphics. A fundamental knowledge on computer aided graphics. Ability of freehand sketching. Developing cognitive and psychomotor skills; visualization of images and their dimensions.

Prerequisites:			
Student Workload			
Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study: 61 hrs.**	**Laboratory:**

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

- **Continuous Assessment with Assignment, Tests and Projects (50%),**
- **Final examination (50%)**

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

David A. Madsen and David P. Madsen, Engineering Drawing and Design (MindTap Course List), 6th Ed.

Engineering Professions *

Program: Software Engineering and Computing Technology	
Module Title: **Basic Engineering Skills**	Module Code: **GENG-1061**
	Module ECTS: 10
Course Title: **Introduction to Engineering Profession**	Course Code: **GENG-1061**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

- Interrelations and distinction among engineering, science and technology - R&D in engineering;
- Technical professional levels, qualifications and duties & responsibilities of technicians, technologists and engineers (carrier structures and levels);
- Engineers in academics and in industrial professions - B.Sc. and B.Eng., M.Sc. and M.Eng.
- Engineering disciplines - civil, mechanical, chemical, industrial, manufacturing, electrical, computer, biomedical, aeronautics and space;
- Specific roles of electrical engineers in the engineering profession;
- Engineering professional ethics and moral issues - engineering for civil and military applications;
- Normative interaction of engineers with other professionals as well as art and business;
- Engineers as leaders and managers;
- Engineering professional associations

Course Objectives

To enable students to understand the constructive interrelation of natural & social sciences as

well as business and art to engineering and their positive impact on the socio economic aspect of a society.

Content			
Prerequisites:			
Student Workload			
Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study: 61 hrs.**	**Laboratory:**

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

- **Continuous Assessment with Assignment, Tests and Projects (50%),**
- **Final examination (50%)**

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

Engineering Mechanics (stat)

Applied Mathematics I

Program: Software Engineering and Computing Technology	
Module Title: *Applied Mathematics*	Module Code: Math-M1043
	Module ECTS: 24
Course Title: *Applied Mathematics I*	Course Code: Math-1041
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 20_ _ Semester: I / II / III Meeting day: Meeting time: Meeting location:

Course Description
Algebra and trigonometry, including algebraic fractions, systems of linear equations, quadratic equations, literal equations, word problems and their solutions, right triangles, and vectors. Applications will be stressed. Prerequisite: High School Algebra II.
Course Objectives
Students shall understand the principles of vector and scalars, definition and operation of matrices & determinants, basics of limit and continuity, basic rules of derivatives & their applications, integrals, integration techniques and their application in volume, arc length, and surface area determination.
Learning Outcomes The program provides a knowledge and understanding of the following:
1. Apply mathematical concepts and principles to perform computations 2. Apply mathematics to solve problems 3. Create, use and analyze graphical representations of mathematical relationships 4. Communicate mathematical knowledge and understanding 5. Apply technology tools to solve problems 6. Perform abstract mathematical reasoning 7. Learn independently

Prerequisites: Applied Mathematics I

Student Workload

Lecture:48 hrs.	**Tutorial: 32 hrs**	**Home Study: 61 hrs.**	**Laboratory: 48hrs**

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Exam – 70%
- Assignments – 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- R. Ellis: Calculus with Analytic Geometry
- S.Lang: Linear Algebra
- Demissu Gemeda: An Introduction to Linear ALGEBRA
- E.W. Swokowski: Calculus
- J. Stewart, Calculus

Department: Software Engineering	
Module Title: **Civics and Ethical Education**	Module Code: **CESt-M11023**
	Module ECTS: 5
Course Tile: ***Civics and Ethical Education***	Course Code: **CESt-1023**
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 1 Semester: 2 Meeting day: Meeting time: Meeting location:

Course Description
This course attempts to introduce the fundamental concepts of • State, Government and Citizenship • Learning about Constitutions • Constitutional experience in Ethiopia • Morality, Ethics and Civic virtues
Course Objectives
Students shall learn about: • Essence and origins of the state and Government • The concept of citizens and citizenship • Origins, essence and forms of constitution • The ideas of Morality, Ethics and Civic virtues
Prerequisites: None
Assessment Method:
Assessment type

Labs (20 %)

Assignment and Projects (30 %)

Mid exam (20 %)

Final exam (30 %)

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- What Is Democracy? by Touraine, Alain
- Globalizing Democracy: Power, Legitimacy, and the Interpretation of Democratic Ideas by Fierlbeck, Katherine

SEM II

Fundamental of Computer Systems

Department: Software Engineering	
Module Title: **Computer Systems and Programming**	Module Code: **SITE-M1071**
	Module ECTS: 17
Course Tile: ***Fundamentals of Computer Systems***	Course Code: **SITE-1071**
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 1
Office:	Semester: 2
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
The course is the first introduction to computer systems as such; it introduces students to the world of information technology and computing. Students will be able to understand history of computing and their evolution through time. They will see the role of hardware and software in a computer system and how this components work together in order to provide a service for the user. The course describes how the computers represents, store and processes data. It explains the problem solving process as well as how to implement those solutions using a computer. It also describes basic concepts of operating systems, information systems and networking as well as social and ethical issues related with computer systems and limitations of such systems.
Course Objectives
• Understand history of computing • Explain the fundamental concepts of computer hardware and software. • Understand basics of data and data representation in a computer system. • Explain basic architecture of a computer systems • Analyze a problem, decide whether it can or should be solved by a computer, and

provide an appropriate solution
- Describe and use the major components of applications software in the areas of word processing, spreadsheets, database management, presentation graphics, data communications, and Internet.
- Understand the fundamental concepts related with operating systems and be able to work with different operating systems.
- Explain how computer systems are interconnected to share and transfer information.
- Understand the social implications of computer systems.
- Understand limitations of computers.

Learning Outcomes

1. Knowledge of computer representations of data, and arithmetic and conversions in binary, octal, and hexadecimal.
2. Knowledge of combinational logic design principles.
3. Knowledge of sequential logic design principles.
4. Ability to make breadboard circuits.
5. Ability to work effectively as part of a team.
6. Knowledge of hardware description language (HDL) and use for combinational and sequential design.

Prerequisites: None

Student Workload

Lecture: 16	**Tutorial:**	**Home Study**	**Laboratory**

Course Content

Part 1: Introduction

- Introduction
- History of computing and Generations of computers.
- Types of computers.
- Classification of computer systems:
  - Hardware: input, output, processing, memory and storage
  - Software: system and application software

Part 2: Basics of data, data representation and processing.

- Data, information and data processing
  - Basic operations of data processing, input, processing, output, storing, disseminating.
- Number systems and binary numbers
- Data representation
  - Digital and Analog data
  - Representing numeric data: signed magnitude representation, fixed size numbers, two complement, representing real numbers.
  - Representing text data: character sets, ASCII, Unicode.
  - Representing audio data: audio formats.

- Representing image and graphics: representing color, digitized images and graphics.
- Representing video: video codecs.

Part 3: The hardware

- Gates and circuits
  - Computers and electricity
  - Gates
    - NOT, AND, OR, XOR, NOR, NAND gates
    - Constructing gates
  - Circuits
    - Combinational circuits
    - Integrated circuits
    - CISC and RISC computers
- Computer architecture and computing components
  - Stored program concept
  - Von Neumann architecture
    - CPU: ALU, Control unit and registers
    - Memory: RAM / ROM
    - I/O devices and operations
  - Fetch decode execute cycle

Part 4: Problem solving and algorithm design

- Problem solving and algorithms
- Programs and Programming
- Programming Languages
  - Machine level programming
  - Assembly programming
  - High level programming
- Programming paradigms
- Software engineering process
- Abstract data types and algorithms

Part 5: Operating system

- Introduction to operating systems
- Roles of operating systems
- Memory management
- Process management
- CPU scheduling
- File systems and directories
- Security and protection

Part 6: Applications and information systems

- Spreadsheets
- Database management systems
- Artificial Intelligence

        - Knowledge representation, expert systems, neural networks, natural language processing, robotics
    - Graphics and computer aided design
    - Embedded systems
    - Parallel and Concurrent systems
    - Distributed systems
    - Cloud computing
    - Internet of Things

Part 7: Communication basics

- Networking
    - Types of networks, network topologies, advantages of networking, network devices and transmission media.
- Open systems and protocols
- Network addresses
- Current networking technologies
- The internet
    - History of the internet
    - World wide web
    - Web pages and web development
    - HTML, CSS and JavaScript
- Network security

Part 8: Computers in society

- Governmental issues
- Intellectual property issues
- Ethical issues
- Computers in business
- Computers in education
- Computers in entertainment
- Limitations of computing

Schedule

Week	**Topics and Sub Topics**
Week 1 - Week 4	
Week 5 - Week 8	
Week 9 - Week 12	
Week 13 - Week 16	

Assessment Method:

Assessment type

Labs (20 %)

Assignment and Projects (30 %)

Mid exam (20 %)

Final exam (30 %)

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Computer science illuminated, Neil Dale & John Lewis, Jones and Bartlett Publishers, Inc. 2015.

2. Bryant, Randal E., and David R. Hallaron. Computer systems : a programmer's perspective. Boston: Pearson, 2016

3. *Fleuncy 5 with information technology: skills, concepts and capabilities, Lawrence Cnyder,* Pearson Education, 2013.

4. *Computer science made simple, V. Anton Spraul*, Broadway (April 12, 2005)

5. Filho, Wladston. Computer science distilled : learn the art of solving computational problems. Las Vegas, Nevada: Code Energy LLC, 2018. Print.

Fundamental of Computer Science

Program: Software Engineering and Computing Technology	
Module Title: **Computer Science**	Module Code: **SITE-M1081**
	Module ECTS: 28
Course Title: **Fundamentals of Computer Science**	Course Code: **SITE-1081**
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

This course is an introduction to the fundamentals of computer programming, which is the foundation of Computer Science. Students design, write, and debug computer programs. No prior knowledge of programming is assumed.

Course Objectives
By the end of the semester, you will be:

- Familiar with programming in an IDE
- Familiar with program readability/understanding including program style/formatting and self documenting code
- Familiar with debugging process
- Able to design and implement basic programming solutions including statements, control structures, and methods
- Learning about the process of moving from a problem statement to a computational formulation of a method for solving the problem
- Learning a basic set of "recipes"—algorithms
- Learning how to use simulations to shed light on problems that don't easily succumb to closed form solutions

- Learning about how to use computational tools to help model and understand data
- Develop computational problem-solving skills using Python.
- Design and implement event-based graphical animations.

Learning Outcomes

Upon successful completion of this course, the student will be able to:

1. Describe the major components of computer architecture
2. Analyze and demonstrate conversion of data encoding techniques
3. Explain the concept of data storage and named memory locations.
4. Apply decision and repetition structures in program design.
5. Implement methods and functions to improve readability of programs
6. Demonstrate the use of Python lists and dictionaries
7. Describe and apply object-oriented programming methodology.
8. Apply top-down concepts in algorithm design.
9. Write Python programs to illustrate concise and efficient algorithms.

Contents

Chapter 1: Introduction

- What is computation?
- History of computing and programming
- Formal and Natural languages

Chapter 2: Variables, Expressions and Statements

- Primitive data types and values
- Variables
- Data Types & Casting
- Booleans and Expressions

Chapter 3: Functions

- What are functions?
- Built-in Functions
- Function calls
- Parameters and Arguments
- Flow of execution
- Function composition

Chapter 4: Conditionals

- Boolean Expression and Logical Operators
- Conditional Execution
- Chained and Nested Conditions
- Keyboard input and Defensive Programming

Chapter 5: Functions (Not a typo … functions again)

- User-Defined Functions

- Fruitful and void functions
- Return vs Print
- Function Scope
- Decomposition and Abstractions

Chapter 5: Iteration

- Looping Basics
- The while loop
- Break and Continue
- For Loops, Nested Loops
- Nested Lists & Looping

Chapter 6: Strings

- String as a sequence
- Indexing
- Mutability and Immutability
- Searching
- String Comparison

Chapter 7: Data Structures

- What are Data Structures
- Lists
- Numerical Indexing
- Dictionaries
- Tuples and Sets

Chapter 8: Persistence

- File
- Reading and Writing
- Common file formats
- Pickling

Chapter 9: Object Oriented Programming

- Class and Object
- Object Modeling

Prerequisites:

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study: 61 hrs.**	**Laboratory:**

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

- **Assessment**
    - **Lab:** 20%

- **Projects:** 15%
- **Assessments:** 20%
- **Assignments and Quiz:** 15%
- **Final Exam:** 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Zelle, John M. Python programming : an introduction to computer science. Portland, Oregon: Franklin, Beedle & Associates Inc, 2017. Print.
2. Matthes, Eric. Python crash course : a hands-on, project-based introduction to programming. San Francisco: No Starch Press, 2016. Print.
3. How to Think Like a Computer Scientist
4. https://books.trinket.io/pfe/index.html

Basic Writing Skills

Program: Software Engineering and Computing Technology	
Module Title: **Basic English Skills**	Module Code: **EnLa-M11013**
	Module ECTS: 10
Course Title: **Basic Writing Skills**	Course Code: **EnLa-1012**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2018 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description
The course prepares the inexperienced writer for college level reading and academic writing by focusing on critical thinking, reading, writing, and study skills necessary to succeed at the academic level.
Course Objectives
After completing this course students will: • understand and use different skills and strategies to write • understand the meaning of what is written • read different materials for a variety of purposes • write clearly and effectively • understand and use the steps of the writing process • analyze and evaluate the effectiveness of written work • analyze and evaluate the effectiveness of formal and informal communication • communicate ideas clearly and effectively
Learning Outcomes The program provides a knowledge and understanding of the following:
After completing this course students will: • understand and use the steps of the writing process • analyze and evaluate the effectiveness of written work • analyze and evaluate the effectiveness of formal and informal communication • communicate ideas clearly and effectively

Prerequisites:			
Student Workload			
Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study: 61 hrs.**	**Laboratory:**

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

- **Continuous Assessment with Assignment, Tests and Projects (50%),**
- **Final examination (50%)**

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

References:

- Writing with Contemporary Readings by Baker, Barbara and Catherine Baker
- Text & Thought: An Integrated Approach to College Reading and Writing by Lester, Lanny, and Judith Resnick
- The Writers Response: A Reading-Based Approach to College Writing by McDonald, Stephen, and William Salomone

Program: Software Engineering and Computing Technology	
Module Title: *Applied Mathematics*	Module Code: *Math-M1043*
	Module ECTS: 24
Course Title: **Linear Algebra**	Course Code: **Math-1042**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

This course covers matrix theory and linear algebra, emphasizing topics useful in other disciplines, especially in Computer Science. Linear algebra is a branch of mathematics that studies systems of linear equations, the properties of matrices and application of matrices. The concepts of linear algebra are extremely useful in computer science, physics, economics and social sciences, natural sciences, and engineering. Due to its broad range of applications, linear algebra is one of the most widely taught subjects in college-level mathematics. Specifically, in computer science it is applied in Numerical Optimization, Computer Vision, Natural Language Processing and many other Machine Learning and AI related fields.

Course Objectives

The goal of the course is to create an understanding of various theories of vector and matrix manipulation and apply those theories using computer programming to solve several problems in computing fields.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Solving System of Linear Equations
- Manipulation of Vectors and Matrices in Vector Spaces
- Decomposition of matrices using different techniques for different purposes
- Transformation of Vectors and Matrices to different basis Vector
- Application of Linear Algebra to different numerical optimization methods
- Implementation of Linear Algebra topics using a computer programming language

- Applying computer program to solve problems
- Students will have knowledge of Linear algebra that can be applied to wide variety of engineering fields
- Student will acquire practical knowledge of application number crunching for applications like Machine Learning, Computer Graphics

Prerequisites:

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study: 39 hrs.**	**Laboratory: 32 hrs**

Course Content

Chapter 1 – System of Linear Equations

- Vectors and Matrices
- Gaussian Elimination
- Matrix Inverse
- Diagonal, Triangular and Symmetric Matrices

Chapter 2 – Vector Spaces and Subspaces

- Spaces of Vector
- Linear Independence
- Coordinate Basis
- Change of Basis
- Rank and Nullity

Chapter 3 – Determinant

- Determinant by Cofactor
- Row Reduction
- Crammers Rules

Chapter 4 –Transformation

- Linear Transformation
- Scaling
- Shear
- Rotation
- Affine Transformation

Chapter 5 – Eigenvalues and Eigenvectors

- Eigenvalues and Eigenvectors
- Diagonalization
- Complex Vector Spaces

Chapter 6 – Inner Product Spaces

- Inner Products
- Angle and Orthogonality
- Gram-Schmidt Process & QR-Decomposition

Chapter 7 – Diagonalization and Quadratic Forms

- Orthogonal Matrices
- Orthogonal Diagonalization
- Quadratic Forms and their applications

Chapter 8 – Numerical Methods

- LU Decomposition
- The Power Methods
- Singular Value Decomposition

Chapter 9 – Application of Linear Algebra

- Curves, Splines and Surfaces
- Linear Regression

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Chapter 1	Vector and Matrices	Practical Assignment I – Gaussian Elimination
Week 2	Chapter 2	Coordinate Systems	
Week 3	Chapter 3	Determinants	Homework I – Coordinates and Determinants
Week 4	Chapter 4	Linear Transformation – Simple image manipulation	Practical Assignment II – Image Manipulation
Week 5	Exam		
Week 6	Chapter 5	Eigenvalue and Eigenvectors	Homework II – Eigenvalues and Eigenvectors
Week 7	Chapter 6	Dot Product, Projections and Gram-Schmidt Process	Practical Assignment III – Gram-Schmidt Process
Week 8	Exam		
Week 9	Chapter 7	Eigen basis, Diagonalization	Homework III – Eigen basis and Diagonalization
Week 10	Chapter 8	Singular Value Decomposition	Practical Assignment IV - Compression and PCA
Week 11	Chapter 9	Linear Regression and	Homework IV –

		Curve Fitting using Normal Equation	Decomposition and Numerical Methods
Week 12	Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 3 Homework – 15%
- 3 Exams – 65%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Introduction to Linear Algebra, 5th Edition *by Gilbert Strang*
- Elementary Linear Algebra, 11th Edition *by Howard Anton and Chris Rorris*

Applied Math II

Program: Software Engineering and Computing Technology	
Module Title: *Applied Mathematics*	Module Code: Math-M1043
	Module ECTS: 24
Course Title: *Applied Mathematics II*	Course Code: Math-1043
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 20_ _ Semester: I / II / III Meeting day: Meeting time: Meeting location:

Course Description

The student is assumed to be capable in the standard Calculus I topics of taking limits, continuity, taking derivatives of fairly complicated functions, using derivatives, calculating the definite integral for basic functions, integration by substitution and the standard applications of the definite integral. Students who are not fully prepared for this course should review the chain rule, the basic integral formulas and integration by substitution, trigonometric equations and polar coordinates.

Course Objectives

This is the second course in the calculus series for science and engineering students. This course helps you know integration techniques as well as applications of integrals, parametric curves in a plane, as well as infinite sequences and series.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

Set up and evaluate integrals to calculate

- Volume of a solid of known cross sections
- Length of a plane curve
- Area of a surface of revolution
- Work done by a variable force
- Moments and center of mass of a plane lamina, centroid of a plane region.

Techniques of Integration

- Carry out integration by parts and apply it to evaluate integrals.
- Use partial fractions to evaluate integrals of rational functions.

- Given an improper integral, determine whether it converges or diverges, and evaluate it if it converges.

Infinite Series

- Apply limit rules, including L'Hôpital's Rule, to calculate limits of sequences. Apply the concept of boundedness to identify convergent monotonic sequences.
- Use the concept of partial sum to distinguish between convergent and divergent series and to define the sum of a convergent series.
- Recognize geometric series and collapsing series and calculate their sums when convergent.
- Determine radius of convergence and convergence set of a power series.
- Expand a function in a Taylor series. Recall and use the Taylor series of elementary functions.
- Use the remainder in Taylor's formula to estimate the approximation error in a Taylor polynomial.

Plane Parametric Curves, Polar Coordinates

- Derive parametric representations for plane curves described ``mechanically'' (for example, the cycliod.)
- Find tangents and compute length for parametric curves.
- Use polar coordinates, and convert between polar and rectangular coordinates.
- Identify the polar equations for lines, circles and conics.
- Compute the area of regions whose boundaries are defined by equations in polar coordinates.

Apply techniques Integrals, Series and Polar Coordinates in Machine Learning, Computer Vision, Modeling motion of machines, Computer Graphics, Physics Engine and many other.

Prerequisites: Applied Mathematics I

Student Workload

Lecture:48 hrs.	**Tutorial: 32 hrs**	**Home Study: 61 hrs.**	**Laboratory: 48hrs**

Course Content

Chapter 1 - Techniques of Integration

- Integration by Parts
- Trigonometric Integrals
- Trigonometric Substitutions
- Partial Fraction Decomposition
- Strategy for Integration
- Approximate Integration
- Improper Integrals

Chapter 2 - Infinite Series

- Sequences
- Series

- The Integral Test
- The Comparison tests
- Alternating Series
- Absolute Convergence and the Ratio and Root
- Strategy for Testing Series
- Power Series
- Representations of Functions as Power Series
- Taylor and Maclauren Series
- Applications of Taylor Polynomials

Chapter 3 – Differential Equations

- Modeling with Differential Equations
- Direction Fields and Euler's Method
- Separable Equations
- How Fast Does a Tank Drain?
- Which Is Faster, Going Up or Coming Down?
- Models for Population Growth
- Linear Equations
- Predator-Prey Systems

Chapter 4 - Parametric Equations, Polar Coordinates and Conic Sections

- Parametric Equations
- Calculus with Parametric Curves:
- Polar Coordinates
- Areas and Lengths in Polar Coordinates
- Conic Sections

Chapter 5 - Partial Derivatives

- Limits
- Partial Derivatives
- Interpretations of Partial Derivatives
- Higher Order Partial Derivatives
- Differentials
- Chain Rule
- Directional Derivatives

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 - Week 2	Chapter 1		
Week 3 - Week 5	Chapter 2		

Week 6 - Week 8	Chapter 3		
Week 9			Exam
Week 10 - Week 11	Chapter 4		
Week 12 - Week 13	Chapter 5		
Week 14			Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Exam – 70%
- Assignments – 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Thomas' Calculus, 14th Edition *by George B. Thomas, JR.*
- Calculus: Early transcendentals, 8th Edition *by James Stewart*
- Calculus, 10th Edition *by Howard Anton and Irl Bivens Davidson College and Stephen Davis Davidson College*

SEM III

Fundamentals of Database

School: School of Information Technology and Engineering	
Module Title: **Full Stack Development Fundamentals**	Module Code: **SITE-M2101**
	Module ECTS: 29
Course Title: **Fundamentals of Database System**	Course Code: **SITE-2101**
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: I
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
This course covers fundamentals of Database system includes extensive coverage of the Database System Concepts and Architecture Data Modeling and Database Design, Relational Data Model and SQL, Database Design Theory and Normalization, Query Processing and Optimization, Transaction Processing, Concurrency Control, and Recovery. Many additional key database topics from the design and application-building perspective are also covered, including indexes, views, and integrity constraints.
Course Objectives
Students successfully completing this course should be able to: • To understand the different issues involved in the design and implementation of a database system. • To study the physical and logical database designs, database modeling relational, hierarchical, and network models. • To understand and use data manipulation language to query, update and manage a database. • To design and build a single database system and demonstrate competence with fundamental tasks involved with modeling, designing and implementing a DBMS.

- To understand database query processing, optimization transaction processing, concurrency control, and recovery

Learning Outcomes

The Course provides a knowledge and understanding of the following:

- Differentiate database systems from file systems by enumerating the features provided by database systems and describe each in both function and benefit.
- Define the terminology, features, classifications, and characteristics embodied in database systems.
- Analyze an information storage problem and derive an information model expressed in the form of an entity relation diagram and other optional analysis forms, such as a data dictionary.
- Demonstrate an understanding of the relational data model.
- Transform an information model into a relational database schema and to use a data definition language and/or utilities to implement the schema using a DBMS.
- Formulate, using relational algebra, solutions to a broad range of query problems.
- Formulate, using SQL, solutions to a broad range of query and data update problems.
- Demonstrate an understanding of normalization theory and apply such knowledge to the normalization of a database.
- Use an SQL interface of a multi-user relational DBMS package to create, secure, populate, maintain, and query a database.
- Demonstrate an understanding of query processing, optimization transaction processing, concurrency control, and recovery

Prerequisites:

No-prerequisite

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

CHAPTER 1: INTRODUCTION TO DATABASES

Part One: Databases and Database Users

Basic Definitions

- Types of Databases and Database Applications
- Typical DBMS Functionality
- Example of a Database (UNIVERSITY)
- Database Users
- When Not to Use Databases

Part Two: Filing System and Database Design Methodology

- Manual File Handling Systems and Limitation
  File Based Approach and Limitation
- Database Approach
- Main Characteristics of the Database Approach
- Advantages of Using the Database Approach

- Levels of Abstraction
- Database Design Methodology

Part-Three: Database System Concepts and Architecture

- Data Models and Their Categories
- Schemas, Instances, and States
- Three-Schema Architecture /Levels of Abstraction
- Data Independence
- Classification of DBMSs

CHAPTER 2: CONCEPTUAL DATA MODELING AND DATABASE DESIGN
Part One: Data Modeling Using the Entity-Relationship (ER) Model

- Overview of Database Design Process
- Example Database Application (COMPANY)
- ER Model Concepts
- ER Diagrams - Notation
- ER Diagram for COMPANY Schema
- Alternative Notations – UML class diagrams, others

Part Two: Enhanced Entity-Relationship (EER) Modeling

- EER stands for Enhanced ER or Extended ER
- EER Model Concepts: includes all modeling concepts of basic ER
- subclasses/super classes,
- specialization/generalization,
- categories (UNION types),
- attribute and relationship inheritance
- object-oriented concepts, such as inheritance

CHAPTER 3: THE RELATIONAL DATA MODEL AND SQL
Part One: The Relational Data Model and Relational Database Constraints

- Relational Model Concepts
- Relational Model Constraints and Relational Database Schemas
- Update Operations and Dealing with Constraint Violations
- Part Two: SQL (Structured Query Language) - Schema Definition, Constraints, And Queries And Views

  Domain Types in SQL
  DDL -Data Definition Language
  Retrieval Queries in SQL
  Set operations in SQL
  Aggregate functions in SQL
  Arithmetic operations in SQL

Specifying Updates in SQL

Part Three: The Relational Algebra And Calculus

- Unary Relational Operations
- Relational Algebra Operations From Set Theory
- Binary Relational Operations
- Examples of Queries in Relational Algebra
- The Tuple Relational Calculus
- The Domain Relational Calculus

Part Four: Relational Database Design by ER- and EERR-to-Relational Mapping

- ER-to-Relational Mapping Algorithm
- Mapping EER Model Constructs to Relations

CHAPTER 4: DATABASE DESIGN THEORY AND NORMALIZATION
Part One: Functional Dependencies And Normalization For Relational Databases

- Functional Dependencies (FDs)
- Normal Forms Based on Primary Keys
- General Normal Form Definitions (For Multiple Keys)
- BCNF (Boyce-Codd Normal Form)
- Part Two: Relational Database Design Algorithms and Further Dependencies
- Further Topics in Functional Dependencies: Inference Rules, Equivalence, and Minimal Cover
- Properties of Relational Decompositions
- Algorithms for Relational Database Schema Design
- About Nulls, Dangling Tuples, and Alternative Relational Designs
- Further Discussion of Multivalued Dependencies and 4NF

CHAPTER 5: QUERY PROCESSING AND OPTIMIZATION

Part One: Strategies for Query Processing

- Translating SQL Queries into Relational Algebra and Other Operators
- Algorithms for External Sorting
- Algorithms for SELECT Operation
- Implementing the JOIN Operation
- Algorithms for PROJECT and Set Operations
- Implementing Aggregate Operations and Different Types of JOINs Combining Operations Using Pipelining
- Parallel Algorithms for Query Processing

Part Two: Query Optimization

- Query Trees and Heuristics for Query Optimization
- Choice of Query Execution Plans
- Use of Selectivities in Cost-Based Optimization
- Cost Functions for SELECT Operation
- Cost Functions for the JOIN Operation
- Example to Illustrate Cost-Based Query Optimization
- Additional Issues Related to Query Optimization
- An Example of Query Optimization in Data Warehouses
- Overview of Query Optimization in Oracle
- Semantic Query Optimization

CHAPTER 6: TRANSACTION PROCESSING, CONCURRENCY CONTROL, AND RECOVERY

Part One: Introduction to Transaction Processing Concepts and Theory

- Transaction and System Concepts
- Desirable Properties of Transactions
- Characterizing Schedules Based on Recoverability
- Characterizing Schedules Based on Serializability
- Transaction Support in SQL

Part Two: Concurrency Control Techniques

- Two-Phase Locking Techniques for Concurrency Control
- Concurrency Control Based on Timestamp Ordering
- Multiversion Concurrency Control Techniques
- Validation (Optimistic) Techniques and Snapshot Isolation Concurrency Control
- Granularity of Data Items and Multiple Granularity Locking
- Using Locks for Concurrency Control in Indexes

Part Three: Database Recovery Techniques

- Recovery Concepts 814
- NO-UNDO/REDO Recovery Based on Deferred Update
- Recovery Techniques Based on Immediate Update Shadow Paging
- The ARIES Recovery Algorithm
- Recovery in Multidatabase Systems

- Database Backup and Recovery from Catastrophic Failures

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 – week 2	Chapter One	Introduction to database working environment and tools	Assignment #1. Lab#1, Lab#2 Project Title Approval
Week 3 – week 4	Chapter Two	Working with Data Modeling and Database Design tools	Lab#3, Lab#4 Quiz #1 Project Proposal
Week 5 – week 8	Chapter Three	Relational Data Model and SQL	Lab#4, Lab#5 Mid_exam Project Progress Report
Week 9 – week 10	Chapter Four	Database Normalization	Lab#6, Lab#7
Week 11 – week 12	Chapter Five	Query Processing and Optimization	Lab#8, Lab#9 Project Progress Report Quiz #2
Week 13 – week 14	Chapter Six	Transaction Processing, Concurrency Control, and Recovery	Lab#10, Lab#11 Final Project Report Lab exam
Week 15 – week 16	Exam		Final Exam

Teaching Methods

- The format of the course will be lecture, class discussion and presentation, lab exercise, assignments and Project.
- Students will work on MySQL/Oracle/SQL Server and modeling tools

Assessment Method

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that

is relevant to the course. It is group-based

References

1. Elmasri, Ramez, and Sham Navathe. Fundamentals of database systems. Hoboken, NJ: Pearson, 2016.(Text Book)
2. Hernandez, Michael J. Database design for mere mortals : a hands-on guide to relational database design. Upper Saddle River, NJ: Addison-Wesley, 2013
3. Ullman, Jeffrey D., and Jennifer Widom. A first course in database systems. Upper Saddle River, NJ: Pearson/Prentice Hall, 2008

Program: Software Engineering and Computing Technology	
Module Title: Applied Mathematics	Module Code: Math-M1043
	Module ECTS: 24
Course Title: *Probability and Statistics*	Course Code: Math-2044
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description
Probability theory is the branch of mathematics that deals with modelling uncertainty. It is important because of its direct application in areas such as genetics, finance and telecommunications. It also forms the fundamental basis for many other areas in the mathematical sciences including statistics, modern optimization methods and risk modelling. This course provides an introduction to probability theory, random variables and Markov processes. Topics covered are: probability axioms, conditional probability; Bayes' theorem; discrete random variables, moments, bounding probabilities, probability generating functions, standard discrete distributions; continuous random variables, uniform, normal, Cauchy, exponential, gamma and chi-square distributions, transformations, the Poisson process; bivariate distributions, marginal and conditional distributions, independence, covariance and correlation, linear combinations of two random variables, bivariate normal distribution; sequences of independent random variables, the weak law of large numbers, the central limit theorem; definition and properties of a Markov chain and probability transition matrices; methods for solving equilibrium equations, absorbing Markov chains.
Course Objectives
Students will learn fundamental rules of Probability, discrete and continuous distributions most commonly used in Computer Science and Software Engineering. They will be introduced to stochastic processes, Markov chains, statistical inference, and Monte Carlo methods and will apply the theory and methods to the evaluation of queuing systems and computation of their vital characteristics. The student will be exposed to R statistical programming tool in the laboratory session

Learning Outcomes

The program provides a knowledge and understanding of the following:

- Distinguish between quantitative and categorical data and know which graphical and tabular techniques to apply to each.
- Produce and interpret graphical displays for simple data sets.
- Identify when correlation and regression analyses are appropriate.
- Set up a basic simulation model.
- Calculate probabilities using Venn diagrams, tree diagrams, and the Addition and Multiplication rules.
- Describe the concepts of mutually exclusive events, conditional probability, dependent and independent events.
- Calculate probabilities using the Binomial distribution.
- Discuss the concept of a sampling distribution.
- Describe what is meant by the central limit theorem, and understand its relevance to statistical inference.
- Calculate and interpret confidence intervals for estimating population proportions and means.
- Conduct hypothesis tests for population proportions and means.
- Explain the meaning of P-values in hypothesis testing.
- Identify when and how to use the t-distribution.
- Determine appropriate sample sizes for estimating an unknown population proportion or mean.
- Conduct a Chi-Square test of independence.
- Learn how to use R programming language to perform different statistical operations
- Apply theories in Probability and Statistics in Machine Learning, Data Mining, Artificial Intelligence and other related fields

Prerequisites:

Applied Mathematics II
Introduction to Computer Science

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study:39 hrs.**	**Laboratory: 32 hrs**

Course Content

Chapter 1 – Introduction to Probability

- Events and Their Probability
- Rules of Probability
- Combinatorics
- Conditional Probability and Independence

Chapter 2 – Discrete Random Variable and Their Distribution

- Distribution of random variables
- Distribution of random vector

- Expectation and variance
- Families of discrete distributions

Chapter 3 - Continuous Distributions
- Probability density
- Families of continuous distributions
- Central Limit Theorem

Chapter 4 - Computer Simulations and Monte Carlo Methods
- Simulation of random variables
- Solving problems by Monte Carlo methods

Chapter 5 - Stochastic Processes
- Markov processes and Markov chains
- Counting processes
- Simulation of stochastic processes

Chapter 6 - Queuing Systems
- Main components of a queuing system
- The Little's Law
- Bernoulli single-server queuing process
- M/M/1 system
- Multi Server queuing systems
- Simulation of queuing systems

Chapter 7 - Introduction to Statistics
- Population and sample, parameters and statistics
- Simple descriptive statistics
- Graphical statistics

Chapter 8 - Statistical Inference I
- Parameter estimation
- Confidence intervals
- Unknown standard deviation
- Hypothesis testing
- Inference about variances

Chapter 9 - Statistical Inference II
- Chi-square tests
- Nonparametric statistics
- Bootstrap
- Bayesian inference

Chapter 10 – Regression
- Least squares estimation
- Analysis of variance, prediction, and further inference
- Multivariate regression

- Model building

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Chapter 1	**Lab 1 – Basic Probability Programming**	
Week 2	Chapter 2	**Lab 2 – Random Variables**	Assignment 1
Week 3	Chapter 3	**Lab 3 - Distributions**	Assignment 2
Week 4	Chapter 4	**Lab 4 – Applying Monet Carlo Methods**	
Week 5 – Week 7	Chapter 5	**Lab 5 – Markov Chains and Processes**	Assignment 3
Week 7	Chapter 6		Assignment 4
Week 8	Exam		
Week 9	Chapter 7	**Lab 6 – Implementing Descriptive Statistics – Mean, Median …**	
Week 10 – Week 11	Chapter 8	**Lab 7 – Correlation, Covariance, Parameter Estimation**	Assignment 5
Week 12 – Week 13	Chapter 9	**Lab 8 – Bayesian Inference**	
Week 14	Chapter 10	**Lab 9 – Application of Probability and Statistics - Regressions**	Assignment 6
Week 15	Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Exams – 60%
- Labs – 15%

- Assignments – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Probability and Statistics for Computer Scientists, 2nd Edition *by Michael Baron*
- Larsen, Richard J., and Morris L. Marx. *An introduction to mathematical statistics and its applications*. Boston: Pearson, 2018
- Nik, Hossein. *Introduction to probability, statistics, and random processes*. Blue Bell, PA: Kappa Research, LLC, 2014
- Fischetti, Tony. *Data analysis with R : load, wrangle, and analyze your data using the world's most powerful statistical programming language*. Birmingham, UK: Packt Publishing, 2015

Program: Software Engineering and Computing Technology	
Module Title: *"Computer Science"*	Module Code: SITE-M1081
	Module ECTS: 28
Course Title: *Discrete Mathematics*	Course Code: SITE-2083
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

The purpose of this course is to understand and use (abstract) discrete structures that are backbones of computer science. In particular, this class is meant to introduce logic, proofs, sets, relations, functions, counting, and probability, with an emphasis on applications in computer science.

Course Objectives

- The fundamental concepts of discrete mathematics.
- The role of formal definitions, formal and informal mathematical proofs, and underlying algorithmic thinking, and be able to apply them in problem solving.
- The role of discrete mathematics in applications in computer science.

Learning Outcomes

The programme provides a knowledge and understanding of the following:

Upon completion of this course students should be able to do the following:

- Discuss definitions and diagram strategies for potential proofs in logical sequential order without mathematical symbols (plain English).
- Construct mathematical arguments using logical connectives and quantifiers.
- Verify the correctness of an argument using symbolic logic and truth tables.
- Construct proofs using direct proof, proof by contradiction, and proof by cases, or mathematical induction.
- Solve problems using counting techniques and combinatorics.

- Perform operations on discrete structures such as sets, functions, relations or sequences.
- Solve problems involving recurrence relations and generating functions.
- Construct functions and apply counting techniques on sets in the context of discrete probability.
- Apply algorithms and use definitions to solve problems to proof statements in elementary number theory.

Prerequisites:

Fundamentals of computer science

Student Workload

Lecture: 32 hrs	**Tutorial: 16hr**	**Home Study:**	**Laboratory: 48hr**

Course Content

1. The Foundations: Logic and Proofs

1.1 Propositional Logic
1.2 Applications of Propositional Logic
1.3 Propositional Equivalences
1.4 Predicates and Quantifiers
1.5 Nested Quantifiers
1.6 Rules of Inference
1.7 Introduction to Proofs
1.8 Proof Methods and Strategy

2 Basic Structures: Sets, Functions, Sequences, Sums, and Matrices

2.1 Sets
2.2 Set Operations
2.3 Functions
2.4 Sequences and Summations
2.5 Cardinality of Sets
2.6 Matrices

3 Algorithms

3.1 Algorithms
3.2 The Growth of Functions
3.3 Complexity of Algorithms

4 Number Theory and Cryptography

4.1 Divisibility and Modular Arithmetic
4.2 Integer Representations and Algorithms
4.3 Primes and Greatest Common Divisors
4.4 Cryptography

5 Induction and Recursion

5.1 Mathematical Induction
5.2 Strong Induction and Well-Ordering
5.3 Recursive Definitions and Structural Induction
5.4 Recursive Algorithms

5.5 Program Correctness

6 Counting

6.1 The Basics of Counting
6.2 Permutations and Combinations
6.3 Binomial Coefficients and Identities
6.4 Generalized Permutations and Combinations

7 Discrete Probability

7.1 An Introduction to Discrete Probability
7.2 Probability Theory
7.3 Bayes' Theorem
7.4 Expected Value and Variance

10 Graphs

10.1 Graphs and Graph Models
10.2 Graph Terminology and Special Types of Graphs
10.3 Representing Graphs and Graph Isomorphism
10.4 Connectivity
10.5 Euler and Hamilton Paths
10.6 Shortest-Path Problems
10.7 Planar Graphs
10.8 Graph Coloring

11 Boolean Algebra

11.1 Boolean Functions
11.2 Representing Boolean Functions
11.3 Logic Gates
11.4 Minimization of Circuits

Schedule

Week 1 -4			
Week 5 - 8			
Week 9 - 12			
Week 13 - 16			

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Mid Exam 15%
- Lab Exam I (practical) 10%
- Lab Exam II(practical) 10%
- Assignment I 10%

- Project 30%
- Final Exam 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Epp, Susanna S. Discrete mathematics with applications. Boston, MA: Brooks/Cole Cengage Learning, 2011
2. Rosen, Kenneth H. Discrete mathematics and its applications. New York: McGraw-Hill, 2012
3. Graham, Ronald L., Donald E. Knuth, and Oren Patashnik. Concrete mathematics : a foundation for computer science. Reading, Mass: Addison-Wesley, 1994

Program: Software Engineering and Computing Technology	
Module Title: *"Computer Science"*	Module Code: SITE-M1081
	Module ECTS: 28
Course Title: *Fundamentals of Data Structures*	Course Code: SITE-2082
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

This course covers the design, analysis, and implementation of basic data structures in C++. Algorithms operating on the data structures are also covered. It is fundamentally a course about data structures, not about C++. However, as we will implement some of the data structures in C++, basic aspects of C++ and (very little) object oriented programming in C++ are also covered. We will start with a brief discussion of the C++ programming language. Then, we survey fundamental data structures (array, vector, lists, queue, stack, hash map, trees, and possibly graphs) and how we can make use of them in C++. Then, we delve deeper into the design, analysis and implementation of such data structures. Asymptotic analysis of algorithms and data structures is also discussed.

Course Objectives

- To introduce algorithmic analysis, fundamental data structures, problem solving paradigms
- To study the representation, implementation and application of basic data structures.
- To introduce algorithmic strategies and time complexity analysis of problems.
- To study applications of Data Structure in solving real life problems

Learning Outcomes:

- Able to understand the concepts of data structure, data type and array data structure.
- Able to understand and apply various data structure such as linked list, stacks, queues, trees and graphs to solve various computing problems using C++ programming language
- Able to analyze algorithms and determine their time complexity.

- Able to effectively choose the data structure that efficiently model the information in a problem

Prerequisites: Fundamentals of Computer Science

Student Workload

Lecture:32 hrs	**Tutorial: 16hr**	**Home Study:**	**Laboratory: 48hr**

Course Content

Chapter One:
Introduction to data structures: Concept of data, Data types, Data Object, Data structure, Abstract, Data types (ADT), linear data structures using sequential organization: Concept of sequential organization, Concept of Linear data structures, arrays as ADT, Multidimensional arrays, Storage representations (row major and column major and their address calculation). Polynomial representation
Chapter Two:
Concept of linked organization, singly linked list, doubly linked list, circular linked list and operations on above data structure. Application of linked list for Representation and manipulations of polynomials
Chapter Three:
Concept of stack and queues as ADT, Implementation of stacks using sequential and linked organization, linear queue, circular queue using sequential and linked organization, Priority Queue, Application of stack for expression conversion, evaluation, processing of function calls, and recursion, Application of queue in job scheduling.
Chapter Four:
Non-linear Data Structures: Trees and binary trees-concept and terminology, Sequential & Linked representation of binary trees, Algorithm for tree traversals, Conversion of general tree to binary tree, Binary search trees, Applications of binary tree : expression tree, decision tree. Graph: Representation of graph -Adjacency matrix and Adjacency list, Graph traversals, application of graph: connected components, Spanning tree, Minimum cost spanning tree, and shortest path computation.
Chapter Five:
Searching and Sorting: Sequential, binary and Fibonacci search. General concepts: sort order, sort stability, efficiency and passes, Internal and external sorting, Bubble sort, Quick and Merge sort. Files: Organization of files: sequential and direct access file and simple index file, hashing function and its characteristics, collision resolution, linear probing, chaining with and without replacement, rehashing

Schedule

Weeks	**Topics and Subtopics**		
	Lecure	**Laboratory**	**Assessment**
Week 1 - Week 4	**Chapter 1 - Chapter 3]**		Lab Exam I

Week 5 - Week 8	**Chapter 4 - Chapter 5]**		Mid Exam Lab Exam I
Week 9 - Week 12	**Chapter 6 - Chapter 7]**		Assignment I and Lab Exam II
Week 12 - Week 16			Project and Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Mid Exam 15%
- Lab Exam I (practical) 10%
- Lab Exam II(practical) 10%
- Assignment I 10%
- Project 30%
- Final Exam 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Data Abstraction & Problem Solving with C++, 6th Ed. Carrano & Henry, Pearson, 2013
2. Data Structures and Algorithms in C++, Cengage India; 4TH, INTERNATIONAL ECONOMY EDITION edition (2013)

Program: Software Engineering and Computing Technology	
Module Title: **Full Stack Software Development**	Module Code: **SITE-M2101**
	Module ECTS: 29
Course Title: *Fundamentals of Networking*	Course Code: **SITE-2102**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

Data Communication and Computer Networking is a course which deals about the basic concepts of ways of communication, types of communications, basic types of networking devices, how devices are communicating each other using the TCP/IP and OSI reference model, designing a small LAN network using peer to peer connection, basic types of LAN and WAN protocols, the Wireless Network and identifying the protocols which are working at the Application and Transport layers.

Course Objectives

General Objective

After completing the course the students are going to evaluate the application of Computer Networking in today's global communities who are linked by common goals and technologies.

Specific Objective

1. Analyze the principles of data communication, computer networks and inter-networking.
2. Describe the trends of the rapidly evolving communication and networking technologies.
3. Categorize the functions and design of different types of computer networks from LANs to WANs.

4. Evaluating the impact of data communication and networking technologies on society
5. Comparing different types of designing and cabling a network
6. Comparing of IPV4 and IPV6 IP addressing
7. Evaluating a small network using basic Cisco IOS commands for routers and switches

Learning Outcomes

The programme provides a knowledge and understanding of the following:

On successful completion of this unit, students should be able to:

a. Explain the principles of networking protocols and standards;
b. Identify reference models and layered architectures in networking;
c. Recognize network topologies, media types and network components;
d. Identify the network (OSI) layers at which various network devices operate;
e. Use some basic tools/utilities for network analysis;
f. Employ basic cabling and network design techniques to connect devices.

Prerequisites: Fundamental of Computer Systems

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

Chapter 1: Introduction to Data Communications and computer networks

Basics of data communications
Basic components of communication network
Fundamentals of Information Transmission Media and Coding
Types of networks [PP &CS]
Current and future networks
Basic Data Communication Hardware [NIC, cables, Switch, Router, Hub, Bridge…]
Basic types of network Topologies: ring, bus, star, and mesh

Chapter 2: Data Communication Layers

OSI 7 layer reference model and concepts of open standards
Motivations for standards
ISO reference model
Open standard
Internetworking with TCP/IP; Structure
Architectures
Protocol standards
Internet IP and Bandwidth
The ISO protocol

Chapter 3: Network Layer

Internetworking,
IPV4 Addressing [Class full]
IPV4 Addressing [Classless- VLSM addressing]
Network layer protocols
Unicast and multicast routing
Introducing IPV6 Addressing

Chapter 4: Physical and data link Layers
Physical Layer
Coding and Multiplexing
Data link layer overview
Flow and error control
Medium access control
Local area networks [Types, Protocols, Performance]
Wireless Networks
Switching ,Bridging and VLANs
Wide Area Networks [Packet switched data network, Circuit switched data networks, ISDN, Private networks]
Internet data link control protocols: PPP and HDLC

Chapter 5: Transport Layer
Overview & process to process delivery
TCP and UDP
Congestion control
Multimedia Communication and Quality of service (QOS)

Chapter 6: Application Layer
Application layer overview
DNS, SMTP, FTP, Telnet
WWW & HTTP
Socket programming using TCP and UDP
Fundamentals of Queuing Theory

Schedule

Weeks	**Topics and Subtopics**		
	Lecure	**Laboratory**	**Assessment**
Week 1- Week 4	Chapter 1- Chapter 2		
Week 5- Week 8	Chapter 3		
Week 9- Week 13	Chapter 4 - Chapter 5		
Week 14- Week 16	Chapter 6		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

Type	Mark
Quiz Exam	5
Assignments	10
Lab Exam	20
Final Project	20
Mid Exam	20
Final Exam	25
TOTAL	**100%**

Course Policies

- Attendance Policy: Regular and punctual attendance is required. A maximum of 2 unexcused absences is allowed for a four-credit course. If a student absent more than 2 days, he/she will not sit on the mid / final exam.
- Assignments: No Late Assignment will be accepted
- Test/Quizzes: Rarely re-examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- Cheating/Plagiarism: No second Chance or excuses.

References

1. Kurose, James F., and Keith W. Ross. *Computer networking : a top-down approach*. Boston: Pearson, 2017
2. A. S. Tanenbaum, Computer Networks, 5th ed, Prentice Hall, 2010
3. W. Stallings, Data and Computer Communications, 10th ed. Prentice Hall, USA, 2014

SEM IV

System Programming

Program: Software Engineering and Computing Technology	
Module Title: **Computer Systems and Programming**	Module Code: **SITE-M1071**
	Module ECTS: 17
Course Tile: *Systems Programming*	Course Code: SITE-2072
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2 Semester: 2 Meeting day: Meeting time: Meeting location:

Course Description

A computer needs an operating system to manage its resources and provide support for common functions such as accessing peripherals like keyboard, screen, mouse, printers, hard disk, or network. The operating system also provides useful abstractions such as processes (for multiprogramming), threads (for multitasking), semaphores (for coordination), scheduling policy, and memory management. System programming refers to writing code that takes advantage of operating system support for programmers.

This course is designed to introduce students to system programming. By the end of this course, students should be proficient at writing programs that take full advantage of operating system support. Students will be able to build, debug and test system programs. The course stresses learning how to use tools such as debuggers. The course also stresses the understanding of how programs execute on today's computers.

Programming will be in C on Unix/Linux based systems to introduce students to a new programming eco system, as well as enable the mapping of high-level language constructs to the underlying machine.

Course Objectives

Demonstrate mastery of the internal operation of UNIX system software.

By the end of the course students should be able to accomplish the following:

- Understand the basic principles in system programming and identify different tools that are used to develop system programs.
- Become acquainted with the basic tools used to develop software on the UNIX platform.
- Write, compile, debug, and execute C programs that correctly use basic C language constructs, arrays, pointers, functions, structures and system interfaces.
- Understand program translation and execution steps involved in building C programs.
- Understand machine level representation of programs.
- Understand different machine level operations and how high level programs are translated into their machine level counterparts.
- List UNIX system calls, and invoke them correctly from within C programs.
- Write, compile, debug, and execute C programs that create, manage and terminate processes.
- Define UNIX signals and signal handlers, and describe their use.
- Write, compile, debug, and execute C programs that use files and I/O on UNIX.
- Describe the protocols (such as TCP and IP) and interfaces (such as sockets) used for communication among different computers.

Learning Outcomes

The course provides an introduction to programming using the C language, and an introduction to Unix systems programming.
Upon completion of this course you will be able to:

1. Write and debug programs in the C programming language;
2. Explain how the Unix command shell processes commands;
3. Write simple shell scripts;
4. Explain how the Unix file system stores information;
5. Explain how concurrent processes are used in Unix;
6. Explain how asynchronous I/O and signals operate;
7. Apply the above knowledge to solve programming problems.

Prerequisites: Fundamentals of Computer science

Student Workload:

Lecture: 32	**Tutorial: 32**	**Home Study:**	**Laboratory: 48**

Course Content

Part 1: System programming overview

- General overview of systems and a computer system, application and system software and programming.
- Introduction to system programming and differences and similarities between system and application programming.
- The role of operating system in system programming.
- Introduction to UNIX / LINUX operating system and C programming language.

Part 2: C Programming Basics

- Information representation in a computer system.
- General structure of C programs, steps of C program development.
- Basic language constructs: data types, operator, control structures and libraries.
- Arrays and Pointers
- Functions
- Types, Structures and Unions
- Dynamic memory allocation
- Programming with GNU C and C tools: debugger, make facility, de-compilers and assemblers

Part 3: Program translation and Execution

- Program translation steps: preprocessing, compilation, assembling, linking and loading.
- Compilation: Phases of compiler and code generation.
- Linking: Object files and structure, symbol resolution and relocation, static and dynamic linking, static and dynamic libraries, executable file formats.
- Loading: steps of loading, dynamic loading, memory layout of a process.

Part 4: Machine Level Representation of Programs

- Overview of high level, assembly and machine languages, basic processors and their types, computer architecture and the fetch-decode-execute cycle.
- Assembly basics and introduction to IA32.
- Data formats
- Arithmetic and logical operations
- Control
- Procedures
- Array allocation and access

Part 5: System level programming

- Exceptional Control flow: Exceptions, classes of exceptions, exception handling, exceptions in Linux / IA32 systems.

- Processes: Logical control flow and concurrent flows, private address space, user and kernel modes, context switching, process control in Linux OS.
- Signals: sending and receiving signals, signal-handling issues, signal handling in Linux OS.
- System level I/O: Unix I/O, opening and closing files, reading and writing files, file metadata, sharing files, I/O redirection, standard I/O library.
- Network programming: The client server programming model, Networks, The sockets interface, web servers.

Content of Laboratory Manuals

Session 1: Introduction Linux OS and the terminal

- Introduction to Linux OS, Linux file structure, file paths, basic commands to work on directories and files, searching in files, I/O redirection and pipes, file system security and permissions, using Linux help facility.

Session 2: C Basics

- Implementing c programs, compiling and executing c programs, performing program translation steps separately (preprocessing, compiling, assembling, linking and loading), implementing simple C programs.

Session 3: Control structures in C

- Implementing C programs using different control structures.

Session 4: Arrays and strings

- Implementing C programs using arrays.

Session 5: Pointers

- Implementing C programs using pointers. Implementing different string processing functions.

Session 6: Type definitions, structures and enumerations

- Implementing C programs that use typedefs, structs and enums

Session 7: Dynamic memory allocation

- Implementing C programs that use dynamic memory.

Session 8: GNU C tools

- Debugging C programs using gdb, make facility, disassembling code, and investigating object codes.

Session 9: Machine level programming.

- Combining C and assembly language. Implementing programs using IA32 assembly language.

Session 10: Process management in Linux.

– Writing C programs that take advantage of system calls define in Linux OS for process management.

Session 11: System level I/O.

– Writing C programs that take advantage of system calls define in Linux OS for system level I/O and files.

Session 12: Networking

– Writing C programs that take advantage of system calls define in Linux OS for networking.

Schedule – optional

Week	Topics and Sub Topics		
Week 1 - Week 4			
Week 5 - Week 6			
Week 7 - Week 12			
Week 13 - Week 16			

Assessment Method:

Assessment Type

Labs (10 %)

Projects (40 %) 4 projects at the end of part 2,3,4 and 5.

Mid exam (20 %)

Final exam (30 %)

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

TEXTBOOKS:

1. *Kernighan, Brian W., and Dennis M. Ritchie. The C programming language. Englewood*

Cliffs, N.J: Prentice Hall, 1988

2. *C Programming: A Modern Approach (Second Edition),* K. N. King, Norton & Co. 2008. .

REFERENCES:

3. *The Practice of Programming, Brian W. Kernighan and Rob Pike,* Addison-Wesley 1999. *The UNIX Programming Environment, B. Kernighan & R. Pike Prentice-Hall, 1984.*

4. *Beck, Leland L. System software : an introduction to systems programming. Reading, Mass: Addison-Wesley, 1997*

Program: Software Engineering and Computing Technology	
Module Title: *"Computer Architecture and Design"*	Module Code: SITE-M2111
	Module ECTS: 12
Course Title: *Fundamental of Electrical Circuits and Electronics*	Course Code: SITE-2111
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

This course introduces the fundamentals of the lumped circuit abstraction. Topics covered include: resistive elements and networks; independent and dependent sources; switches and MOS transistors; digital abstraction; amplifiers; energy storage elements; dynamics of first- and second-order networks; design in the time and frequency domains; and analog and digital circuits and applications. Design and lab exercises are also significant components of the course

Course Objectives
After studying this course students will be able to

- Understand the basic electrical engineering principles and abstractions on which the design of electronic systems is based. These include lumped circuit models, digital circuits, and operational amplifiers.
- Use these engineering abstractions to analyze and design simple electronic circuits.
- Formulate and solve differential equations describing the time behavior of circuits containing energy storage elements.
- Use intuition to describe the approximate time and frequency behavior of circuits containing energy storage elements.
- Understand the concepts of employing simple models to represent non-linear and active elements-such as the MOSFET-in circuits.

- Build circuits and take measurements of circuit variables using tools such as oscilloscopes, multimeters, and signal generators. Compare the measurements with the behavior predicted by mathematic models and explain the discrepancies.
- Understand the relationship between the mathematical representation of circuit behavior and corresponding real-life effects.
- Appreciate the practical significance of the systems developed in the course.

Learning Outcomes

The programme provides a knowledge and understanding of the following:

- Employ simple lumped circuit models for resistors, sources, inductors, capacitors, and transistors in circuits.
- Analyze circuits made up of linear lumped elements. Specifically, analyze circuits containing resistors and independent sources using techniques such as the node method, superposition and the Thevenin method.
- Employ Boolean algebra to describe the function of logic circuits.
- Design circuits which represent digital logic expressions. Specifically, design a gate-level digital circuit to implement a given Boolean function.
- Check static discipline constraints in circuits. For example, determine if the circuit representing a gate provides adequate noise margins.
- Determine the output produced by a circuit for a given set of inputs using the switch resistor model of a MOSFET.
- Perform a small-signal analysis of an amplifier using small signal models for the circuit elements.
- Calculate the time behavior of first order and second order circuits containing resistors, capacitors and inductors.
- Calculate the frequency response of circuits containing resistors, capacitors and inductors.
- Construct simple gates, amplifiers, or filters in the laboratory.
- Determine in the laboratory the time-domain and frequency-domain behavior of an RLC circuit.
- Use operational amplifier models in circuits which employ negative feedback.
- Use complex impedances to determine the frequency response of circuits.
- Determine the power dissipation in digital gates and employ CMOS technology to reduce static power losses.
- Predict how a given circuit will affect an audio signal in the laboratory given the frequency response of the circuit.
- Design, build and test an audio playback system which includes both analog and digital components.

Prerequisites:

Fundamentals of computer science

Student Workload

Lecture:32 hrs	**Tutorial: 16hr**	**Home Study:**	**Laboratory: 48hr**

Schedule

Week 1 -4			
Week 5 - 8			
Week 9 - 12			
Week 13 - 16			

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Mid Exam 15%
- Lab Exam I (practical) 10%
- Lab Exam II(practical) 10%
- Assignment I 10%
- Project 30%
- Final Exam 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Agarwal, Anant, and Jeffrey H. Lang. Foundations of Analog and Digital Electronic Circuits. San Mateo, CA: Morgan Kaufmann Publishers, Elsevier, July 2005

Program: Software Engineering and Computing Technology	
Module Title: **Full Stack Development Fundamentals**	Module Code: **SITE-M2101**
	Module ECTS: 29
Course Title: **Fundamental of Web Design and Programming**	Course Code:**SITE-2104**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description
Although the World-Wide Web was initially conceived as a vehicle for delivering documents, it is now being used as a platform for sophisticated interactive applications, displacing the traditional mechanism of installable binaries. Web-based applications offer numerous advantages, such as instant access, automatic upgrades, and opportunities for collaboration on a massive scale. However, creating Web applications requires different approaches than traditional applications and involves the integration of numerous technologies. This class will introduce students to the Web technologies and give you experience creating Web applications. In the process students will learn about markup languages, scripting languages, network protocols, interactive graphics, event-driven programming, and databases, and see how they all work together to deliver exciting applications. Consequently, the course will not hard coded content as the web development is very dynamic. The instructor at the time should follow the trends at the time of the course by considering properties of varies frameworks, new developments and other known universities syllabuses in accordance with the Practical skill mentioned. The assessment method, books and other materials must be chosen by the instructors at the time. However, Mozilla and w3school are listed in the references section as these institutions drive the technology.
Course Objectives
The objective of this course to introduce students with the current state of the art web

application technologies and integration of these technologies to create a robust, world class web application.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Build websites, web frameworks using basic HTML, CSS and JavaScript
- Build client-side application that responsive
- Build web application that consist AJAX and Server-Side
- Build Web APIs that are read for many client-side applications
- Build applications with database and ORM/ODM integrations
- Build fully integrated enterprise application

Prerequisites:

Fundamentals of Computer Science
Fundamentals of Database
Fundamentals of Networking

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study:65 hrs.**	**Laboratory: 48hrs**

Course Content

Part I – Basics of Front-end Development

- Chapter 1 –Hypertext Markup Language (HTML)
- Chapter 2 – Cascading Style Sheets (CSS) and CSS Preprocessors
- Chapter 3 – Basic JavaScript Programming
- Chapter 4 – Advanced JavaScript Programming
- Chapter 5 – DOM Events

Part II – Server-Side Programming

- Chapter 6 – Client-Server Architecture and Hypertext Transfer Protocol
- Chapter 7 - Basics of Server-Side Framework
- Chapter 7 - Advanced of Server-Side Framework
- Chapter 8 – Storage Tier

Part III – Useful Front-end Frameworks

- Chapter 9 – Basic of "Front-end Framework"
- Chapter 10 – Advanced I of "Front-end Framework"
- Chapter 11 – Advanced I of "Front-end Framework"
- Other Important Topics in Web

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 - Week 6	Part I		

Week 7 – Week 10	Part II		
Week 11 – Week 14	Part III		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

Assessment Type

Labs (10 %)
Projects (40 %)
Mid exam (20 %)
Final exam (30 %)

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Duckett, Jon. *JavaScript & jQuery ; HTML & CSS*. Indianapolis, Ind: John Wiley & Sons, 2014
- Duckett, Jon. *HTML & CSS : design and build websites*. Indianapolis, IN: John Wiley & Sons Inc, 2014
- Myers, Mark. *A smarter way to learn HTML & CSS : learn it faster, remember it longer*. North Charleston, South Carolina: CreateSpace Independent Publishing Platform, 2015
- Myers, Mark. *A smarter way to learn Javascript*. Charleston, SC: ASmarterWayToLearn.com, 2014.
- Duckett, Jon, Gilles Ruppert, and Jack Moore. *JavaScript & jQuery : interactive front-end web development*. Indianapolis, IN: Wiley, 2014
- Ullman, Larry E. *PHP and MySQL for dynamic web sites*. Berkeley, California: Peachpit Press, 2018.
- Welling, Luke, and Laura Thomson. *PHP and MySQL web development*. Hoboken, NJ: Addison-Wesley, 2017
- STAUFFER, MATT. *LARAVEL : up and running*. S.l: O'REILLY MEDIA, 2018
- https://developer.mozilla.org/en-US/
- https://expressjs.com/en/starter/installing.html
- https://www.w3schools.com/

Object Oriented Programming

Department: Software Engineering	
Module Title: **Programming**	Module Code: **SITE-M2091**
	Module ECTS: 11
Course Title: *Object Oriented Programing*	Course Code:**SITE-2091**
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/I Meeting day: Meeting time: Meeting location:

Course Description
This course is an in-depth look at object oriented paradigm using the popular programming language Java. It is not intended for first time programmers. After some preliminaries devoted to basic syntax and program structure, classes, composition, inheritance and polymorphism are examined. The Java collection classes are studied in some detail, as is the rather complex set of I/O classes. Additional topics include exception handling, building GUIs with JavaFX, and multi-threading. Throughout the second half of the course, a series of homework problems develops a non-toy application, illustrating by example how larger object-oriented programs are organized.
Course Objectives
Understand the basic principles of object oriented approach in order to solve different types of problems.
Learning Outcomes The program provides a knowledge and understanding of the following:
Practical Skills – Implement the oop principle using java – Perform object oriented analysis , design

- build a robust and flexible system

- work on exception handling
- Building a distributed and concurrent system
- Use different strategies in for solving specific type of problem
- Effective communication skill acquired during group work
- A hand on experience on java technologies

Prerequisites: Fundamental of Programing

Student Workload

Lecture: 48	**Tutorial: 32**	**Lab:48**	**Home: 61**

Course Content

Topic 1. Introduction to Object Oriented Programing

- Form problem solving to code
- Different types of programming paradigms
- Need for Object Oriented paradigms
- What really is and Benefit of object oriented programming
- The basic concept of class and object

Topic 2. More in to Object Oriented Programing

- How to design class and create objects
  - Identifying Potential Classes ,Attributes and Methods
  - Object constructors and the rules
- The object family
- Object Oriented Principle
  - Encapsulation
  - Abstraction
  - Inheritance
  - Polymorphism

Topic 3. Exception Object

- Basic concept of Exception
- Constructs and Exception Semantics
- Defining Exception Objects and Handlers
- Raising Exceptions
- Code Finalization and Cleaning Up
- Object Finalization and Block Finalization

Topic 4. Network Object

- The network model
- What is socket object
- Listener socket in depth
- The concept of thread
  - Creating thread using thread class and runnable interface
  - Single thread of execution a
  - Multiple thread of execution
- Thread synchronization and resource sharing

Topic 5. Object Oriented Problem Solving In depth

- What are OO Problem Solving strategies
- How to handle object creation
- Class and object composition
- Dealing with the behavior object in depth
- Separation of objects on the model, view and controller

Schedule

Weeks	**Topics and Subtopics**		
	Lecure	**Laboratory**	**Assessment**
Week 1- Week 4	Topic 1 ,2	**Intro to Java** • Data types and Variable • Input and output • Control structure • Basic class and working with the main Method • Arrays ,Array List ,Hash Map **More on OOP** • Object lifetime and constructers • Using object • Abstract vs concrete class • Inheritance • Overloading vs overriding	Assignment 1 Assignment 2 Mid Exam
Week 5- Week 8	Topic 3	**Advanced Programing** • File input output • Exception handling • Gui with swing/JavaFX	Assignment 3 Project 1
Week 9- Week 13	Topic 4	• Working with thread • Working with socket	Assignment 4 Project 2
Week 14- Week 16	Topic 5	-	Assignment 5

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Mid Exam 30%
- Assignment 10%
- Lab assessment 10%
- Project I and II 30%
- Final Exam 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted

- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. HORSTMANN, CAY S. *CORE JAVA - FUNDAMENTALS*. S.l: PRENTICE HALL, 2018
2. Eckel, Bruce. *Thinking in Java*. Upper Saddle River, NJ: Prentice Hall, 2006

Department: Software Engineering	
Module Title: **Full Stack Development Fundamentals**	Module Code: **SITE-M2101**
	Module ECTS: 29
Course Title: Fundamentals of *Software Engineering*	Course Code: **SITE-2103**
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description
As soon as computers were invented, the ability to tell them what to do made programming an awesome task. Programmers used a technique called "code and fix" write some code, and fix it when it breaks. But things started getting too compacted too soon. Writing bigger software through the so-called "code and fix" mechanism had its problems. The inability to assess progress, quality, or risks, the difficulty to accommodate changes without a major design overhaul, and the unclear delivery features, timing, and support made "code and fix" strategy is difficult to use in larger projects. Therefore, software engineering emerged as a discipline to manage software projects. The activities in software engineering include defining what the software should do (software requirements), describing the solution specified in the requirements on paper (design), writing software that meets the solution using a programming language (implementation), making sure that the implementation meets the design and requirements (testing), and progressively modifying the software (maintenance and evolution). But software engineering is not just a prerequisite to coding. It also defines the way we communicate and keep stream of our tasks, it is the way we collaborate too. The key idea that software engineering introduced in projects is traceability. The things written in the requirements must have a corresponding solution described in the design document, and a piece of code that implements a certain artifact on the design. This course introduces software engineering though a project based delivery strategy and you are expected to apply what you studied in classes on your projects

Course Objectives

Understand the basic principles of Software Engineering that helps to understand a problem, perform requirement analysis and work on the solution. Students must write requirements specification documents, Design a system, component, or process to meet desired needs.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills	**Transferable Skills**
– Perform a requirement analysis on different area of problem – Perform an object-oriented design – Prepare different types of documents. (requirement, design and testing) – Verify and validate a Software system	– Students will be able to work with different platforms and have a handy experience on working on documentation skills – Effective communication skill acquired during group work – A hand on experience on proposal writing and different collaborative tools

Prerequisites:

Object Oriented Programing

Student Workload

Lecture:	**Tutorial:**	**Lab:**	**Home:**

Course Content

Topic one: ***Introduction to Software Engineering***

- What is software engineering
- How it is different from programing
- Aspects of software engineering
- The relationship with other fields
- What is software project and what roles being available
- How to work on a group of teams for software project

Topic two: ***software life cycle***

- Characteristics of a Software Process
- Software Development Process
- Software life cycle and process models
- Process assessment models
- Software process metrics

Topic three: ***Requirement Engineering [three lectures]***

- Defining requirements of a project and documentation
- Why requirement?
- Gathering and classifying requirement
- Use case modeling

- Prototyping requirement

Topic Four: software *design [four lectures]*

- What is software architecture and architectural style
- Represent architecture in UML [Component and Deployment] diagram
- Class diagram
- Sequence diagram

Topic Five: software *implementation and testing [three lectures]*

- Case scenario of loan calculator
- State diagrams
- Software testing

Topic six: *Agile Development*

- Scrum
- Scrum framework

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1- Week 4	Topic 1 ,2	collaboration tools [trelo, slack]	Proposal (Doc)
Week 5- Week 8	Topic 3	Edrawe mx for developing use case Mockup tools [for UI]	Mid Exam SRS(Doc)
Week 9- Week 13	Topic 4	Edrawe mx for developing Component, Deployment, Class, Sequence and sate chart diagram	SDS (Doc)
Week 14- Week 16	Topic 5,6	Working on version control(Git) Unit testing using Junit	TPD (Doc)+ Implementation Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Mid Exam 25%
- Proposal SRS, SDS, TPD, IMP 50%
- Final Exam 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Sommerville, Ian. *Software engineering*. Boston: Pearson, 2016
2. Fowler, Martin. *UML distilled : a brief guide to the standard object modeling language*. Boston: Addison-Wesley, 2004
3. Pressman, Roger S. *Software engineering : a practitioner's approach*. New York, NY: McGraw-Hill Education, 2015
4. Software engineering, Ivan Marsic
5. Bourque, Pierre, and R. E. Fairley. *Guide to the software engineering body of knowledge*. Los Alamitos, Calif: IEEE Computer Society, 2014
6. Hunt, Andrew, and David Thomas. *The pragmatic programmer : from journeyman to master*. Reading, Mass: Addison-Wesley, 2000
7. Jalote, Pankaj. *An integrated approach to software engineering*. New York: Springer, 2005
8. Object-Oriented Software Engineering: Using UML, Patterns and Java, 3e
9. Dennis, Alan, et al. *Systems analysis and design*. Hoboken, NJ: Wiley, 2014
10. Booch, Grady. *Object-oriented analysis and design with applications*. Upper Saddle River, NJ: Addison-Wesley, 2007

SEM V

Algorithm Analysis

Program: Software Engineering and Computing Technology	
Module Title: *Computer Science*	Module Code: SITE-M1081
	Module ECTS: 28
Course Title: Algorithmic Analysis	Course Code: SITE-3084
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Algorithm design and analysis is a fundamental and important part of computer science. This course introduces students to advanced techniques for the design and analysis of algorithms, and explores a variety of applications.
Course Objectives
• Understanding of the design and analysis on algorithms for a variety of problems • Develop skills to reason about and prove properties of algorithms such as their correctness and running time. • Understand how data structures can provide space-efficient ways to quickly answer queries about data, and understand how these data structures can be used to build efficient algorithms. • Understand analysis techniques such as amortized analysis, probabilistic analysis, and minimax optimality. • Understand how and why randomness can be used to provide efficient solutions to some problems. • Learn to implement advanced algorithms efficiently. • Use the concept of approximation to find efficient algorithms for hard problems. • Use the theory of NP-completeness and other lower bounding techniques to argue for the difficulty of some problems.

Learning Outcomes

The programme provides a knowledge and understanding of the following:

Students who complete the course will have demonstrated the ability to do the following:

- Argue the correctness of algorithms using inductive proofs and invariants.
- Analyze worst-case running times of algorithms using asymptotic analysis.
- Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize divide-and-conquer algorithms. Derive and solve recurrences describing the performance of divide-and-conquer algorithms.
- Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize dynamic-programming algorithms, and analyze them.
- Describe the greedy paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize greedy algorithms, and analyze them.
- Explain the major graph algorithms and their analyses. Employ graphs to model engineering problems, when appropriate. Synthesize new graph algorithms and algorithms that employ graph computations as key components, and analyze them.
- Explain the different ways to analyze randomized algorithms (expected running time, probability of error). Recite algorithms that employ randomization. Explain the difference between a randomized algorithm and an algorithm with probabilistic inputs.
- Analyze randomized algorithms. Employ indicator random variables and linearity of expectation to perform the analyses. Recite analyses of algorithms that employ this method of analysis.
- Explain what amortized running time is and what it is good for. Describe the different methods of amortized analysis (aggregate analysis, accounting, potential method). Perform amortized analysis.
- Explain what competitive analysis is and to which situations it applies. Perform competitive analysis.
- Compare between different data structures. Pick an appropriate data structure for a design situation.
- Explain what an approximation algorithm is, and the benefit of using approximation algorithms. Be familiar with some approximation algorithms, including algorithms that are PTAS or FPTAS. Analyze the approximation factor of an algorithm.

Prerequisites:

Fundamental of Data Structures

Student Workload

Lecture: 48 hrs	**Tutorial: 32**	**Laboratory: 48**	**Home Study: 61**

Course Content

1. Introduction
2. Mathematics for Algorithmic
    - Sets
    - Functions and Relations
    - Vectors and Matrices
    - Linear Inequalities and Linear Equations
3. Greedy Algorithms
    - Knapsack Problem
        - O-I Knapsack
        - Fractional Knapsack
    - Activity Selection Problem
    - Huffman's Codes
    - Minimum Spanning Tree
    - Kruskal's Algorithm
    - Prim's Algorithm
    - Dijkstra's Algorithm
4. Divide & Conquer Algorithms
5. Dynamic Programming
    - Matrix-chain Multiplication
    - Knapsack Problem DP Solution
    - Activity Selection Problem DP Solution
6. Amortized Analysis
    - Aggregate Method
    - Accounting Method
    - Potential Method
    - Dynamic Table
7. Hash Table
8. Binary Search Tree
9. Graph Algorithms
    - Breadth First Search (BFS)
    - Depth First Search (DFS)
    - Topological Sort
    - Strongly Connected Components
    - Euler Tour
    - Generic Minimum Spanning Tree
    - Kruskal's Algorithm
    - Prim's Algorithm
    - Single Source Shortest Path
        - Dijkstra's Algorithm
        - Bellman-Ford Algorithm
10. String Matching
    - Naïve String Matching
    - Knuth-Morris-Pratt Algorithm
    - Boyer-Moore Algorithm
11. Linear-Time Sorting
    - Counting Sort

- Radix Sort
- Bucket Sort

12. Computational Complexity
    - Information-Theoretic Argument
    - Adversary Argument
    - NP-Completeness And Reduction
13. Approximate Algorithms
    - Vertex Cover
    - The Traveling Salesman Problem
14. Linear Programming

Schedule

Week 1 -4			
Week 4 - 6			
Week 6 - 9			
Week 9 - 16			

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The Mark Breakdown is as follows

- Lab –and Assignments 30%
- Exam – 45%
- Project – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Thomas Cormen, Charles Leiserson, Ronald Rivest and Clifford Stein, Introduction to Algorithms, MIT Press, 2009 (third edition).
2. Sedgewick, Robert, and Kevin Wayne. *Algorithms*. Upper Saddle River, NJ: Addison-Wesley, 2011
3. The Algorithm Design Manual (05 November 2010) by Steven S. Skiena
4. S. Dasgupta, C.H. Papadimitriou, and U. V. Vazirani, Algorithms, Mcgraw-Hill, 2006.
5. J. Kleinberg and E. Tardos, Algorithm Design, Addison-Wesley, 2006.

Department: Software Engineering	
Module Title: **Full Stack Development Fundamentals**	Module Code: *SITE*-**M2101**
	Module ECTS: 29
Course Title: *Mobile Application Development*	Course Code: **SITE-3105**
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

This course introduces mobile application development for the Android platform. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Students will learn skills for creating and deploying Android applications, with particular emphasis on software engineering topics including software architecture, software process, usability, and deployment.

Course Objectives

Understand the basic principles of mobile application development using android platform

Learning Outcomes

The program provides a knowledge and understanding of the following:

- User-interface design for mobile applications
- Managing application data
- Integrating mobile application with cloud services. Lecture hours: 3.
- Addressing enterprise requirements in mobile applications – performance, scalability, modifiability,
- availability and security. Lecture hours:
- Testing methodologies for mobile applications.
- Publishing, deployment, maintenance and management of mobile application.

- Effective communication skill acquired during group work
- A hand on experience on mobile technologies

Prerequisites: Object Oriented Programing

Student Workload

Lecture:	**Lab:**	**Home:**	**Tutorial:**

Course Content

Topic 1. Introduction
- Introduction to Android
- Android application structure
- Android UI architecture [application context ,intent ,activity lifecycle]
- UI Components [Text control,]

Topic 2. Notification ,Menus and Dialogs
- Intents with parameter
- Notification with status bar and toast
- Localization
- Context/option menu
- Alert dialog
- Dialog as activity

Topic 3. Location and Map
- How to work with Google Map
- How to work with GPS

Topic 4. Working with Data Storage
- Shared preference
- Working with files
- Working with SQL light

Topic 5. Content Providers
- Introduction to content provider
- Query provider

Topic 6. Network Communication
- Introduction to web service
- HTTP Client
- XML and JSON

Topic 7. Service
- Service Lifecycle
- Foreground Service

Topic 8. Publishing
- How to prepare for publishing
- Publishing to android market

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1- Week 4	Topic 1 ,2	– **Installation and configuration** – **Basics on android studio** – **UI design** – **Notification and**	Assignment 1 Assignment 2

		menu	
Week 5- Week 8	Topic 3,4	– **Configuration of Google map** – **Working on Google map** – **Working on GPS**	Assignment 3 Project 1
Week 9- Week 13	Topic 5,6	– **Testing query providers** – **Using web service ,http client and Json**	Assignment 4 Project 2
Week 14- Week 16	Topic 7,8	– **-how to work on foreground service** – **How to publish app**	Assignment 5 Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Assignment 15%
- Lab assessment 15%
- Project I and II 40%
- Final Exam 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Ramnath, Rajiv, Roger Crawfis, and Paolo Sivilotti. *Android 3 SDK programming for dummies*. Hoboken, N.J. Chichester: Wiley John Wiley distributor, 2011

Phillips, Bill, Chris Stewart, and Kristin Marsicano. *Android programming : the Big Nerd Ranch guide*. Atlanta, GA: Big Nerd Ranch, 2017

2. 4. Tomasz Nurkiewicz and Ben Christensen, Reactive Programming with RxJava, O'ReillyMedia, 2016.
3. Brian Fling, Mobile Design and Development, O'Reilly Media, Inc., 2009.

Maximiliano Firtman, Programming the Mobile Web, O'Reilly Media, Inc., 2nd ed., 2013.Cristian Crumlish and Erin Malone, Designing Social Interfaces, 2nd ed., O'Reilly Media, Inc., 2014.

4. Benjamin Muschko, Gradle in Action, Manning Publications, 2014.
5. Craig Larman, Applying UML and Patterns: A Guide to Object-Oriented Analysis and Design and Iterative Development, 3rd ed., Prentice Hall, 2004.

Department: Software Engineering	
Module Title: **Computer Systems and Programming**	Module Code: **SITE-M1071**
	Module ECTS: **17**
Course Tile: *Operating Systems*	Course Code:**SITE - 3073**
	Course ECTS: 7
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 3 Semester: 1 Meeting day: Meeting time: Meeting location:

Course Description

The Operating System is the name given to the pieces of software separating your application code from the details of the bare hardware. The function of an Operating System is to abstract away as many hardware details as possible and to present users and application programmers with a consistent interface to the computer system. Specific details about memory, hard disks, network cards, video hardware, etc. are all abstracted away so that a programmer need only think about abstractions like files and network sockets. However, to be effective, the operating system must do all this as transparently, fairly, and as efficiently as possible. These requirements are what make OS design and implementation a serious engineering challenge.

This course will serve as an introduction to Operating Systems and System Programming in general. The course will cover the basics of OS design including: file systems, processes, threads, and virtual memory. Lectures will cover both theoretical issues as well as implementation.

Course Objectives

The following are the main objectives of the course.

- Explain the structure and functions of an operating system.
- Explain processes and threads and how they are implemented by the operating systems.
- Explain the different states that a task may pass through and the data structures needed to support the management of many tasks.
- Discuss and use different inter process communication mechanisms implemented by the operating system
- Describe concurrency and reasons for its importance.
- Describe scheduling and illustrate how it works to improve computer performance.
- Illustrate key operating system aspects by concrete examples.
- Investigate the wider applicability of scheduling in such contexts as disk I/O, networking scheduling.
- Introduce memory hierarchy and cost-performance tradeoffs,
- Explain virtual memory and its realization in hardware and software,
- Explain disk structures, disk scheduling and file systems.
- Explain how I/O is managed by the operating system
- Discuss the implications and implementations of security and protection in relation with the operating system.

Learning Outcomes

By the end of the course you should be able to

- Describe the general architecture of computers
- Describe, contrast and compare differing structures for operating systems
- Understand and analyse theory and implementation of: processes, resource control (concurrency etc.), physical and virtual memory, scheduling, I/O and files

Prerequisites: Systems programming

Student Workload

Lecture: 32	**Tutorial: 16**	**Home Study**	**Laboratory: 64**

Course Content

Part 1: Computer system and Operating system overview

- Overview of computer systems, hardware, instruction execution, I/O structures, Interrupts, memory hierarchy, introduction and history of operating systems, OS structure, OS functions and services.

Part 2: Processes Management

- Processes: processes, context switching, process states, process representation
- Inter process communication: shared memory communication, message passing communication, direct and indirect communication, mailboxes, message passing

interface, message queues, pipes, sockets.
- Threads: multithreading, thread representation and execution, thread usage and benefits, types of threads, implementation of threads.
- Process Synchronization: race conditions, critical section problem, locks, mutexs, semaphores, monitors, non-blocking synchronization.
- CPU Scheduling: introduction, scheduling metrics, scheduling algorithms, FCFS, round robin, SJF, SRTF, priority scheduling, multilevel queues and multilevel queue scheduling
- Deadlock: deadlocks, methods of handling deadlocks, deadlock detection, deadlock prevention, deadlock avoidance

Part 3: Memory Management
- Main memory: Background, swapping, memory allocation, segmentation, paging, multilevel addresses, address translation, caching, TLB's.
- Virtual memory: demand paging, copy on write, page replacement, and thrashing.

Part 4: Storage Management
- Mass storage structure: disk structure, disk scheduling, disk management
- File systems: file system interface, files and directories, access methods, file sharing, file system mounting, file system structure and implementation, allocation methods, free space management
- I/O systems: I/O hardware, application I/O interface, kernel I/O subsystem.

Part 5: Security and Protection
- The security problem, program threats, system and network threats, goals and principles of protection, security and protection mechanisms.

Content of Laboratory Manuals

Session 1: Process management concepts.
- Investigating Linux OS commands for processes management as well as writing C programs that take advantage of system calls define in Linux OS for process management.

Session 2: Threads.
- Implementing threads in Linux OS using POSIX Thread.

Session 3: Inter process communication
- Investigating different inter-process communication methods implemented in Linux OS and implementing different programs using those IPC methods.

Session 4: Synchronization I.
- Investigating Linux OS synchronization mechanisms including locks and mutexes.

Session 5: Synchronization II.

- Investigating Linux OS synchronization mechanisms including semaphores and monitors.

Session 6: File and I/O.

- Investigating Linux OS File and I/O facilities and writing C programs that take advantage of system calls define in Linux OS for system level I/O and files.

Session 7: Linux kernel modules

- Developing and implementing Linux kernel modules.

Session 8: Virtual Memory

- Investigating Linux OS virtual memory implementation and writing programs that take advantage of it.

Session 9: System calls

- Developing and implementing Linux system call.

Session 10: Device drivers

- Developing and implementing device drivers.

Schedule

Week	Topics and Sub Topics		
Week 1 - Week 4			
Week 5 - Week 8			
Week 9 - Week 12			
Week 13 - Week 16			

Assessment Method:

Labs (10 %)

Projects (40 %)

Mid exam (20 %)

Final exam (30 %)

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted

- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

Textbook:

1. *Operating System Concepts (10th ed.)* Abraham Silberschatz, Peter Baer Galvin, Greg Gagne.

Reference Books:

2. *Modern Operating Systems (3rd edition),* Andrew S. Tanenbaum, Prentice Hall, 2008.
3. *Operating Systems Design and Implementation (3rd edition),* Andrew S. Tanenbaum and Albert S,
4. *Linux Kernel Development (3rd edition),* Robert Love

Program: Software Engineering and Computing Technology	
Module Title: Computer Architecture and Design	**Module Code: SITE-M2111**
	Module ECTS: 19
Course Title: *Digital Logic Design*	**Course Code: SITE-3112**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

An introduction to digital hardware systems design with an emphasis on practical design techniques and circuit implementations. Topics include: Data representation in digital systems; Boolean algebra; the analysis, design, optimization and implementation of combinational and sequential circuits; modern digital circuit technologies; use of Hardware Description Languages (HDLs) and Computer Aided Design (CAD) tools in digital systems design.

Course Objectives

This is the first course in computer hardware design, which covers the fundamentals of digital logic design and switching theory. During the course, the student is expected to learn about the following:

- Number systems, binary systems, base conversion, representation of numbers and characters using binary codes, and binary arithmetic.
- Boolean algebra and its laws, axioms, theorems, and operations
- Manipulation and simplification of Boolean algebraic expressions and functions using Boolean theorems and K-maps.
- Combinational devices such as multiplexers, decoders, and adders.
- Analysis and design of combinational circuits using basic gates and/or combinational devices.
- Latches and flip-flops (SR, JK, D, and T).
- Analysis and design of synchronous sequential logic circuits.
- Registers and counters: Their design and use in implementing specific

operations.

Learning Outcomes

Upon successful completion of this course, students should:

- Have an overview of digital systems and their applications.
- Have a thorough understanding of different numbering systems (with emphasis on the binary, decimal, octal, and hexadecimal systems) and be able to convert from one base to another.
- Understand how signed numbers, unsigned numbers, and alphanumeric characters are represented in binary.
- Be able to perform binary arithmetic
- Comprehend the basic definitions, axioms, and theorems of Boolean algebra.
- Be able to manipulate and simplify Boolean algebraic expressions and functions of different forms.
- Understand the basic logic operations and gates.
- Be able to use Karnaugh maps to minimize Boolean expressions and functions.
- Be able to analyze and design digital combinational circuits using basic gates (AND, OR, NOT, NAND, NOR, XOR).
- Understand the design and operation of decoders, encoders, multiplexers, comparators, adders, and subtractors.
- Be able to analyze and design combinational logic circuits using decoders and multiplexers.
- Understand the design and operation of different latch and flip-flop types (SR, JK, D, and T).
- Be able to analyze and design clocked synchronous sequential circuits.
- Understand the operation and be able to design different types of registers and counters using flip-flops.
- Have good understanding of programming logic devices and their use in digital systems design.
- Be familiar with hardware description languages and CAD tools for digital logic synthesis and simulation.

Prerequisites: Fundamental of Electrical circuits and electronics

Student Workload

Lecture:32 hrs	**Tutorial: 32hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

- Introduction to modern digital logic design
- Combinational logic
    - Switch logic and basic gates
    - Boolean algebra
    - Two-level logic

    - Regular logic structures
    - Multi-level networks and transformations
    - Programmable logic devices
    - Time response
    - Case studies
- Sequential logic
    - Feedback Concept
    - Basic latches and flip-flops
    - Timing methodologies
    - Registers and counters
    - Programmable logic devices
    - Case studies
- Finite state machine design
    - Concepts of FSMs
    - Basic design approach
    - Specification methods
    - State minimization
    - State encoding
    - FSM partitioning
    - Implementation of FSMs
    - Programmable logic devices
    - Case studies
- Elements of a Processor
    - Arithmetic circuits
    - Arithmetic and logic units
    - Register and bus structures
    - Controllers
- Computer-aided design tools for logic design
    - Schematic entry
    - Hardware description language entry
    - Compilation to logic networks
    - Simulation
    - Mapping to programmable logic devices

References

1. Fundamentals of Digital Logic with Verilog Design, Stephen Brown and Zvonko Vranesic, 3rd Edition
2. Digital Design, Morris Mano, 5th Edition
3. Digital Design: Principles and Practices and Applications, John F. Wakerly, 4th Edtion, Prentice Hall.

4. Digital Design and Computer Architecture, Second Edition, David Harris, Sarah Harris, 2nd Edition

Computer Graphics

Program: Software Engineering and Computing Technology	
Module Title: *Graphics and HCI*	Module Code: **SITE-M3121**
	Module ECTS: 9
Course Title: *Computer Graphics*	Course Code:**SITE-3121**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

This course offers an in-depth exploration of fundamental concepts in 2D and 3D computer graphics. The course provides introduction to computer graphics algorithms, software and hardware. Topics include: ray tracing, the graphics pipeline, transformations, texture mapping, shadows, sampling, global illumination, splines, animation and color. The course will assume a good background in programming in C or C++ and a background in mathematics including familiarity with the theory and use of coordinate geometry and of linear algebra such as matrix multiplication. Students will use the standards based on OpenGL library in several programming projects illustrating the theory and practice of programming computer graphics applications.

Course Objectives

The objective the course is to introduce students to the basic mechanics of drawing vectors images of 2D and 3D using the latest OpenGL and the Graphics Pipeline.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Understand the structure of modern computer graphics systems
- Understand the basic principles of implementing computer graphics primitives

- Familiarity with key algorithms for modelling and rendering graphical data
- Develop design and problem-solving skills with application to computer graphics
- Gain experience in constructing interactive computer graphics programs using OpenGL

- Students will be able to work on projects that contain simulations and visualizations in different fields like Biology, Physics

Prerequisites: Linear Algebra, Applied Math I

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study:39 hrs.**	**Laboratory: 32 hrs**

Course Content

Chapter 1 – Introduction to Computer Graphics
- What's is Computer Graphics?
- History of Computer Graphics
- Application of Computer Graphics

Chapter 2 – OpenGL and Shaders
- OpenGL and it's Architecture
- Shaders Language
- Data in OpenGL Buffers

Chapter 3 – Vector Geometry
- Lines and Planes
- Distances
- Angles
- Intersections
- Homogeneous Coordinates

Chapter 4 – Transformations
- Linear Transformations
- Three Dimensions
- Affine Transformations

Chapter 5 – Viewing
- Camera Model
- View Transformation and Shaders
- Clipping

Chapter 6 – Curve, Splines and Surfaces
- Curves Descriptions
- Bezier Curves
- B-Splines
- Nurbs

- Surfaces

Chapter 7 – Textures and Materials
- Texture Mapping
- Creating and Initializing Textures
- Specifying texture Data in OpenGL
- Complex Texture Types
- Filtering
- Rendering to Textures

Chapter 8 – Lighting and Shadow
- Basic Lighting
- Materials
- Lighting Maps
- Shadows
- HDR

Chapter 9 – Raytracing
- Shadows
- Reflection
- Refraction

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Chapter 1		
Week 2	Chapter 2	**Lab 1 – Hello OpenGL**	
Week 3	Chapter 3	**Lab 2 – OpenGL Drawing Shapes**	Assignment I
Week 4	Chapter 4	**Lab 3 – Applying Transformations**	
Week 5	Exam		
Week 6	Chapter 5	**Lab 4 – Applying Camera**	Assignment II
Week 7 – Week 8	Chapter 6	**Lab 5 – Drawing curved surfaces**	
Week 9 – Week 10	Chapter 7	**Lab 6 – Applying Texture and Material**	Assignment III
Week 11	Exam		
Week 12	Chapter 8	**Lab 7 – Applying different kinds of lighting**	Assignment IV

Week 13 – Week 14	Chapter 9	**Lab 8 – Implementing Simple Graphics Rendering Engine**	Assignment V
Week 15	Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Lab – 20%
- Exam – 60%
- Project – 20%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Janke, Steven J. Mathematical structures for computer graphics. Hoboken, New Jersey: Wiley, 2015
- Computer Graphics: Principles and Practice 3rd Edition 3rd Edition by John F. Hughes,ʹ Andries van Dam, Morgan McGuire, David F. Sklar,ʹ James D. Foley,ʹ Steven K. Feiner ,ʹ Kurt Akeley
- Kessenich, John M., Dave Shreiner, and Graham Sellers. OpenGL programming guide : the official guide to learning OpenGL, version 4.5 with SPIR-V. Boston: Addison-Wesley, 2017
- Lengyel, Eric. Mathematics for 3D game programming and computer graphics. Boston: Course Technology PTR, 2012

SEM VI

Fundamental of IT security

Program: Software Engineering and Computing Technology	
Module Title: Security Fundamentals	Module Code: SITE-M3131
	Module ECTS: 5
Course Title: Fundamentals of IT Security	Course Code: SITE-3131
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: Semester: I Meeting day: Meeting time: Meeting location:

Course Description
This course provides a one-semester overview of information security. It is designed to help students with prior computer and programming knowledge — both undergraduate and graduate — understand this important priority in society today. The technical content of the course gives a broad overview of essential concepts and methods for providing and evaluating security in information processing systems (operating systems and applications, networks, protocols, and so on). In addition to its technical content, the course touches on the importance of management and administration, the place information security holds in overall business risk, social issues such as individual privacy, and the role of public policy.
Course Objectives
Students successfully completing this course should be able to: • Provide fundamental concepts in IT systems security • Enable students to understand threats to security of IT systems • Introduce students to the principles of secure system design

- Introduce students to the fundamentals of cryptography
- Introduce students to the security issues and defense techniques in operating systems, networks, web applications, database and mobile devices
- Introduce students to the state-of-the-art techniques for vulnerability analysis, prevention, and defense

Learning Outcomes

The Course provides a knowledge and understanding of the following:

- Be familiar with security issues and defense mechanisms in operating systems, networks, web applications, and mobile devices
- Be conversant with automated analysis of security threats, design and implementation of countermeasures
- Possess solid understanding of principles and practices of computer systems security
- Be able to recognize, explain, and act-on threats to confidentiality, integrity, and availability of a computer system
- Apply formalisms and technologies to defend computer systems from security threats

Prerequisites:

No-prerequisite

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

Chapter One: Introduction of IT security

- What is security?
- Key Objectives of Security
- Computer Security Challenges
- Security Services and Mechanisms
- Security Attacks
- Security Techniques
- Model for Network Security

Chapter Two: Cryptography

- Basic of Cryptography
- Overview of cryptanalysis
- Encryption Techniques
- Symmetric cryptography (DES, AES)
- Public key cryptography (RSA)
- Key management and exchange
- Cryptographic hash functions and certificates

Chapter Three: Application and OS security

- Application code attacks (buffer overflow)
- Prevention: System design, robust coding, isolation.
- Users management

- Enforcement of security
- OS Access control, authentication
- Risk management
- Information Security: Policies and Procedures
- Legal Issues and Information Security: Computer Forensics

Chapter Four: Web security

- Website attack and defenses
- Browser policies, session management, user authentication
- HTTPS and web application security

Chapter Five: Database security

- Database attack and Defenses
- SQL Injection and its Prevention

Chapter Six: Network security

- Network protocols and vulnerabilities,
- Passive vs Active Attacks
- Network Defenses:
    - Kerberos, IPSec,
    - SSL/TLS,
    - Firewall, IDS/IPS
    - Malware,
    - botnets, DDoS, network security testing.

Chapter Seven: Mobile Security

- Mobile security model
- Vulnerabilities
- Prevention and defense

Chapter Eight: Security Evaluation

- Parameters for Security Evaluation
- Security Assessment Components
    - Recon: Information Gathering
    - Scanning
    - Exploitation
    - Covering streams
- Network Security Assessment
- System Security Assessment
- Application Security Assessment
- Physical Security Assessment
- Reporting and Follow-up

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 – week 2	Chapter One		Assignment #1. Lab#1,

			Lab#2 Project Title Approval
Week 3 – week 4	Chapter Two		Lab#3, Lab#4 Quiz #1 Project Proposal
Week 5 – week 8	Chapter Three		Lab#4, Lab#5 Mid_exam Project Progress Report
Week 9 – week 10	Chapter Four		Lab#6, Lab#7
Week 11 – week 12	Chapter Five		Lab#8, Lab#9 Project Progress Report Quiz #2
Week 13 – week 14	Chapter Six		Lab#10, Lab#11 Final Project Report Lab exam
Week 15 – week 16	Exam		Final Exam

Teaching Methods

- The format of the course will be lecture, class discussion and presentation, lab exercise, assignments and Project.
- Students will work on Windows/Linux and multiple security tools

Assessment Method

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. Neil Daswani, Christoph Kern, and Anita Kesavan. Foundations of Security: What Every Programmer Needs to Know. Apress, 2007
2. Matt Bishop. Introduction to Computer Security. Addison-Wesley, 2004
3. Charles P. Pfleeger and Shari L. Pfleeger. Security in Computing (4th Edition). Prentice-Hall, 2006

4. Dieter Gollmann. Computer Security (2nd Edition). Wiley, 2006
5. John Viega, Gary McGraw. Building Secure Software: How to Avoid the Security Problems the Right Way. Addison-Wesley, 2002
6. William Stallings. Cryptography and Network Security: Principles and Practice (6th Edition). Prentice-Hall, 1998
7. R. Baloch, Ethical Hacking and Penetration Testing Guide, 6000 Broken Sound Parkway NW, Suite 300: CRC press, 2015.
8. Hacking exposed 7th edition
9. Hacking – The art of exploitation 2nd edition
10. Computer Security: Art and Science (Crypto)

Program: Software Engineering and Computing Technology	
Module Title: *Artificial Intelligence*	Module Code: SITE-M3111
	Module ECTS: 5
Course Title: *Fundamentals of AI*	Course Code: SITE-3141
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

Artificial intelligence (AI) is a research field that studies how to realize the intelligent human behaviors on a computer. The ultimate goal of AI is to make a computer that can learn, plan, and solve problems autonomously. Although AI has been studied for more than half a century, we still cannot make a computer that is as intelligent as a human in all aspects. However, we do have many successful applications. In some cases, the computer equipped with AI technology can be even more intelligent than us. The Deep Blue system which defeated the world chess champion is a well-know example.

The main research topics in AI include: problem solving, reasoning, planning, natural language understanding, computer vision, automatic programming, machine learning, and so on. Of course, these topics are closely related with each other. For example, the knowledge acquired through learning can be used both for problem solving and for reasoning. In fact, the skill for problem solving itself should be acquired through learning. Also, methods for problem solving are useful both for reasoning and planning. Further, both natural language understanding and computer vision can be solved using methods developed in the field of pattern recognition.

Course Objectives

The primary objective of this course is to provide an introduction to the basic principles and applications of Artificial Intelligence. Programming assignments are used to help clarify basic concepts. The emphasis of the course is on teaching the fundamentals, and not on providing a mastery of specific commercially available

software tools or programming environments. In short, this is course is about the design and implementation of intelligent agents---software or hardware entities that perform useful tasks with some degree of autonomy. Upon successful completion of the course, students will have an understanding of the basic areas of artificial intelligence including problem solving, knowledge representation, reasoning, decision making, planning, perception and action, and learning -- and their applications (e.g., big data analytics, data mining, computational discovery, information retrieval).

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Find appropriate idealizations for converting real world problems into AI search problems formulated using the appropriate search algorithm.
- Given a search problem, analyze and formalize the problem (as a state space, graph, etc.), select the appropriate search method, and write the algorithm for it.
- Explain important search concepts, such as the difference between informed and uninformed search, the definitions of admissible and consistent heuristics and completeness and optimality. Give the time and space complexities for standard search algorithms.
- Implement A* and iterative deepening search. Derive heuristic functions for A* search that are appropriate for a given problem.
- For constraint satisfaction problems, implement backtracking search with conflict directed backjumping, arc consistency, and the Minimum Remaining Values and Least Constraining Value heuristics. Implement local search with the mi conflicts heuristic.
- Implement and execute by hand alpha-beta search. Design good evaluation functions and strategies for game playing.
- Carry out proofs in first order and propositional logic using techniques such as resolution, unification, backward and forward chaining.
- Explain the difference between plan space and state space. Describe and implement several of the major approaches to classical planning, including planning graphs, POP, and propositionalization.
- Design appropriate Bayes Nets corresponding to the causal relationships and conditional independence of a real-world situation. Derive the conditional independence of variables given a Bayes Net.
- Given a real world supervised learning problem, choose and implement appropriate learning algorithms such as decision trees, support vector machines, and boosting.

Prerequisites:

Probability and Statistics
Linear Algebra
Math I

Student Workload			
Lecture:32 hrs.	Tutorial: 32 hrs	Home Study:65 hrs.	Laboratory: 48hrs

Course Content

Chapter 1. Artificial Intelligence
- a. Introduction
- b. Intelligent Agents

Chapter 2. Problem-solving
- a. Solving Problems by Searching
- b. Beyond Classical Search
- c. Adversarial Search
- d. Constraint Satisfaction Problems

Chapter 3. Knowledge, reasoning, and planning
- a. Logical Agents
- b. First-Order Logic
- c. Inference in First-Order Logic
- d. Classical Planning
- e. Planning and Acting in the Real World
- f. Knowledge Representation

Chapter 4. Uncertain knowledge and reasoning
- a. Quantifying Uncertainty
- b. Probabilistic Reasoning
- c. Probabilistic Reasoning over Time
- d. Making Simple Decisions
- e. Making Complex Decisions

Chapter 5. Learning
- a. Learning from Examples
- b. Knowledge in Learning
- c. Learning Probabilistic Models
- d. Reinforcement Learning

Chapter 6. Communicating, perceiving, and acting
- a. Natural Language Processing
- b. Natural Language for Communication
- c. Perception
- d. Robotics

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 – Week 2	Chapter 1	Lab 1	

Week 3 – Week 4	Chapter 2	Lab 2	Assignment I
Week 5 – Week 7	Chapter 3	Lab 3	Assignment II
Week 8	Mid Exam		
Week 9 – Week 11	Chapter 4	Lab 4	Assignment III
Week 12 – Week 13	Chapter 5	Lab 5	
Week 14 – Week 15	Chapter 6	Lab 6	Assignment IV
Week 16	Final Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 6 Labs – 10%
- 1 Project – 20%
- 2 Exams – 60%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Artificial Intelligence: A Modern Approach (3rd Edition) 3rd Edition *by Stuart Russell, Peter Norvig.*
- Introduction to Linear Algebra, 5th Edition *by Gilbert Strang*

Program: Software Engineering and Computing Technology	
Module Title: *Graphics and HCI*	Module Code: **SITE-M3121**
	Module ECTS: 9
Course Title: *Human Computer Interaction(HCI)*	Course Code: **SITE-3122**
	Course ECTS: 4
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 20 _ _
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
* Background--the development and scope of HCI. Practical goals. * HCI relevant issues in human perception, memory and thinking processes. * Approaches to designing information appliances--software objects and physical things. * Design methodologies and notations--levels of interface design. Task analysis, grammars, state charts. * Techniques and technologies--dialogue styles, information presentation, protocols for human-to-machine and machine-to-machine interactions; mobile computing, distributed wireless computation, wireless sensors. * The design process--user involvement, iterative design, prototyping. * Evaluation--methodologies, formative and summative. Performance analysis. * Specific issues in HCI: the internet of things; novel interfaces.
Course Objectives
After completing this course you will be able to understand and make use of The human components functions The computer components functions The interaction between the human and computer components Paradigms

Interaction design basics
HCI in the software process
Design rules
Implementation Supports
Evaluation Techniques

Learning Outcomes

1 - Demonstrate, in writing, knowledge of the issues and problems in HCI.
2 - Demonstrate an understanding of human perception and behaviour in analysing their interactions with technology in their everyday lives.
3 - Use established design principles and methodologies to solve HCI problems.
4 - Acquire confidence in handling different disciplinary perspectives on HCI and the ability to apply them to design problems.
5 - The ability to devise, plan and execute task analysis and system evaluation studies from an HCI perspective, and present findings in a clear and effective manner.
6 - Demonstrate awareness of current areas of research by locating and summarising examples of recent progress.

Prerequisites: None

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

Chapter 1: Introduction

- Introduction To User Interface
- Importance Of User Interface
- Definition Of User Interface
- Importance Of Good Design
- The Benefits Of Good Design
- A Brief History Of The Human-computer Interface
- A Brief History Of Screen Design

Chapter 2: The Graphical And Web User Interface

- Introduction To Graphical User Interface
- Popularity Of Graphics
- The Concept Of Direct Manipulation
- Graphical Systems: Advantages And Disadvantages
- Characteristics Of The Graphical User Interface
- Introduction To Web User Interface
- Popularity Of The Web
- Characteristics Of Web Interface
- The Merging Of Graphical Business Systems And The Web
- Principles Of User Interface Design

Chapter 3: Design Process

- Introduction

- Human Interaction With Computers
- Importance Of Human Characteristics
- Human Considerations In Design
- Human Interaction Speeds
- Understanding Business Functions

Chapter 4: Screen Designing
- Introduction To Screen Designing
- Design Goals
- Screen Planning And Purpose
- Organizing The Screen Elements
- Ordering Of Screen Data And Content
- Screen Navigation And Flow
- Visually Pleasing Composition
- Amount Of Information
- Focus And Emphasis
- Presenting Of Information Simply And Meaningfully
- Guidelines To Organize Display/ Read-only Screens
- Information Retrieval On Web
- Statistical Graphics
- Technological Considerations In Interface Design

Chapter 5: Windows
- New And Navigation Schemes
- Selection Of Window
- Selection Of Devices Based Controls
- Selection Of Screen Based Controls

Chapter 6: Components
- Text And Messages
- Icons And Images
- Multimedia
- Colours Uses And Problems
- Choosing Colours

Chapter 7: Software Tools
- Introduction
- Specification Methods
- Interface Building Tools

Chapter 8: Interaction Devices
- Introduction To Interaction Devices
- Keyboards And Function Keys
- Pointing Devices
- Speech Recognition, Digitization And Generation
- Image And Video Displays

- Printers

Schedule

Weeks	Topics and Subtopics		
	Lecure	**Laboratory**	**Assessment**
Week 1- Week 4	Chapter 1- Chapter 2		Assignment I (practical)
Week 5- Week 8	Chapter 3, Chapter 4		Mid Exam
Week 9- Week 13	Chapter 5 - Chapter 6		Assignment II (practical)
Week 14- Week 16	Chapter 7		Assignment III Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Tests 48%
- Assignments 12%
- Final Exam 40%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

1. Human Computer Interaction, 3rd Edition, Prentice Hall.
2. Rogers, Sharp & Preece, Interaction Design, 3rd Edition, John Wiley & Sons, 2011.
3. HCI Models, Theories and Frameworks: Toward a multidisciplinary science. Edited by John. Carroll. Morgan Kaufmann 2003
4. William M. Newman, & Michael G. Lamming, Interactive System Design. Addison-Wesley.

Program: Software Engineering and Computing Technology	
Module Title: ***Computer Science***	**Module Code: SITE-M1081**
	Module ECTS: 28
Course Tile: ***Design Pattern***	**Course Code: SITE-3085**
	Course ECTS: 4
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
This course is designed for students with Java background. The course objective is to familiarize the student with techniques for designing reusable combinations of Java classes and organizing their cooperation to produce modular and maintainable Java programs
Course Objectives
Upon completion of the course, the student will be able to understand most important design patterns and apply object-oriented techniques for designing reusable, maintainable and modifiable software
Prerequisites: Object Oriented Programming
Student Workload

Lecture:	**Tutorial**	**Home Study**	**Laboratory**

Course Content
Unit 1

Course introduction. Review of the software development context in relation to design patterns. Advantages of using classes.

The uses of inheritance in software design. Principles of strong typing and substitution. Polymorphism with abstract classes and interfaces.

Unit 2

Design techniques for modifiability. The Strategy Pattern

Unit 3

The Observer Pattern

Unit 4

The Decorator Pattern, the Simple Factory Pattern

Unit 5

The Factory Methods, the Abstract Factory Patterns

Unit 6

The Singleton Pattern, the Command Pattern

Unit 7

The Adapter and Facade Patterns

Unit 8

The Template Method Pattern

Unit 9

The Iterator and Composite Patterns

Unit 10

The State Pattern

Unit 11

The Proxy Pattern

Unit 12

Compound Patterns

Assessment Method:

- Assignments = 15%
- Projects = 25%
- Midterm = 25%
- Final Exam = 35%

Course Policies

Academic Honesty: Academic dishonesty including cheating, fabrication and plagiarism will not be tolerated in this course. It is expected that all work handed in by the student is their own. You will be reported to concerned bodies automatically including the center's academic council. The center deals with such academic dishonesty severely.

While we expect you to work individually, that doesn't mean that you cannot share ideas with other students or us. But you can only share ideas. No code or written work should be shared amongst you. And such an activity is considered academic dishonesty.

You are expected to attend at least 85% of the classes according to the university senate legislation. We will not allow you to sit for the final if you don't meet the criteria.

References

- Eric Freeman, Elisabeth Freeman, Kathy Sierra, Bert Bates Head First Design Patterns, 2004
- Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides Design Patterns: Elements of Reusable Object-oriented Software Addison-Wesley, 1995

Program: Software Engineering and Computing Technology	
Module Title: Computer Architecture and Design	**Module Code: SITE-M2111**
	Module ECTS: 19
Course Title: *Computer Organization and Architecture*	**Course Code: SITE-3113**
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

Computer architecture is the science and art of selecting and interconnecting hardware components and designing the hardware/software interface to create a computer that meets functional, performance, energy consumption, cost, and other specific goals. This course introduces the basic hardware structure of a modern programmable computer, including the basic laws underlying performance evaluation. Topics include, Overview of computer technologies, instruction set architecture (ISA), ISA design considerations, RISC vs. CISC, assembly and machine language, translation and program start-up. Computer arithmetic, arithmetic logic unit, floating-point numbers and their arithmetic implementations. Processor design, data path and control implementation, pipelining, hazards, pipelined processor design, hazard detection and forwarding, branch prediction and exception handling. Memory hierarchy, principles, structure, and performance of caches, virtual memory, segmentation and paging. I/O devices, I/O performance, interfacing I/O. Introduction to multiprocessors, multicores, and cluster computing.

Course Objectives

- Discuss the basic concepts and structure of computers.
- Understand concepts of register transfer logic and arithmetic operations.
- Explain different types of addressing modes and memory organization.
- Learn the different types of serial communication techniques.
- Summarize the Instruction execution stages.

Learning Outcomes

After having completed the course the student will be able to:

- Calculate the performance of a modern digital computer from parameters such as processor speed, cycles per instruction, and profiling results for various algorithms in a benchmark;
- display sterling understanding of the instruction set of a modern RISC processor;
- formulate instructions of a high-level imperative language in terms of the instruction set of a RISC processor;
- explain how fixed-point and floating-point numbers are represented in a computer;
- describe hardware algorithms for fixed-point arithmetic;
- design, implement, and document (with a user manual in English) a disassembler for a modern RISC processor which reads words from instruction memory and decomposes them into the individual fields which may be fed to an emulator;
- display wide understanding of data and control flow in a modern RISC processor;
- discuss how pipelining in a processor functions and describe how hazards are resolved in various ways, including bubbles, flushes, and forwarding;
- design, implement, and document (with a user manual in English) an emulator for a modern RISC processor; which models faithfully the flow of control;
-
- * display wide understanding of how memory is organized and managed in a modern digital computer, including virtual and physical memory, address translation, multilevel, unified, and multi-way set-associative caches, the translation-lookaside buffer (TLB), and the page table.
- compute how large a cache must be, given the size of the index, the size of the tag, and the level of associativity;
- compute the performance of various forms of memory from given parameters;
- discuss input-output units, including in particular hard disks and solid-state disks (SSDs), how they communicate with the processor, and how their performance is computed.

Prerequisites: Digital Logic Design

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

- History of the development of the digital computer
- Information theory and digital representations of data
- Computer system organization
- Instruction set architecture (ISA)
    - Instruction types, registers, addressing modes, load/store architectures
- Instruction sequencing, conditional branches, and subroutines

- Procedure linkage, activation records, stacks, and low-level support for recursion
- Structure of assembly language programs
- Architectural support for high-levellanguages
- From high-level to machine language
  - Roles of compilers, assemblers, linkers and loaders
- Design and organization of a one-instruction-per-clock computer
- Techniques for improving computer performance
  - Pipelining and interleaving. Pipelining impact on the ISA and system architecture. Pipeline hazards, interlocks, and branch delay slots. Improving memory performance with caches
- Virtualizing and sharing computers
  - Memory management, virtual memory. Time-sharing and process management.
  - Motivate the need for an operating system
- Multi-core and parallel processing

References

1. M. Morris Mano, Computer System Architecture,William Stallings, 4th Ed, 2012.
2. Hennessy, John L., David A. Patterson, and Krste Asanovic. *Computer architecture : a quantitative approach*. Cambridge, MA: Morgan Kaufmann Publishers, an imprint of Elsevier, 2019
3. Patterson, David A., and John L. Hennessy. *Computer organization and design : the hardware/software interface*. Oxford Waltham, MA, USA: Morgan Kaufmann, 2014

Program: Software Engineering and Computing Technology	
Module Title: **Programming**	Module Code: **SITE-M2091**
	Module ECTS: 11
Course Tile: *Programming Paradigms*	Course Code: **SITE-3092**
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course will introduce students with the fundamental ideas surrounding the design and implementation of high level programming languages. The course stress on underlying theoretical concepts as well as significant practical sessions. Students will gain experience with different programming languages and their basic structures. It also introduces language theory and grammars.

Course Objectives

- Be aware of some of the wide variety of concepts available in various programming languages.
- Understand basic representation of languages
- Be aware of some of the many trade-offs in programming language design and implementation
- Understand the different phases of a compiler
- Understand the internal working principles of a compiler

Prerequisites: Data Structures, Programming

Student Workload

Course Content

Chapter 1: Introduction to programming languages

- Definitions of programming languages
- The lexical and syntactic structure of a language
- Evolution of Programming Languages
- High-level programming languages
- Elements of Programming languages: Data elements, Data structures, Operators, Assignment, Statements, Program units, Data environments, Parameter transmission, Storage management.

Chapter 2: Program language translation

- Language Processors: Compilers and translators, interpreters, pre-processors, assemblers, linkers, loaders.
- Introduction to Compilers: The structure of a compiler, Lexical analysis, Syntax analysis, Intermediate code generation, Optimization, Code generation, Bookkeeping, Error handling, Compiler writing tools
- Programming Language basics:

Chapter 3: Finite automata and lexical analysis

- The role of the lexical analyzer: Lexical scanning, token classes, keyword recognition.
- A simple approach to the design of lexical analyzers
- Strings and languages
- Regular expressions
- Finite automata (DFA and NFA)
- From regular expressions to finite automata
- Minimizing the number of states of a DFA
- A language for specifying lexical analyzers
- Implementation of a lexical analyzer

Chapter 4: Syntax Analysis

- Context-free grammars
- Derivations and parse trees
- Capabilities of context-free grammars
- Basic Parsing Techniques: Parsers, Shift-reduce parsing, Operator-precedence parsing, Top-down parsing, Predictive parsers, LR Parsers

Chapter 5: Syntax-Directed Translation

- Syntax-directed translation schemes
- Implementation of syntax-directed translators
- Intermediate code, Postfix notation, Parse trees and syntax trees, Three-address code, quadruples, and triples, Translation of assignment statements, Boolean expressions, Statements that alter the flow of control, Postfix translations, Translation with a top-down parser
- More about Translation: Array references in arithmetic expressions,Procedure calls, Declarations, Case statements, Record structures

Chapter 6: Symbol Tables and Errors

- The contents of a symbol table, Data structures for symbol tables, Representing scope information
- Symbol tables: Implementation of a simple stack allocation scheme, Implementation of block-structured languages, Storage allocation in block-structured languages
- Error detection and recovery: Errors, Lexical-phase errors, Syntactic-phase errors, Semantic errors

Chapter 7: Code optimization and code generation

- Introduction to code optimization:
- The principle sources of optimization: Loop optimization: The DAG representation of basic blocks, Value numbers and algebraic laws, Global data-flow analysis
- Code generation: Object programs, Problems in code generation, A machine model, A simple code generator, Register allocation and assignment, Code generation from DAG's, Peephole optimization

Assessment Method:

- Assignments = 15%
- Projects = 25%
- Midterm = 25%
- Final Exam = 35%

Course Policies

Academic Honesty: Academic dishonesty including cheating, fabrication and

plagiarism will not be tolerated in this course. It is expected that all work handed in by the student is their own. You will be reported to concerned bodies automatically including the center's academic council. The center deals with such academic dishonesty severely.

While we expect you to work individually, that doesn't mean that you cannot share ideas with other students or us. But you can only share ideas. No code or written work should be shared amongst you. And such an activity is considered academic dishonesty.

You are expected to attend at least 85% of the classes according to the university senate legislation. We will not allow you to sit for the final if you don't meet the criteria.

References

Textbook:

1. *Compilers: Principles, techniques and tools*, 2nd Edition (Aho, Ullman)
2. *Programming language design concepts*, John willy and sons (David A Watt)

Appendix 1C- Syllabuses per Stream

Software Engineering Stream

SEM VII

Industrial Internship Guideline

SEM VIII

Software Requirement Engineering

Program: Software Engineering and Computing Technology	
Module Title: System Modeling	Module Code: SITE-M4151
	Module ECTS: 10
Course Title: Software Requirement Engineering	Course Code:SITE -4151
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

Software requirements engineering is the process of determining what is to be produced in a software system. It has the widely recognized goal of determining the needs for, and the intended external behavior, of a system design.

The four specific steps in software requirements engineering are: requirements elicitation, requirements analysis, requirements specification, and requirements validation. The importance of high quality software requirements, long recognized as the foundation from which all software work follows, cannot be overstated. Arriving at a shared vision of the product to be developed is one of the greatest challenges facing the software project team, and customer involvement is among the most critical factors in software quality. This course will give participants a number of best practices, based upon the Software Engineering Body of Knowledge (SWEBOK), so that they can improve the quality of the requirements elicitation and development process in their organization. Characteristics of high quality requirements specifications are completeness, conciseness, accuracy, modularization, prioritization, analysis, and verification. Ultimately, the techniques presented reduce project risk, improve product quality, and allow for effective control of requirements volatility - all measures for increasing the likelihood of a successful software project.

Course Objectives

After completion of the course the student must be able to: Knowledge and understanding

- Explain why requirements engineering is a key to successful software engineering
- Describe the challenges involved in requirements engineering
- explain the importance of identifying stakeholders and their knowledge, context and goals, explain the difference between functional and quality requirements
- describe how to conduct bespoke (in-project/single-customer) requirements engineering in terms of common processes and techniques
- explain how market-driven differs from bespoke (in-project/single-customer) requirements engineering
- describe how requirements engineering in agile projects differ from traditional requirements engineering.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Skillfully elicit software requirements
- Clearly document software requirements according to industry standards and state of-the-art
- Priorities requirements
- Assure the quality of requirements and requirements specifications
- Assess current requirements engineering practices in a software project or a software development company

Prerequisites: None

Student Workload

Lecture	Lab	Tutorial	Home Study

32	**48**	**32**	

Course Content

Topic 1 Software Requirements Basics

- Introduction to requirement engineering (what, why, how to do)
- Definitions of a software requirement
- Product and process requirements
- Functional and non-functional requirements
- Emergent properties
- Quantifiable requirements
- System and software requirements
- Level and Importance of Requirement
- Project Failure Issues
    - Incomplete requirements
    - Lack of user involvement
    - Requirements defects
    - Requirements churn
    - Wasted resources
    - Gold plating
    - Inaccurate estimates
    - Requirements and the life cycle
- Standards and models related to requirements
    - SEI CMM® and CMMI SM
    - SEI CMM® and CMMI SM
    - Requirements management process area
    - Requirements development process area
    - Generic practices
    - IEEE software engineering standards
    - ISO 9000:2000 on requirements
- Requirements engineering process
    - Requirements engineering process
    - Incremental requirements development
    - Requirements engineering context

Topic 2 Requirements Elicitation

- Business level requirements
    - Purpose of business level requirements
    - Vision statement
    - Defining business objectives
    - Characteristics of "good" business objectives
    - Product scope and limitations
    - Product context diagram
- Stakeholders
    - Stakeholder defined
    - Benefits of identifying stakeholders
    - Identifying stakeholders and User types
    - Prune the stakeholder list
    - Participation strategy
    - Stakeholder conflict management

    - Decision criteria alternatives
    - Customer's bill of rights
    - Customer's bill of responsibilities
- Requirements elicitation techniques
    - Direct-two-way communications
    - Before the interview
    - Open-ended questions
    - Context-free questions
    - During the interview
    - Listening actively
    - Interviewing tips
    - After the interview
    - Focus groups
    - Facilitated requirements workshop
    - Benefits of facilitated requirements workshops
    - Documentation studies
    - Other requirements elicitation techniques

Topic 3 Requirements Analysis

- Requirements modeling
    - Requirements modeling
    - Benefits and Types of models
- Object oriented models
    - Use cases
    - Activity network
- Structured analysis models
    - Data flow diagram
    - Entity relationship diagram
    - State transition diagram
    - State transition table
- Other models
    - Process flow diagram
    - Decision tree
    - Event/response table
- Identifying product requirements
    - Levels of requirements information
    - Functional requirements and Nonfunctional requirements
    - Quality attributes
    - Usability requirements
- Prototyping
    - Prototype
    - Prototyping suggestions
    - Prioritizing requirements
    - Benefits of prioritizing requirements
    - Prioritization considerations

Topic 4 Requirements Specification

- Concept of operations
    - Documenting user level requirements
    - Concept of operations document
- Software requirements specification
    - System requirements
    - System vs. software requirements
    - Software requirements specification

- Writing "good" requirements
- Data dictionary
  - Data dictionary
  - Data dictionary notation

Topic 5 Requirements Validation

- Requirements review
  - Reviews defined
  - Cost of rework
  - Hold many peer reviews
  - Inspection process
  - Inspection meeting
  - Who should inspect requirements
  - Requirements completeness
  - Requirements checklist
  - Requirements checklist
  - Ambiguity
  - Peer review
- Requirements test planning
  - Testability
  - Writing test cases
  - Test matrix

Topic 6 Practical Considerations

- Iterative nature of the requirements process
- Change management
- Requirements attributes
- Requirements tracing
- Measuring requirements

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Topic 1		• Assignment
Week 2- Week 8	Topic 2,3		• Vision document • Project • Mid
Week 9- Week 15	Topic 4,5		• Paper Review
Week 16	Topic 6		• Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

Assignment	15%
Paper Review	15
Project	20%
Class Activity	5%
Mid	15%
Final	30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- System Requirements Engineering, P. Loucopoulos and V. Karakostas, McGraw-Hill: Requirements Engineering: From System Goals to UML Models to Software Specifications, Axel van Lamsweerde, John Wiley Sons
- *Managing Software Requirements: A Use Case Approach*, 2nd edition, Dean Leffingwell, Don Widrig, Addison Wesley: Boston
- *Software Requirements: Objects, Functions, & States*, A. M. Davis, Prentice Hall: Englewood Cliffs, 1993.
- *System and Software Requirements Engineering: Tutorial*, R. H. Thayer and M. Dortman (Editors),

IEEE Computer Society Press

- *Problem Frames: Analyzing and Structuring Software Development Problems*, M. Jackson, Addison-Wesley Professional; 1st edition (December 15, 2000)

- *Requirements Engineering: Processes and Techniques*, G. Kotonya and I. Sommerville, John Wiley Sons
- *Requirements Engineering - A Good Practice Guide*, I. Sommerville and P. Sawyer, Wiley
- *Scenarios, Stories, Use Cases Through the Systems Development Life-Cycle,* I. Alexander and N. Maiden (eds.), John Wiley & Sons, 2004.
- *Requirements Engineering: Frameworks for Understanding*, R. Wieringa, Wiley, 1997
- *Requirements Engineering: Social and Technical Issues*, J. Goguen, and M. Jirotka (Eds.), Academic Press, 1994.
- *Requirements Engineering*, L. Macaulay, Springer Verlag, 1996
- *User-Centered Requirements Analysis*, C. F. Martin, Prentice-Hall, 1994
- *Information System Requirements: Determination and Analysis*, D. Flynn, McGraw-Hill, 1992
- *Are Your Lights On?: How to Figure Out What the Problem Really Is,* D. C. Gause and G. M. Weinberg, Dorset House, 1990.
- *Exploring Requirements*, D. Gause and G. Weinberg, Dorset House, 1989
- *Managing Systems Requirements: Methods, Tools, and Cases*, S. J. Andriole, McGraw-Hill, 1996.
- *System Requirements Analysis*, J. O. Grady, McGraw Hill, 1993.
- *Systems Engineering: Coping with Complexity,* R. Stevens, K. Jackson, P. Brook, and S. Arnold, Prentice Hall 1998.
- *Requirements Engineering and Rapid Development: A Rigorous, Object-Oriented Approach*, I. S. Graham, Addison-Wesley, 1998.
- *Practical Software Requirements; A Manual Of Content And Style*, B. L. Kovitz, Manning Publications, 1998
- *User-Centered Requirements: The Scenario-Based Engineering Process*, K. L. McGraw and K. Harbison, Lawrence Erlbaum Associates,

1997.

- *The Complete Systems Analysis,* J. Robertson and S. Robertson, Dorset House, 1998.
- *Applying Use Cases: A Practical Guide,* G. Schneider and J. P. Winters, Addison-Wesley, 1998.
- Object-*Oriented Analysis and Design, with Applications*, G. Booch, Benjamin-Cummings, 1994
- *Object-Oriented Methods: A Foundation*, J. Martin and J. Odell, Prentice-Hall, 1995
- *The Unified Modeling Language Reference Manual*, J. Rumbaugh, I. Jacobson and G. Booch, Addison-Wesley, 1998
- *The Unified Modeling Language User Manual*, G. Booch, J. Rumbaugh and I. Jacobson, Addison-Wesley, 1998.

Department: Software Engineering	
Module Title: System Modeling	Module Code: SITE-M4151
	Module ECTS: 10
Course Title: *Software Architecture and Design*	Course Code: SITE -4152
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

At the heart of every well-engineered software system is its software architecture. Software architecture deals with the high-level building blocks that represent an underlying software system. These building blocks are the *components* (units of computation in a system), the *connectors* (models of the interactions between software components), and the *configurations* (arrangements of software components and connectors, and the rules that guide their composition). Software architectures that are found particularly useful for families of systems are often codified into *architectural styles.* The course will add treatment of software architecture, its foundation, principles, and elements, including those described above. The Course will also introduce software architecture research, future trends and state-of-the-practice.

Course Objectives

After completion of the course the student must be able to: Knowledge and understanding

- Understand the importance and role of software architecture in software systems.
- Define system architecture
- Define architectural design patterns and styles
- Identify the architecture of systems, critique them, and learn from them
- Create architectures for new or improved systems
- Execute the role of a system architect
- Produce deliverables of the architect needed to define the architecture of a system

Learning Outcomes

The program provides a knowledge and understanding of the following:

- Understanding the role of software architecture in the software lifecycle
- Defining viewpoints
- Describe architectural styles
- Select and use appropriate architectural styles
- Describing an architecture using several views
- Evaluating an architecture using both review-based and scenario-based methods
- Applying architectural patterns
- Identifying concurrency problems and their solution
- Understanding the role of components and services
- Define architectural design decisions
- Evaluate the coming attractions in software architecture research and practice

Prerequisites:

Student Workload

Lecture:	**Lab:**	**Tutorial:**	**Home Study:**

Course Content

Topic 1: Introduction to the fundamentals of software architecture

- What is software architecture?
- Definitions of software architecture
- Architecture addresses non-functional requirements
- Architectures and technologies

Topic 2: Software architecture and quality requirements of a software system

- Quality attributes
- Performance
- Scalability
- Modifiability
- Security
- Availability
- Integration

Topic 3: Fundamental principles and guidelines for software architecture design and architectural styles

- **Architectural Design**

Choosing the architecture framework
Allocate Components

- **Architectural Styles**

General structure
Distributed systems
Interactive systems
Others

Topic 4: Architectural patterns and frameworks.

- Architectural patterns

Creational patterns
Behavioral patterns
Structural patterns

Topic 5: Methods, techniques, and tools for describing software architecture and documenting design rationale.

- What to document
- UML
- Architecture views
- Component diagrams

Topic 6: Software architecture design and evaluation processes.

- Architectural design decisions
- Validation
- Using scenarios
- Prototyping

Topic 7: Rationale and architectural knowledge management in software architecting.

- What is architectural knowledge
- Knowledge and knowledge management in software architecture
- Tools and techniques
- State-of-art in architecture knowledge management
- Justification

Topic 8: Approaches and tools for designing and evaluating software architectures for the state-of- the-art technologies

- cloud-computing
- service-orientation
- mobile computing

Topic 9: Future challenges and emerging trends in software architecture discipline.

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1			
Week 2- Week 8			

Week 9- Week 15			
Week 16			

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

Assignment	15%
Paper Review	15
Project	20%
Class Activity	5%
Mid	15%
Final	30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Software Architecture: Foundations, Theory, and Practice, Taylor et al., John Wiley, 2010.
- Gorton, I., Essential Software Architecture, 2nd edition (2011), Springer.
- Chen, L., Ali Babar, M., Nuseibeh, B., Characterizing Architecturally Significant Requirements, IEEE Software 30(2): 38-45 (2013)
- Jane Cleland-Huang, Adam Czauderna, Ed Keenan: A Persona-Based Approach for Exploring Architecturally Significant Requirements in Agile Projects. REFSQ 2013:18-33.
- Hofmeister, C., et al., A general model of software architecture design derived from five industrial approaches, Journal of Systems and Software (JSS), 2006.
- Fielding, R., Taylor, R., Principled Design of the Modern Web Architectures by Roy Fielding and Richard Taylor, ICSE, 2000.
- Singh, M., and Huhns, M., Service-Oriented Computing: Key Concepts and Principles, IEEE Internet Computing: 9(1): 75-81 (2005).

- Clements, P., Kazman, R. and Klein, M., Evaluating Software Architectures: Methods and Case Studies, 2002, Addison-Wesley.
- Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M., Pattern-Oriented Software Architecture: A System of Patterns, 1996, Wiley.
- Model Driven Architecture: A Perspective by Singh and Sood, 2009

Program: Software Engineering and Computing Technology	
Module Title: *"High Performance Computing"*	Module Code: **SITE-M4191** Module ECTS: 12
Course Tile: Fundamentals of Parallel Computing	Course Code: **SITE - 4191** Course ECTS: 5 Course Duration:16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course will cover many aspects of parallel systems development. Starting initially by looking at different types of computer architecture and hardware capabilities for parallel programs as well as software. It will also investigate different performance measures for parallel programs.

The course will discusses different parallel processing models include theoretical approaches (Flynn taxonomy), as well as practical aspects (programming for shared memory machines, message passing paradigm). Lectures present technologies and

related standards of parallel and concurrent computing: Pthreads, OpenMP, MPI. The subject of design and analysis (especially concerning correctness and efficiency) of concurrent and parallel programs is also covered. Students will also be able to implement some parallel algorithms using the technologies presented.

Course Objectives

By the end of the course students should be able to accomplish the following:

- Understand the need for parallel computing and different hardware and software used for parallel processing.
- Critically evaluate approaches to concurrent and parallel systems development
- Compare and contrast the use of different parallel systems techniques to solve particular problem types.
- Design and construct parallel algorithms to solve particular problem types
- Analyze the performance of parallel systems undertaking particular work types

Learning Outcomes

This course introduces critical methods and techniques related to parallel computing. Particularly, the course focuses on hardware, algorithm, and programming of parallel systems, providing students a complete picture to understand pervasive parallel computing.

Through this course, the students will have a deep understanding of how parallel systems are designed and what are the fundamental methods to program and analyze them.
By the end of this course, students will be able to:

- Explain how large□scale parallel systems are architectured and how massive parallelism are implemented in accelerator architectures; Write parallel programs for large□scale parallel systems, shared address space platforms, and heterogeneous platforms;
- Design efficient parallel algorithms and applications; Be conversant with performance analyze and modeling of parallel programs;

Prerequisites: Operating systems

Student Workload

Lecture:	**Tutorial**	**Home Study**	**Laboratory**

Course Content

Chapter 1: Introduction:

- **Introduction:** parallel, concurrent and distributed systems, the need for parallelism.
- **Parallel Hardware:** Flynn's taxonomy of computer architecture, Parallelism in sequential machines, Abstract model of parallel computer, Multiprocessor architecture, Pipelining, Array processors.
- **Parallel Software:** Overview, Operating system support, Types of operating systems, parallel programming models, Processes and threads, shared memory, distributed memory, hybrid systems, Software tools.
- **Performance:** Speedup and efficiency, Scalability, Amdahl's law, Gustafson-Barsis's law.
- **Parallel Programming languages:** Fortran90, nCUBE C, Occam, C-Linda

Chapter 2: Data Dependency Analysis:

- Types of dependencies loop and array dependences, Loop dependence analysis, Program transformations.

Chapter 3: Shared Memory Programming:

- General model of shared memory programming, Process model under UNIX. Shared memory programming using Pthreads. Shared memory programming using OpenMP

Chapter 4: Algorithms for Parallel Machines:

- Speedup, Complexity and cost, Histogram computation, Parallel reduction, Quadrature problem, Matrix multiplication, Parallel sorting algorithms, Solving linear systems, Probabilistic algorithms.

Chapter 5: Message Passing Programming:

- Introduction, Model, Interface, Introducing collective, benchmarking parallel performance. Message passing programming using MPI.

Chapter 6: Debugging Parallel Programs:

- Debugging techniques, debugging message passing parallel programs, Debugging shared memory parallel programs.

Assessment Method:

- Assignments = 15%
- Projects = 25%
- Midterm = 25%
- Final Exam = 35%

Course Policies

Academic Honesty: Academic dishonesty including cheating, fabrication and plagiarism will not be tolerated in this course. It is expected that all work handed in by the student is their own. You will be reported to concerned bodies automatically including the center's academic council. The center deals with such academic dishonesty severely.

While we expect you to work individually, that doesn't mean that you cannot share ideas with other students or us. But you can only share ideas. No code or written work should be shared amongst you. And such an activity is considered academic dishonesty.

You are expected to attend at least 85% of the classes according to the university senate legislation. We will not allow you to sit for the final if you don't meet the criteria.

References

1. *Advanced computer architecture and parallel processing,* John willy and sons inc publishing (Hesham El-Rewini Mostafa Abd-El-Barr)
2. *An introduction to parallel programming* (Peter pacheco)
3. *Parallel Programming: For Multicore and Cluster Systems*, Springer (Thomas Rauber · Gudula Ru¨nger)

Program: Software Engineering and Software Engineering	
Module Title: Advanced Full Stack Development	**Module Code: SITE-M5201**
	Module ECTS: 17
Course Tile: Advanced Computer Networks	**Course Code: SITE - 4201**
	Course ECTS: 7
	Course Duration: 16 Weeks

Instructor's Contact Information:	Course Information:
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course introduces the student to the world of internetworking. There are in-depth discussions about routing protocols and how quality of service is delivered in an IP network. Multi-Protocol Label Switching is an emerging standard and is covered an introductory topic. The course wraps up with a case study, which the students work in groups to discuss, design and present their solutions

Course Objectives

Upon completion of this course the student will be able to:

- Identify the components of an Internetwork
- Describe the functionality of internetworking components
- Articulate how the Internet Protocol is implemented in a network

- • Describe how routers and routing protocols operate

Learning Outcomes

Having successfully completed this course, students will be able to demonstrate knowledge and understanding of:

- Operation of Switched Networks
- A range of network architectures and protocols
- Emerging topics in computer networks
- Design and critically analyses networking protocols for a range of technologies and scenarios

Prerequisites: Fundamental of Computer Networking

Student Workload (hrs.)

Lecture:	Tutorial	Home Study	Laboratory
32	16	32	32

Course Content

Chapter 1: Introduction to Computer Networking

- The development of networking
- The architecture today
- Communication models
- OSI Overview
- TCP/IP Overview
- Communication protocol overview
- Bridging and Switching Overview
- Virtual Private Networks Overview
- LAN Protocol Review
- WAN Protocol Review

Chapter 2: LAN design concepts and configuration

- Layer 2 Switching Overview
- Multi-link Trunking
- Layer 3 Switching
- Packet-by-Packet vs. Cut Through
- VLAN
- STP, MSTP, RSTP
- Ether-cannel Technologies

Chapter 3: WAN design concepts and configurations: Routing protocols

- Define Routing and Need to Route
- Types of Routing and Types of Routes
- RIP , RIP v2, RIPing
- OSPF V2, OSPF v3
- EIGRP V2, EIGRP for IP v6
- IS-IS
- BGP

Chapter 4: Multi-Protocol Label Switching

- Overview • Routing vs. Switching
- Concepts of MPLS
- Applications of MPLS
- Label Distribution Protocols
- Traffic Extension for Routing
- Constraint-Based Routed LDP
- Resource Reservation Protocol –TE

- **Chapter 4: Network QoS and VolP**
-

- QOS use, design and implementation techniques
- Building Converged networks
- **Chapter 5: Emerging topics in computer networks**

Schedule – optional

Weeks	Topics and Sub Topics
1	Chapter 1
2 -5	Chapter 2
6	Mid exam
7 -11	Chapter 3
12- 13	Chapter 4
14-15	Chapter 5
16	Final exam

Assessment Method:

- Continuous assessments (20%)
- Course projects (30%)
- Mid exam (20%)
- Final exam (30 %)

Course Policies

Maximum an excused absence from the class only 3 days. Late submission for any course evaluation is not allowed.

References

1. W. Stallings, Foundations of Modern Networking: SDN, NFV, QoE, IOT, and Cloud, Addison-Wesley Professional, 2016.
1. M. Meyers, CompTIA Network+ Certification Exam N10-006, 6 ed., McGraw-Hill Education, Sydney, Australia, 2015.
2. T. Lammle, CCNA Routing and Switching Complete Deluxe Study Guide, 2nd ed., John Wiley & Sons, Indianapolis, Indiana, 2016
3. Routing and Switching Essentials Companion Guide, Cisco Systems, USA, 2015
4. J. S. Beasley, P. Nilkaew, *Practical Guide to Advanced Networking*, 3rd Edition, Pearson IT Certification, 2013

Program: Software Engineering and Computing Technology	
Module Title: Advanced Full Stack Development	**Module Code:** SITE-M5201
	Module ECTS: 17
Course Title: ***Enterprise Application development***	**Course Code:** SITE - 4203
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	**Academic Year: 2016**
Office:	**Semester: I/II**
Phone:	**Meeting day:**
Email:	**Meeting time:**
Office Hours:	**Meeting location:**

Course Description
This course will cover the issues in designing and engineering large enterprise software systems. Such systems are typically distributed and require increasingly complex inter-enterprise as well as intra-enterprise coordination. Technologies such as Web Services and cloud computing provide platforms for building such systems, and architectures such as service-oriented architecture, event driven architecture and REST are idioms for structuring such systems. This course will focus on analysis and design of enterprise software systems, with particular emphasis on the architectures recommended for such systems. Some background in Web Services is also covered. The course is largely vendor-neutral, with short reviews in one or two places of technological solutions
Course Objectives
The objective of this course is to make engineering students familiar with design and

implementation of scalable enterprise applications and introduce problem solving design patterns. It gives idea about development of service oriented solutions & design and implantation of Rich Internet Applications

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- The process for developing large scale enterprise applications
- Build MVC Applications
- Build rich internet applications
- Talking enterprise level problems
- How to hone their critical thinking and presentation skills by developing the business plan and presenting their work to a professional audience

Prerequisites:

Fundamentals of Computer Science
Fundamentals of Database
Fundamentals of Networking

Course Content

1. Introduction
    1. Enterprise Applications trends and Challenges
    2. Application Architecture
    3. Multi-tier Architecture
    4. MVC Architecture
2. Database Concepts
    1. Database Design
    2. Enterprise Database (Oracle/DB2/MSSQL)
    3. Database Connectivity (JDBC/ODBC)
    4. Connection Pool
3. Service-Oriented Architecture
    1. SOA Concepts and principles
    2. XML/SOAP
    3. REST
    4. Web services
4. Platform for Enterprise Solutions: Java EE5: (9 hours)
    1. Java EE Platform Overview
    2. Web Core Technologies: Servlets and JSP

5. Enterprise Java Bean (6 hours)
    1. Enterprise JavaBean architecture
    2. Developing EJB3.0
    3. Session and message-driven EJBs
6. Advanced Web Technology (12 hours)
    1. Web2.0 Introduction and Concepts
    2. Rich Internet Application Development
    3. AJAX
    4. AJAX Frameworks(Prototype Library, DWR Java Ajax Framework)

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 - Week 6	**Topic 1 ,2**	**MVC Database connectivity**	**Lab Assignment**
Week 7 – Week 10	**Topic 3,4**	**Web service Servlet and JSP**	**Project 1 Mid Lab Assignment**
Week 11 – Week 14	**Topic 5 ,6**	**Java Beans Messaging RIA**	**Project 2 Lab Assignment Finale**

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

Project - 30 %
Lab - 20%
Mid - 20%
Final - 30%

Course Policies

- ❏ Attendance: It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- ❏ Assignments: No Late Assignment will be accepted
- ❏ Test/Quizzes: Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- ❏ Cheating/Plagiarism: No second Chance or excuses.

References

- Kevin Mukhar, Beginning Java EE 5, Apress,20
- Markl Grand, Patterns in Java, John Wiley & Sons,2003
- Dana Moore,Raymond Budd, Edward Benson, Professional Rich Internet Application, John Wiley & Sons, 2007.

SEM IX

Advanced Database Systems

Department: Software Engineering	
Module Title: **Advanced Full Stack Development**	Module Code: **SITE-M5201**
	Module ECTS:17
Course Title: *Advanced Database Systems*	Course Code:
	Course ECTS: 7
	Course Duration: 16 weeks

Instructor's Contact Information: Name: _____ Office: _____ Phone: _____ Email: _____ Office Hours:_____	**Course Information:** Academic Year: 2016 Semester: I/II Meeting day: Meeting time: Meeting location:

Course Description

This course addresses topics at the intersection of database system, operating system, and distributed computing research and development. It mainly focuses on different database systems, their architecture, and application areas. Specifically, Distributed and relational database systems, object database systems, and overview of graph database will be covered in this course.

Course Objectives

The objectives of this course are

- To introduce database distribution and ways to resolve complications related to distribution.
- To give students in depth information about system implementation techniques, data storage, representing data elements, database system architecture, the system catalog, query processing and optimization, transaction processing concepts, concurrency control techniques, database recovery techniques, database security and authorization, enhanced data models for advanced applications, temporal databases, deductive databases, database technology for decision support systems, distributed databases and client server architecture, advanced database concepts, and emerging technologies and applications.
- To provide an in depth skill on object databases
- To equip students with an indepth understanding of graph databases.

Learning Outcomes

The programme provides a knowledge and understanding of the following:

Practical Skills

Upon successful completion of this course, students will be able to:

- Implement and evaluate complex, scalable database systems, with emphasis on providing experimental evidence for design decisions.
- Apply and customize state-of-the-art implementation techniques for multi-node database management systems following modern coding practices.
- Interpret and comparatively criticize state-of-the-art research talks and papers, with emphasis on constructive improvements.
- Ability to compare and select systems to store and manipulate big data
- Apply database optimization techniques to resolve system performance issues in general
- Know how to effectively use and customize tools that may interact with database systems
- Apply object oriented database system concepts to design application layers
- Model graph and hypergraph based data for any domain of knowledge

Prerequisite/s: Fundamentals of Database Systems

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Laboratory:48 hrs**	**Home Study: 65 hrs**

Course Content

Part I: Introduction to Database Systems
Chapter 1: Object and Object Relational Database Systems
- Object Oriented Conceptual Model
  - Static Aspects of OOCM
    - Objects and Classes
    - Object Relationships
    - Complex Objects
  - Dynamic Aspects of OOCM
    - Encapsulation
    - Generic and User Defined Methods
    - Static and Dynamic Polymorphism
- Object Relational Approaches

Chapter 2: Overview NOSQL and Graph Database Systems

Part II: Distributed Database Systems
Chapter 3: Introduction to Distributed Database System
- Distributed Data Processing
- Distributed Systems
- Distributed Database System
- Data Delivery Alternatives
- Promises of Distributed Database System
- Problems of Distribution
- Distributed DBMS Issues
- Architectural Alternatives of Distributed DBMSs
  - Autonomy
  - Distribution
  - Heterogeneity
- Distributed Database architecture
  - Client/Server
  - Peer-to-Peer distributed systems
  - MDBS Architecture

Chapter 4: Distributed Database Design
- Distributed Database Design Problem
- Distributed Database Design Strategies
- Distributed Design issues
  - Fragmentation
  - Allocation
  - Information Requirements
  - Fragmentation
    - Primary Horizontal Fragmentation
      - Information Requirements
      - Completeness and Minimality of Predicates
    - Derived Horizontal Fragmentation
    - Vertical Fragmentation
    - Mixed/Hybrid Fragmentation
  - Allocation

Allocation Model

Chapter 5: Distributed Query Processing

Query Processing Components
Query Processing Problems and Objectives
Characteristics of Distributed Query Processors
Query Optimization issues
Layers of Query Processing
Query Decomposition
Data Localization
Global Query Optimization
Query Execution

Chapter 6: Distributed Transaction Management

Overview of Transaction
Termination Conditions of Transactions
Formalization of the Transaction Concept
Properties of Transactions
Characterization of Transactions
Types of transaction
Workflows Transaction Processing Issues
Distributed Query Execution Monitoring Architecture

Chapter 7: - Distributed Concurrency Control

Concurrency Control
Execution History/ Schedule
Formalization of History
Schedule Definition
Serial History
Serializable History
Serializability in Distributed DBMS
Global Non-Serializability
Concurrency Control Algorithms
Locking-Based Concurrency Control Algorithms
Two-Phase Locking (2PL) Strict, Centralized and Distributed 2PL
Timestamp-Based Concurrency Control Algorithms
Timestamp Ordering
Conservative Timestamp Ordering
Multiversion Timestamp Ordering
Optimistic Concurrency Control Algorithms
Deadlock Management
Local vs Global Wait for Graphs
Deadlock Prevention
Deadlock Avoidance
Wait-Die and Wound-Wait Algorithm
Deadlock Detection and Resolution
Centralized, Hierarchical, and Distributed
Non- Serializable Schedule and Nested Distributed

Chapter 8: Distributed Object Database Management

Object Distribution Design
Fragmentation Alternatives
Horizontal Class Partitioning
Vertical Class Partitioning
Path Partitioning
Object Server Architecture
Page Server Architecture
Cache Consistency
Alternative Cache Consistency Algorithms
Object Identifier Management
Object Migration
Distributed Object Storage
Object Query Processing
Transaction Management

Schedule

Weeks	**Topics and Subtopics**		
	Lecure	**Laboratory**	**Assessment**
Week 1- Week 2	Chapter 1	Postgres/Oracle Based Object Implementation	Assignment I (practical)
Week 3- Week 4	Chapter 2 and Chapter 3	Postgres and Oracle Based Object Implementation Neo4J based graph db implementation	Lab Exam I
Week 5 - Week 6	Chapter 3 and Chapter 4	Neo4J based graph db implementation	Assignment II (practical)
Week 7 - Week 8	Chapter 4	Postgres/Oracle - Database Partitioning	Assignment III (Seminar) and Mid Exam
Week 9 - Week 10	Chapter 5	Postgres/Oracle - Fragment Allocation	Project
Week 11 - Week 12	Chapter 6 and Chapter 7	Postgres/Oracle - Transaction Management	Assignment IV (HLL algorithm impl. assignment)
Week 13 - Week 14	Chapter 7 and Chapter 8	HLL - Concurrency control Algorithms	
Week 15 - Week 16		Course Project and Seminar	Final Exam and defenses

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, and classroom action research.

Assessment Method
The mark breakdown is as follows • Mid Exam 20% • Assignment I, II, III, IV and other assessments) 40% • Final Exam 40%
Course Policies
• **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course. • **Assignments:** No Late Assignment will be accepted • **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation. • **Cheating/Plagiarism:** No second Chance or excuses.
References
Text Book 1. Özsu, M.T., Valduriez, P.: Principles of Distributed Database Systems. Springer, New York (2011). https://doi.org/10.1007/978-1-4419-8834-8Google Scholar

Non Engineering Elective II

Software Project Management

Program: Software Engineering and Computing Technology	
Module Title: Project Management	Module Code: SITE-M5351
	Module ECTS:
Course Title: Software Project management	Course Code: SITE - 5351
	Course ECTS: 4
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
This course is mainly designed to prepare IT project managers, novice or experienced, with project management skills needed to better manage IT projects. Built along the IT project management lifecycle, this course covers detailed topics of the basic concepts of IT project management, including initiating, planning, controlling, executing, and closing projects. The course also shows how IT projects should be managed, from inception to post implementation review. The audience who take this course will likely improve their management skills and abilities to define the project scope, create a workable project plan, and manage within the budget and schedule.
Course Objectives
After completion of the subject the student will understands the problematics of design, planning and processing of project with software package Microsoft Project and acquires theoretical knowledge and practical skills and techniques from the field. From practical aspect is student able to develop a project (the allocation of work, material, cost resource and their optimization), process visual output of project. From advanced techniques is student able to work in server environment by processing

multiple projects simultaneously, as well as the basic techniques of analysis and economic interpretation of output project

Learning Outcomes

The program provides a knowledge and understanding of the following:

- Build a performing organization and project team
- Develop Work Breakdown Structures (WBS)
- Establish project estimates and project schedules
- Create project plans
- Manage overall change control
- Control project execution processes
- Terminate a project with a close-out strategy
- Build up the baseline knowledge for further career in IT project management fields.
- Understand the job roles of an IT project manager
- Recognize the key issues during the IT project management procedures
- Describe the best practices in IT project management processes

Prerequisites: None

Student Workload

Course Content

Topic I Overview

- **Software Development processes**
- Project Management Fundamentals
- Project Management Vs Portfolio
- Project Maturity
- Software project management tasks

Topic 2 Software Project Planning

- Software Project Management Plan

Topic 3 WBS and Schedule

- Work Breakdown Structure
- WBS Techniques
- Scheduling Overview
    - Scheduling Techniques (Network Diagrams, PERT, CPM, GERT, Bar Charts, Milestone Chart, and Gantt Chart)

Topic 4 Software Project Estimation

- Introduction
- Estimation Problem

- Estimation Approaches
    - Ad hoc approaches
    - Analogy approach
    - Expert judgment approaches
    - Empirical models

Topic 5 Risk and Change Management

- Software Risks

- Risk Management [identification , Analysis]
- Risk Prioritization and Control
- Risk Resolution and Monitoring

Topic 6 Change Control management

- Overview
- Control Activates
- Progress Monitoring and Report
- Project Metrics Earned Value Analysis

Topic 7 Software Quality Assurance

- Introduction
- Process and Product Quality
- QA Management
- Quality Control Tools and Techniques
- Quality Assurance and Standards

Topic 8 Team Management

- Motivation Theories
- Leadership and Teamwork
- Team Management
- Responsibility Assignment Matrix

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 ,2	Topic 1.2	Introduction to Ms Project Creating Task Resource Assignment with task	
Week 3,4,5,6	Topic 3,4	Project Plan Tracking Progress	SPM Mid Exam
Week 5,6,7,8	Topic 5	-	RMP
Week 9,10,11,12	Topic 6,7,8	Working On Collaborative Tools	SCMP SQAP Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

Assignment (SPMP, RMP, SCMP, and SQAP) – 50

Mid Exam – 20

Final Exam – 30

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Steve McConnell, Software Project – Survival Guide, Microsoft Press.
- Walker Royce, Software Project Management, Pearson.
- Robert K. Wysocki, Effective Software Project Management, Wiley.
- http://www.epmbook.com/
- http://w3-3.ibm.com/transform/project/
- http://www.pmi.org
- http://www.maxwideman.com/pmglossary/index.htm
- http://www.gantthead.com/article.cfm?ID=94587
- https://castlelearning.com/

Program: Software Engineering and Computing Technology	
Module Title: ***High Performance Computing***	**Module Code: SITE-M4191**
	Module ECTS: 12
Course Tile: Fundamentals of Distributed systems	**Course Code: SITE - 5192**
	Course ECTS: 7
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: Semester: Meeting day: Meeting time: Meeting location:

Course Description

Distributed systems aim at creating systems that are distributed over a network that can work autonomously, providing the required services as well as cooperating and collaborating to provide a general goal or service. Distributed systems appear to the user as a single, consistent and coherent system while in practice are distributed over the network providing or supporting all or some part of the services provided by the system.

This course covers the foundations of distributed systems including models of computing, communication, logical clocks and synchronization, distributed indexing, concurrency, consistency, failures, reliability and security. The course examines popular applications of distributed computing; analyze how the foundations manifest themselves in different ways depending on real-world constraints. Such applications include distributed file systems, peer-to-peer systems, distributed transactions, and web services.

Course Objectives

By the end of the course students should be able to accomplish the following:

- Understand the basic concepts of distributed systems and their difference with centralized systems.
- List the hardware and software environments of distributed systems and

understand the architectures of distributed systems.

- List different distributed systems and their characteristics.
- Understand how communication takes place in distributed systems, and implement some applications, which perform communication between two systems.
- Define code migration and its importance in distributed systems. Describe how code migration can be performed in distributed systems.
- Describe the role of software agents in distributed systems and list characteristics of different types of software agents.
- Describe how different entities of a distributed system can be identified and located using names, identifiers and addresses. Describe how name resolution is performed and how unreferenced entities can be removed.
- Understand and describe the importance of synchronization in distributed systems and how it can be implemented.
- Understand the importance of mutual exclusion and election are in distributed systems, understand different methods and algorithms for mutual exclusion and election and implement them using a computer program.
- Describe how distributed transactions are carried out.
- List how we can ensure the consistency of distributed systems. Describe consistency and distribution protocols.
- Describe different cases in which a fault can arise in distributed system and discuss how distributed systems can be made fault tolerant and how a distributed system can recover from faults.
- Describe distributed system security.

Learning Outcomes

The primary learning outcome of the course is two-fold:

1. Students will identify the core concepts of distributed systems: the way in which several machines orchestrate to correctly solve problems in an efficient, reliable and scalable way.
2. Students will examine how existing systems have applied the concepts of distributed systems in designing large systems, and will additionally apply these concepts to develop sample systems.

Prerequisites: Fundamentals of Computer Networks

Student Workload

Lecture:	**Tutorial**	**Home Study**	**Laboratory**

Course Content

Chapter 1: Introduction to Distributed System:

- **Introduction to Distributed systems**

- **Goals and distribution transparency**
- **Hardware concepts:** Multiprocessors, Homogeneous and heterogeneous Multi-computers
- **Software concepts:** Distributed and network Operating systems, Middleware
- **System architectures:** System architectures, Client server Model

Chapter 2: Communication:

- **Layered protocols**
- **Remote procedures call:** Basic RPC operations, Parameter passing, Extended RPC Models
- **Remote object invocation:** Distributed Objects, Binding, Dynamic vs. static RMI and Parameter passing.
- **Message-oriented communication:** Persistence and synchronicity in Communication, Message-oriented transient and persistent communication
- **Web Services:**

Chapter 3: Processes:

- **Threads, Clients and Servers**
- **Code Migration and Software agents:** Approaches, Migration and local resources, Software Agents

Chapter 4: Naming:

- **Naming entities and name resolution:** Names, Identifiers and Addresses, Name space and resolution.
- **Locating mobile entities:** Naming vs. locating entities, Solutions for locating entities
- **Removing un-referenced entities:** Problem of un-referenced and unreachable objects and the solutions

Chapter 5: Synchronization:

- **Clock synchronization:** Physical clocks, Clock synchronization algorithms, Logical clocks and global state
- **Election algorithms**
- **Mutual exclusion**
- **Distributed transactions:** Models, Classification and implementation of transactions, Concurrency control.

Chapter 6: Consistency and Replication:

- **Introduction**
- **Consistency Models**
- **Distribution protocols**
- **Consistency protocols**

Chapter 7: Fault Tolerance and Security:

- **Introduction to fault tolerance**
- **Reliable communication**

- **Distributed commit and recovery.**
- **Introduction to Security**

Schedule – optional

	Topics and Sub Topics		
Week	**Lecture**	**Lab**	**Assessment**
Week 1 - Week 4			
Week 5 - Week 8			
Week 9 - Week 12			
Week 13 - Week 16			

Assessment Method:

Assignments = 15%
Projects = 25%
Midterm = 25%
Final Exam = 35%

Course Policies

Academic Honesty: Academic dishonesty including cheating, fabrication and plagiarism will not be tolerated in this course. It is expected that all work handed in by the student is their own. You will be reported to concerned bodies automatically including the center's academic council. The center deals with such academic dishonesty severely.

While we expect you to work individually, that doesn't mean that you cannot share ideas with other students or us. But you can only share ideas. No code or written work should be shared amongst you. And such an activity is considered academic dishonesty.

You are expected to attend at least 85% of the classes according to the university senate legislation. We will not allow you to sit for the final if you don't meet the criteria.

References

Textbook:

- Distributed Systems: Concepts and Design, 5th Edition (Coulouris, Dollimore, Kindberg, Blair)
- Distributed Systems: Principles and Paradigms, 2nd Edition, (Tanenbaum, Van Steen)

Project I

SEM X

Technology Entrepreneurship

Ethics and Professionalism

Program: Software Engineering and Computing Technology	
Module Title: "Social and Professional Issues"	Module Code: SITE-M5171
	Module ECTS: 5
Course Title: Ethics and Professionalism	Course Code: SITE- 5171
	Course ECTS: 4
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
We can notice that the general public awareness on ethical aspects of technology is increasing. The high level of media attention given to computer related disasters in technical systems such as the explosion of Ariane V in 1996 and the Therac-25 computerized radiation machine overdoses has increased interest in Computer Ethics. The aim of this paper is to shed light upon the significance of social, ethical and professional issues in the field of Computing. It argues that ethics education should be incorporated into computing curricula. Engineering has a direct and vital impact on the quality of life of people. Accordingly, the services provided by engineers must be dedicated to the protection of the public

safety, health and welfare.

A code of professional ethics appears when an occupation organizes itself into a profession. It is central to advising individual professionals how to conduct themselves, to judging their conduct, and to understanding of a profession. Most engineering takes place within profit-making organizations that are embedded in a complex structure of societal and regulatory constraints. Engineering Ethics is therefore relevant for the majority of people within Computing. Research Ethics or Educational Ethics apply to those professionals in Computing who are active within those fields. Many other branches of Ethics may as well apply such as Healthcare Ethics and similar for some of Computing professionals.

Course Objectives

This course is intended to give students a chance to reflect on the humanitarian, social, and professional impact of computer technology by focusing on ethical issues faced by and brought about by computing professionals, including those related to networking and the internet, intellectual property, privacy, security, reliability, and liability. We will also focus on issues raised by the possible emergence in the future of highly intelligent machines.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Students will understand the contributions made by pioneers in mathematics and computer science.
- Students will understand the emerging issues related to ethics in cyberspace.
- Students will be aware of intellectual property rights, including: copyrights and patents.
- Students will understand issues associated with privacy.
- Students will be aware of methods and tools of analysis (ethical frameworks): Identify and evaluate ethical choices.
- Students will understand professional and ethical responsibilities, including

those defined in the ACM/IEEE Professional Code of Ethics.

- Recognize milestones in computing, networking, and information storage and retrieval
- Be familiar with the language and content of ethical discourse
- Understand modern debates surrounding intellectual property
- Appreciate the threats to privacy posed by modern information gathering techniques

Prerequisites: Fundamentals of Computer Systems

Student Workload

Lecture:	**Tutorial:**	**Home Study:**	**Laboratory:**

Course Content

Topic I Course overview

- Engineering as Social Experimentation
- What Is Ethics (*Normative ethics, Meta Ethics, Applied Ethics, Descriptive Ethics*)?
- *Ethics Contexts*
- Why Study Professional Ethics?
- Computer Ethics
- Ethical Decision Making

Topic II Ethics for IT Professionals and IT Users

- IT Workers as Professionals
- Relationships Between IT Professionals and Employers
- Legal Overview
- The Ethical Behavior of IT Professionals
- Professional Codes of Ethics
- Professional Organizations, Certification and Government Licensing
- Common Ethical Issues for IT Users

Topic III Computer and Internet Crime

- IT Security Incidents
- Software with Known Vulnerabilities
- Types of Attacks
- Classifying Perpetrators
- Detection and Prevention
- Response plan

Topic III Privacy

- Privacy Protection and The Law
- Key Privacy and Anonymity Issues
- Governmental Electronic Surveillance
- Recap of Data Encryption
- Identity theft (*Spyware, and* Consumer Profiling)
- Workplace Monitoring
- Spamming
- Advanced Surveillance Technology

Topic IV Key Issues in Software Development

- Software product liability
- Inexperienced/quality-ignorant software coding
- Software Quality Assurance
- Development of Safety-critical system
- Quality Management Standards

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Part I	-	Assignment 1
Week 2,3,4	Part II	-	Assignment 2 Paper Review Mid Exam
Week 5,6,7	Part III	-	Assignment 3 Paper Review
Week 8,9,10	Part IV	-	Assignment 2 Paper Review Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

Mid Exam – 20

Final Exam – 30

Assignment – 15

Paper Review - 25

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Case Studies in Information and Computer Ethics ,1st by Prentice Hall PTR Upper Saddle River, NJ, USA ©1996
- http://www.onlineethics.org/
- http://www.iep.utm.edu/ethics/

Program: Software Engineering and Computing Technology	
Module Title: Software Quality Assurance	Module Code: SITE-M4161
	Module ECTS: 5
Course Title: Software Testing, Verification and Quality Assurance	Course Code: SITE-M4161
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
This course stresses on uncovering the approaches and techniques to software Testing. Students will be introduced to various testing strategies, types and tools. The course mainly bases on foundations given on software engineering I. Students are required to undertake two software development projects as a way to exploit what they got in prerequisites. Moreover, the interim as well as the final deliverables of the aforementioned projects will be used as a playground for Unit Testing, Integration Testing, and System Testing.
Course Objectives
At the end of this course attendees will be able to: • Understand quality management processes

- Distinguish between the various activities of quality assurance, quality planning and quality control.
- Understand the importance of standards in the quality management process and their impact on the final product.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Upon completion of this course each student will be able to:

- Understand the processes, principles and techniques of software testing, software verification and validation
- Develop software testing plans and models
- Generate test cases according to various testing criteria
- Apply different types of testing approaches
- Develop and apply continuous verification methods
- Conduct test reviews and formalize documentation
- Efficiently use software verification and validation tools

Prerequisites: Fundamentals of Computer Systems

Student Workload

Lecture:	**Tutorial:**	**Home Study:**	**Laboratory:**

Course Content

Chapter One: Architectural Styles, Design Patterns, and Objects: Preliminaries

Architectural Design

Architectural Design Decisions

Architectural Views

Architectural Patterns

Layered Architecture

Repository architecture

Client–server architecture

Pipe and filter architecture

Application architectures

Object Oriented Design

Object - Oriented Design Using the UML

Design Patterns

Assignment 1.1: Software Requirement and Design Specification Document

Chapter Two – Software Verification and Validation

Quality Revolution

Software Quality

Distinguishing Verification and Validation
Limitations of Verification and Validation
The Role of V & V in Software Evolution
Types of Products and Objectives of V & V
TESTING CONCEPTS, ISSUES, AND TECHNIQUES
Purposes of Testing
Terms in Testing
Failure, Error, Fault, and Defect
Generic Testing Processes
Functional Vs. Structural Testing
Test-Case Design [4]
Concept of Complete Testing
When to Stop Testing
Notion of Software Reliability
Assignment II: Software Testing; Seminar I

Chapter Three - Theory of Program Testing
Basic Concepts In Testing Theory
Theory of Goodenough And Gerhart
Theory of Weyuker And Ostrand
Theory of Gourlay
Adequacy of Testing
Limitations of Testing
Chapter Four – Unit Testing
Concept of Unit Testing
Phases of Unit Testing
Static Unit Testing
Program Inspections,Walkthroughs, and Reviews [4]
Dynamic Unit Testing
Test-Case Design Techniques [4]
Mutation Testing
Debugging [4]
Debugging by Brute Force
Debugging by Induction
Debugging by Deduction
Debugging by Backtracking
Debugging by Testing
Extreme Testing
Assignment 4.1: Document Review
Assignment 4.2: Code Review

Chapter Five - Test Case Design Techniques

- Control Flow Testing
  - Control Flow graph
  - Paths and Path Selection Criteria
  - Generating Test Input
  - Infeasible Paths
- Data Flow testing
  - Dynamic Data Flow Testing
  - Dataflow Graph
  - Data Flow Testing Criteria
- Domain Testing
  - Testing for Domain Error
  - Sources and types of domain error
  - Test Selection Criterion

Assignment 5.1: Literature Review

Assignment 5.2: Software Test Plan

Chapter Six – System Integration Testing

- Concept of Integration Testing
- Interfaces and Interface Errors
- System Integration Techniques
- Software and Hardware Integration Testing
- Off-the-Shelf Component Integration Testing

Chapter Seven – System Testing

Basic Testing
Functionality Testing
Robustness Testing
Interoperability Testing
Performance Testing
Scalability Testing
Stress Testing
Load and Stability
Reliability
Regression Testing
Documentation
Regulatory Testing

Assignment 7.1: Final Project - Test Automation

Laboratory Content

Manual VNV-1: Software Validation Tools

Manual VNV-2: Unit Testing

- JUnit
- Website Testing - Selenium
- Website Testing - PHP Unit Testing Framework

Manual VNV-3: Integration Testing

- Arquillian

Manual VNV-4: System Testing

Manual VNV-5: Review of Version Control Tools

Git

Manual VNV-6: Continuous Integration Tools

- Jenkins
- Jenkins with Junit

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Part I	-	Assignment 1
Week 2,3,4	Part II	-	Assignment 2 Paper Review Mid Exam
Week 5,6,7	Part III	-	Assignment 3 Paper Review
Week 8,9,10	Part IV	-	Assignment 2 Paper Review Final Exam

Teaching Methods

The teaching methods that shall be employed for this course are lecturing,

demonstration, classroom discussion and classroom action research.

Assessment Method

Students will be evaluated through continuous assessment which includes

Assessments

Requirement Specification, System Design Document and Implementation 10%

Static Unit Testing (Requirement Validation) - 5%

Literature Review - 5%

Mid Exam - 15%

Practical Exam - 10%

Project - 25%

Software Test Plan

Implementing Built-in Tests

Final Exam - 30%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Jeff Tian, "Software Quality Engineering - Testing, Quality Assurance, and Quantifiable Improvement", Southern Methodist University - Department of Computer Science and Engineering, 2005.
- Ian Sommerville, "Software Engineering", Pearson Education, Ed. 9, 2011.
- Kshirasagar Naik and Priyadarshi Tripathy, "Software Testing and Quality Assurance - Theory and Practice", University of Waterloo, 2008.
- Glenford J. Myers, "The Art of Software Testing", John Wiley & Sons, Inc., 2nd Ed., 2004.

Selected Topics in Software Engineering

Project II

Information Technology Stream

SEM VII

Industry Internship Guideline

SEM VIII

Database Administration and Security

Program: Software Engineering and Computing Technology	
Module Title: System and Database Administration	Module Code: *SITE-M4241*
	Module ECTS:
Course Title: Database Administration and Security	Course Code:SITE - 4241
	Course ECTS:5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Database Architecture, Database Installation, Creating Database Using DBCA, Database Instances, ASM Instances, Network Environment, Storage Structures, User Security, Concurrency Control Mechanisms, Database Auditing and Maintenance, Performance Management, Backup and Recovery Concepts, Moving Data, Database Restart. The course provides a strong foundation in database security and auditing. This

course utilizes

Different scenarios and step-by□step examples. The course is covered: security, profiles, password policies, privileges and roles, Virtual Private Databases, and auditing. The course also covers advanced topics such as SQL injection, database management security issues such as securing the DBMS, enforcing access controls, and related issues.

Course Objectives

At the end of this course, students should be able to:

- Installing Database Software
- Creating an Database Using DBCA
- Managing Database instances and ASM instances
- Managing and controlling database network environment
- Define and devise transaction management, concurrency control, crash recovery components
- Managing storage structures
- Controlling user security
- Designing Database backup and recovery procedures
- Take Decisions related with Database Maintenance
- Understand the fundamentals of security, and how it relates to information systems
- Identify assets in your organization and their values
- Identify risks and vulnerabilities in operating systems from a database perspective
- Learn good password policies, and techniques to secure passwords in your organization
- Learn and implement administration policies for users
- Create policies, profiles and roles
- Understand the various database security models and their advantages or disadvantages
- Learn how to implement a Virtual Private Database using views, roles, and application context
- Gain an overview of auditing fundamentals, and create your own auditing model
- Learn the purpose and use of data dictionaries, encryption and SQL injection
- Explore an interesting research topic of your choice related to database security

Learning Outcomes

The Course provides a knowledge and understanding of the following:

- Design, model and install any database management systems by using Oracle database as sample.
- Plan, design, construct, control and manage database instances, database

network environment, storage structures, user security, database backup and recovery, database maintenance
- Define and devise transaction management, concurrency control, crash recovery components
- Examine and perform database administration roles and operations by using Oracle database system as a sample.
- Compare and contrast by examining the database systems and new trends in data storage, data retrieval and maintenance techniques.
- Understand the various database security models and their advantages or disadvantages
- Explain the various access control models and their advantages or disadvantages
- Understand the replication models used to increase availability and understand their advantages and disadvantages
- Understand common attacks used against database confidentiality and explain how to defend against the attack
- Use common integrity constraints to guarantee information correctness

Prerequisites:

Fundamental of Networking

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

Part I: Database Administration

- Exploring the Database Architecture
- Transactions
- Concurrency control
- Crash Recovery Components.
- Deadlocks
- Managing the Database instance
- Managing the ASM instance
- Configuring the Oracle Network environment
- Managing Database Storage Structures
- Administering User security
- Managing Data Concurrency
- Implementing Database Auditing
- Database Maintenance
- Performance Management
- Backup and Recovery Concepts
- Performing Database Backups
- Performing Database Recovery
- Moving Data

Part II: Database Security

- Database Access Controls

- Database Intrusion Detection
- Database Obscurity
- SQL Injection
- Database Security Testing
- Database Vulnerabilities, Threats & Physical Security
- Database Security Lifecycle
- Security Regulation

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 – week 2	• Exploring the Database Architecture • Transactions • Concurrency control • Crash Recovery Components. • Deadlocks • Managing the Database instance • Managing the ASM instance	• **Lab 1:** Installing Database Software • Database administrator (DBA) role and typical tasks and tools • Database software installation • Database Grid Infrastructure installation for a standalone server **Lab 2:** Creating a Database Using DBCA • Database software Installation Creating a Database using DBCA (Data Base Configuration Assistant) • Database creation scripts with the DBCA • Database design templates management with the DBCA • Additional tasks with the DBCA • **Lab 3:** • Understanding the COMMIT/ROLLBACK/LOCKING • **Lab4:**	Assignment #1. Lab#1, Project Title Approval Lab#2 Lab#3, Lab#4

		• Starting and stopping Database control (Start and stop the Oracle database and components) • Oracle Enterprise Manager • Database Home page • Using SQL*Plus • View Parameters • Database Startup and Shutdown credentials Viewing Alert Log	
Week 3 – week 4	• Configuring the Oracle Network environment • Managing Database Storage Structures • Administering User security • Managing Data Concurrency • Implementing Database Auditing • Database Maintenance •	• **LAB 5:** • Manage ASM users (Enterprise Manager) • Start/Stop ASM instance (SQL *Plus) • Start/Stop ASM instance (srvctl) • Start/Stop ASM instance (asmcmd) • Create and drop disk groups • Adding disk to disk groups • ASM Management Using Enterprise Manager • Disk Group Attributes • Retrieving ASM Meta data **Lab 6:** • Administering Users • Creating a user • Authenticating user Lock/unlock accounts, Reset password • Grand/revoke privileges • Create and manage roles • User profiles	Lab#5 Lab#6 Lab#7 Quiz #1 Project Proposal

		İmplementing password security features/create password profile • Password verification function • Assigning quotas to users **Lab 7:** • Detecting Lock conflicts • Resolving Lock conflicts • Managing Undo segments • Changing Undo Tablespace to fixed size • General Undo Information • Using Undo advisor • Viewing Systems Activity	
Week 5 – week 8	• Performance Management • Backup and Recovery Concepts • Performing Database Backups • Performing Database Recovery • Moving Data	**Lab 8:** • Listener control utility • Using srvctl to start stop the listener • Listener home page • Net services administration page • Creating listener • Database service registration • Configuring service aliases • Advanced connection options • Managing Database Storage Structures • Exploring storage structure • Creating new table space Storage for	Lab#8 Lab#9, Lab#10 Lab#11 Lab#11 Mid_exam Project Progress Report

		Table spaces • Altering / actions/drop / view table space & contents **Lab 9:** • Configuring Audit Trail • Specify audit Options • Default Auditing • Enterprise Manager Audit Page • Using and maintaining audit page/information • Value-Based Auditing • Proactive Maintenance • Viewing Alert History • Using Manager Optimizer Statistics Page • Enterprise Manager and AWR • Managing AWR • ADDM Findings/Recommendations • EM and Advisors • Automated Maintenance Tasks • Setting thresholds • Creating and testing Alert • Alert notifications **Lab 10:** • Performance Management • Backup & Recovery **Lab 11:** • Performing Backups • Performing DB Recovery	

		Lab 12: • Moving Data	
Week 9 – week 10	• Database Access Controls • Database Intrusion Detection	**Lab 13:** • Install Oracle 10g • Install MS SQL Server • Oracle Access Control • SQL Server Access Control • Oracle System Trigger • Oracle - Intrusion Detection through Table Triggers • MS SQL Server - Intrusion Detection through Table Triggers •	Lab#14 Assignments#2 Reading On: • Basta Chapter 1 • "2010-11 CSI/FBI Computer Crime and Security Survey" • Basta Chapter 2 • Basta Chapter 4 • Basta Chapter 9
Week 11 – week 12	• Database Obscurity	**Lab 15:** • Data Hiding using Views • Oracle Data Hiding using VPDs • SQL Server Data Hiding using VPDs •	Lab#15 Quiz#2 Reading • Database Security Literature Review • Basta Chapter 7 & 8
Week 13 – week 14	• SQL Injection • Database Security Testing • Database Vulnerabilities, Threats & Physical Security	**Lab 16:** • Oracle – DB Encryption • SQL Injection	Lab#16 Final Project Report Lab exam

Week 15 – week 16	Exam		Final Exam

Teaching Methods

- The format of the course will be lecture, class discussion and presentation, lab exercise, assignments and Project.

Assessment Method

- Quiz#1 – 5%
- Quiz#2 – 5%
- Assignments#1 – 5%
- Assignments#2 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. Physical Database Design, Lightstone/Teorey/Nadeau,MorganKaufman,2007, Publisher: ELSEVIER · Database Design and Implementation, Edward Sciore,Wiley,2008
2. Concurrency Control and Recovery in Database Systems, Bernstein/Hadzilacos/Goodman, 1987, Addison-Wesley
3. Fundamentals of Data Base Management Systems, Elmasri/Navathe, Pearson International Edition
4. Data Base Systems, Design, Implementation and Management , Rob, Coronel, Thomson Course Technology
5. Databases and Transaction Processing, Lewis, Bernstein, Kifer, Addison Wesley, 2001
6. Concepts of Data Base Management, Pratt/Adamski, Thomson Course Technology
7. Database Management Systems, Ramakrishnan/Gehrke, McGraw-Hill

8. Basta A, Zgola, M, DATABASE SECURITY, 1 st Edition, 2012, Course Technology Cengage Learning, Boston, MA, USA, ISBN-13: 978-1-4354-5390-6 ISBN-10:1-4354-5390-5
9. Oracle Technology Network1 : http://otn.oracle.com Complete list of Oracle 12C reference books http://www.oracle.com/pls/db112/homepage SQL Server Books on-line http://technet.microsoft.com/en-us//library/ms130214(SQL.105).aspx

E-business Strategy and Development

Program: Software Engineering and Computing Technology	
Module Title: **eBusiness**	Module Code: **SITE-M4251**
	Module ECTS:
Course Tile: *"E-business Strategy and Development"*	Course Code:**SITE-4251**
	Course ECTS: 4
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: Semester: Meeting day: Meeting time: Meeting location:

Course Description

Electronic business or e-business has been dramatically changing the way today's businesses operate and compete in the global marketplace. Nevertheless, most people including the businessmen are focusing on how to utilize the e-Business system as a tool to effectively implement traditional business strategies, few cares about the engineering construction of those systems. This course provides an overview of e-business engineering from an architectural perspective, and introduces the fundamental concepts and frameworks for e-business with which to examine the existing e-business models, the prevalent enabling technologies, and the strategic issues confronting the legacy firms.

Course Objectives

The objective of this course is to provide a foundation to prepare students, as future IT engineers, system architects, or project managers, to play leading roles in the application and management of e-business system construction. Upon successful completion of the course, students will be able to:

- Understand the current status of e-business concepts and follow its

development;

- Recognize the key implementation issues along with the e-business system design;
- Design the e-business engineering solution using the e-Business patterns;
- Describe the best practices in e-business engineering system development;
- Catch the enterprise data storage technology and management issues for e-Business systems;
- Evaluate the existing e-business system with an architectural vision and performance perspective;
- Appreciate the criticalness of performance, service-level-agreement, and latest IT technology to e-business solutions;

Build up baseline knowledge for further study and research in Electronic Business and Information Technology fields

Learning Outcomes

Knowledge and Understanding

Having successfully completed this module, you will be able to demonstrate knowledge and understanding of:
Describe the principles of strategic management
Describe systems architecture and technologies for systems integration
Subject Specific Intellectual and Research Skills
Having successfully completed this module you will be able to:
Define and quantify the benefits, costs, risks and time-scales associated with new strategic e business or IT initiatives
Compare and evaluate alternative e-business strategies and technologies
Justify and promote strategic initiatives, such as adopting a new e-business system or technology

Transferable and Generic Skills

Having successfully completed this module you will be able to:
Prepare and deliver senior management reports and presentations

Prerequisites:

Student Workload

Lecture:	**Tutorial**	**Home Study**	**Laboratory**

Course Content

Chapter 1:

- *Introduction to e-business and e-commerce*
  - *Introduction*
  - *Difference between E-commerce and E-business*
  - *E-business Opportunities*
  - *E-business risks and barriers*
  - *Management responses to e-commerce and e-business*

Chapter 2:

- *E-Commerce Fundamentals*
  - *Introduction*
  - *The e-commerce environment*
  - *Location of trading in the marketplace*
  - *Business models for e-commerce*
  - *Focus on auction business models*
  - *Focus on Internet startup companies*

Chapter 3:

- *E-Business Infrastructure*
  - *Introduction*
  - *Internet technology*
  - *Web technology*
  - *Internet-access software applications*
  - *How does it work? Internet Standards?*
  - *Managing e-business infrastructure*

Chapter 4:

- *Digital Marketing Channels for E-business*
  - *Introduction*
  - *Digital Marketing Channels*
  - *E-business Models: B2B and B2C*
  - *E-business Marketing Trends*

Chapter 5:

- *E-environment*
  - *Introduction*
  - *Social and Legal factors*
  - *Environmental and green issues related to internet usage*
  - *E-government*

Chapter 6:

- *E-Business Strategy*
  - *Introduction*
  - *Strategic Analysis*
  - *Strategic Objectives*
  - *Strategy definition*
  - *Strategy implementation*

Chapter 7:

- *Supply Chain Management for E-business*
  - *Introduction*
  - *What is supply chain management?*

- *Options for restructuring the supply chain?*
- *Using e-business to restructure the supply chain*
- *Supply chain management implementation*

Chapter 8: Selected Topics in E-Business

Assessment Method:

First exam: 30 marks
Reports, research projects, cases: 30 marks
Final exam: 40 marks

Course Policies

- **No make-up exam (except documented medical or family emergencies) will be offered nor will there be any changes made to the final exam schedule.**
- **It is your responsibility to submit assignments on or before due date. Assignments submitted after the due date will not be evaluated.**
- **Attendance is mandatory. Missing classes without valid reason may result in disqualification from the course.**
- **General principles of academic honesty include the concept of respect for the intellectual property of others, the expectation that individual work will be submitted unless otherwise allowed by an instructor, and the obligations both to protect one's own academic work from misuse by others as well as to avoid using another's work as one's own. All students are expected to understand and abide by these principles.**

References

1. **E-BUSINESS AND E-COMMERCE MANAGEMENT STRATEGY, IMPLEMENTATION AND PRACTICE, Dave Chaffey , 4th edition**
2. **Managing (E)Business Transformation, Farhoomand, A.**

Enterprise Systems and Network Administration

Program: Software Engineering and Computing Technology	
Module Title:Enterprise Systems Security	Module Code: SITE-M4211
	Module ECTS: 10
Course Title: Enterprise Systems and Network Administration	Course Code:SITE- 4211
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Computer information systems form the backbone of many large organizations, and many students will be called upon in their careers to help create, manage and maintain these large systems. This course will give students knowledge and experience working with enterprise level computer systems including workstation management, file servers, web servers, networking devices, configuration management, monitoring, and more. We will mainly focus on the GNU/Linux and Microsoft Windows server software, and much of the learning will take place in hands-on lab activities working directly with these systems. In addition, students will be responsible for developing some technical documentation and communicating information about their systems in a variety of ways. Finally, throughout the course students will be exposed to a variety of viewpoints and information directly from system administrators across AAU campuses.
Course Objectives
At the end of this course, students should be able to: • Understand the major components of an enterprise level computer network and

server system
- Design and implement a simple enterprise level server system and network, as well as provision workstations on that network quickly and easily
- Communicate and articulate information about enterprise systems clearly and effectively to users of all skill levels and interests
- Develop ways to increase efficiency by automating tasks whenever possible using scripting and configuration management tools
- Understand and describe security risks in any enterprise system and any ways that they can be mitigated
- Show how to monitor enterprise systems for problems and use that information to locate and fix any issues within the system

Learning Outcomes

The Course provides a knowledge and understanding of the following:

- Install, configure and manage enterprise systems/networks, including hardware/software.
- Implement and administer desktop and server operating systems (client/server), switching and routing devices.
- Implement and configure active directory
- Create user/group accounts and configure server roles.
- Administer permissions for users, files and network resources.
- Use group policies to configure and secure the network.
- Manage desktops and server computers using remote access.
- Install and configure TCP/IP for network and Internet connectivity.
- Configuration Management using Puppet
- Creating Secure Workstations (CIS 225 overview/review)
- Setting up an Enterprise Directory Service & Single Sign On
- Enterprise File Sharing
- Web Servers administration
- Core Networking Services administration
- System Monitoring
- Backup Strategies
- Maintain and troubleshoot enterprise networks and system.

Prerequisites:

Fundamental of Networking

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

Chapter 1: Enterprise NAT Operation, Configuration and Troubleshooting for IPv4

- Explain how NAT provides IPv4 address scalability in a small to medium-sized enterprise network.
- Configure NAT services on the edge router to provide IPv4 address scalability in a small to medium-sized enterprise network

- Troubleshoot NAT issues in a small to medium-sized enterprise network.

Chapter 2: Enterprise Network Device Discovery, Management, and Maintenance

- Use discovery protocols to map a network topology.
- Configure NTP and Syslog in a small to medium-sized enterprise network
- Maintain router and switch configuration and IOS files.

Chapter 3: Enterprise LAN Redundancy

- Spanning Tree Concepts
- Varieties of Spanning Tree Protocols
- Spanning Tree Configuration
- First-Hop Redundancy Protocols

Chapter 4: Wireless LANs

- Wireless LAN Concepts
- Wireless LAN Operations
- Wireless LAN Security
- Wireless LAN Configuration

Chapter 5: Adjust and Troubleshoot Single and Multi-area OSPF

- Advanced Single-Area OSPF Implementations
- Troubleshooting Single-Area OSPF Implementations
- Multi area OSPF
- Configuring Multi area OSPF

Chapter 6: EIGRP Advanced Configurations and Troubleshooting

- Characteristics of EIGRP
- Configuring EIGRP for IPv4
- Operation of EIGRP
- Configuration of EIGRP for IPv6
- Advanced EIGRP Configurations
- Troubleshooting EIGRP

Chapter 7: Enterprise Hierarchical Network Design

- Enterprise Hierarchical Network Design
- Evolving Network Architectures

- Enterprise Network Architecture

Chapter 8: Connecting to Enterprise WAN

- WAN Technologies Overview
- Selecting a WAN Technology

Chapter 9: Enterprise Point-to-Point Connections

- Serial Point-to-Point Overview
- PPP Operation
- Configuring PPP
- Troubleshooting WAN Connectivity

Chapter 10 Frame Relay

- Introduction to Frame Relay
- Configuring Frame Relay
- Troubleshooting Connectivity

Chapter 11: Securing Site-to-Site Enterprise Connectivity

- Enterprise VPNs
- Site-to-Site GRE Tunnels
- Introducing IPsec
- Remote Access

Chapter 12: Monitoring Enterprise Network

- Syslog
- SNMP
- NetFlow

Chapter 13: Deploying and Managing Enterprise Server

- Installing Domain Controllers
- Managing Active Directory Domain Services Objects
- Automating AD DS Administration by Using
- Implementing IPv4, DHCP, DNS, IPv6, Local Storage, File and Print Services, Group Policy
- Increasing Security for Server Resources
- Configuring AppLocker and Windows Firewall
- Implementing Server Virtualization with Hyper-V

Chapter 14 Storage Administration

- Storage Area Networks
- Network Attached Storage
- Storage Virtualization
- Enterprise Backup and Restoration Issues

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 – week 2	Chapter 1	• Enterprise NAT Operation, Configuration and Troubleshooting for IPv4	Assignment #1. Lab#1, Project Title Approval
Week 3 – week 4	Chapter 2 &3	• Enterprise Network Device Discovery, Management, and Maintenance • Enterprise LAN Redundancy	Lab#2 Lab#3, Quiz #1 Project Proposal
Week 5 – week 8	Chapter 4,5,& 6	• Wireless LANs • Adjust and Troubleshoot Single and Multi-area OSPF • EIGRP Advanced Configurations and Troubleshooting	Lab#4 Lab#4, Lab#5 Lab#6 Mid_exam Project Progress Report
Week 9 – week 10	Chapter 7 & 8	• Enterprise Hierarchical Network Design • Connecting to Enterprise WAN	Lab#7 Lab#8,
Week 11 – week 12	Chapter 9, 10 & 11	• Enterprise Point-to-Point Connections • Frame Relay • Securing Site-to-Site Enterprise Connectivity	Lab#9 Lab#10, Lab#11 Project Progress Report Quiz #2
Week 13 – week	Chapter 12,	• Monitoring Enterprise Network	Lab#12 Lab#13 Lab#14

14	13 & 14	• Deploying and Managing Enterprise Server • Storage Administration	Final Project Report Lab exam
Week 15 – week 16	Exam		Final Exam

Teaching Methods

- The format of the course will be lecture, class discussion and presentation, lab exercise, assignments and Project.

Assessment Method

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. CCNA Routing and Switching Complete Study Guide, 2nd Edition, Todd Lammle, Oct 2016
2. LPI Linux Essentials Certification All-in-One Exam Guide, 3nd Edition, by Robb Tracy
3. Linux: The Complete Reference, Sixth Edition 6th Edition, by Richard Petersen
4. MCSA Windows Server 2012 R2 Complete Study Guide, by William Panek , Jan 27, 2016
5. Dinesh Chandra Verma, 'Principles of Computer Systems and Network Management

Program: Software Engineering and Computing Technology	
Module Title: Cloud Computing and IOT	Module Code: SITE-M5221
	Module ECTS: 8
Course Title: Internet of Things	Course Code: SITE - 4221
	Course ECTS: 4
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

The Internet of Things (IoT) is a course about the new paradigm of objects interacting with people, with information systems, and with other objects. The course will focus on creative thinking and on hands-on project development. The students will learn: – IoT concepts – IoT technologies – Creative thinking techniques – Co-creation techniques

The course will examine the security and ethical issues of the vast implementation of smart devices known as the Internet of Things (IoT). The IoT is an environment where smart devices sense, anticipate, and respond to our needs as we manage them remotely. These smart devices often act as the gateway between our digital and physical world. The IoT touches many aspects of life including transportation, health care, safety, environment, energy, and more. This course will examine and discuss IoT technology and market specific topics, relevant case studies of IoT security vulnerabilities and attacks, and mitigation controls. Students will assess the health, safety, privacy, and economic impacts of IoT security events.

Course Objectives

- Understand the impact of IoT technologies
- Be able to draw the big picture of IoT ecosystem

- Be able to identify the architecture of IoT systems
- Be able to describe the essential components of IoT
- Have the knowledge of the emerging technologies of IoT
- Be able to examine the security and privacy challenges of IoT
- Be able to find appropriate security/privacy solutions for IoT

Learning Outcomes

The Course provides a knowledge and understanding of the following:

At the end of this course, students should be able to:

- Explain in a concise manner how the general Internet as well as Internet of Things work.
- Understand constraints and opportunities of wireless and mobile networks for Internet of Things.
- Use basic measurement tools to determine the real-time performance of packet based networks.
- Analyses trade-offs in interconnected wireless embedded sensor networks
- Demonstrate knowledge and understanding of the security and ethical issues of the Internet of Things
- Conceptually identify vulnerabilities, including recent attacks, involving the Internet of Things
- Conceptually describe countermeasures for Internet of Things devices
- Analyze the societal impact of IoT security events
- Develop critical thinking skills
- Compare and contrast the threat environment based on industry and/or device type

Prerequisites:

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

Chapter 1: INTRODUCTION TO INTERNET OF THINGS (IOT)

- Internet of Things (IOT) : Background and Development
- Internet of Things (IOT) Applications in Different Domains

Chapter 2: CONCEPTS BEHIND THE INTERNET OF THINGS.

- The IoT paradigm
- Smart objects
- Bits and atoms
- Goal orientation
- Convergence of technologies

Chapter 3: INTERNET OF THINGS (IOT) ARCHITECTURE

- Technical Definition of Internet of Things (IOT)
- Basic Architecture of Internet of Things (IOT)

- Internet of Things (IOT) demonstration
- History of IoT, M2M – Machine to Machine, Web of Things, IoT protocols
- Applications:
    - Remote Monitoring & Sensing,
    - Remote Controlling,
    - Performance Analysis
- The Architecture
    - The Layering concepts ,
    - IoT Communication Pattern,
    - IoT protocol Architecture,
- The 6LoWPAN

Chapter 4: IoT APPLICATION DEVELOPMENT

- Learning fundamentals and programming on Arduino IDE
- Interfacing Sensors and Peripherals with Arduino
- Developing Internet of Things Prototypes
- Application Protocols
    - MQTT, REST/HTTP,CoAP, MySQL
- Back-end Application Designing
    - Apache for handling HTTP Requests,
    - PHP & MySQL for data processing,
    - MongoDB Object type Database,
    - HTML, CSS & jQuery for UI Designing,
    - JSON lib for data processing, Security & Privacy during development,
    - Application Development for mobile Platforms: Overview of Android / IOS App Development tools

Chapter 5: IoT SERVICES AND APPLICATIONS

- AMR (Automatic Meter Reading)

- Smart City
- Smart Home
- Industrial Control
- Smart Social Networks
- Big Data Analytics

Chapter 6: IoT SYSTEMS ARCHITECTURES

- IoT system components
    - IoT Devices
    - IoT Gateways
    - Cloud Access
    - Cloud Components
- Cross connectivity across IoT system components
    - Device to Gateway –Short Range Wireless
    - Cell Phone as Gateway
    - Dedicated Wireless Access Point
    - Gateway to Cloud- Long Range connectivity
        - Wired

        - Cellular
        - Satellite
        - WAN
    - Direct Device to Cloud connectivity

Chapter 7: IoT DEVICE POWER CONSTRAINTS

- Powered and Unpowered Sensors
- Power Harvesting
- Energy Storage Technologies

Chapter 8: IoT PHY Connectivity

- Background on digital communication techniques up-to-date: wireless, satellite and wired media, modulation, coding, antennas and MIMO
    - MSK (example 802.15.4g)
    - OFDM (example 802.11)
    - DSSS (example 802.15.4)
    - FHSS (Bluetooth)
- Architectures: cellular, star, mesh, ring
- Cellular communication through generations and applicability to IoT
- LTE Advanced for Machine Type Communications (LTE-MTC), an evolution of LTE communications for connected things by 3GPP.
- NarrowBand IoT, (NB-IOT), standardization effort by 3GPP for a LPWAN used in cellular networks,

Chapter 9: NEED OF INTERNET OF THINGS (IOT) SECURITY

- Requirements and Basic Properties
- Main Challenges
- Confidentiality
- Integrity
- Availability
- Non-Repudiation

Chapter 10: SECURITY CLASSIFICATION & ACCESS CONTROL

- Data Classification (Public and Private)
- Privacy Issues in IOT
- IOT Authentication and Authorization
- IOT Data Integrity

Chapter 11: CRYPTOLOGY

- Cipher
    - Symmetric Key Algorithms (AES and DNS)
- Attacks
    - Dictionary and Brute Force
    - Lookup Tables
    - Reverse Look Tables
    - Rainbow Tables
- Hashing
    - MD5, SHA256, SHA512, RipeMD, WI
    - Objectives of Data Mining
    - Key aspects of Data Mining

Chapter 12: ATTACK SURFACE AND THREAT ASSESSMENT

- Embedded Devices
  - UART, SPI, I2C, JTAG
- Attacks
  - Software and Cloud Components
  - Firmware of the devices
  - Web Application Dashboard
  - Mobile Application used to Control, Configure, and Monitor Devices

Chapter 13: IoT PROTOCOL BUILT-IN SECURITY FEATURES

- Transport Layer
  - SSL / TLS and DTLS
- Application Layer
  - MQTT
  - CoAP
  - XMPP
  - AMQP

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 – week 2	Chapter 1 & 2	• Introduction To Internet Of Things (IoT) • Concepts Behind The Internet Of Things.	Assignment #1. Lab#1, Project Title Approval
Week 3 – week 4	Chapter 3 & 4	• Internet Of Things (Iot) Architecture • IoT Application Development	Lab#2 Lab#3, Quiz #1 Project Proposal
Week 5 – week 8	Chapter 5, 6, 7 & 8	• IoT Services and Applications • IoT Systems Architectures • IoT Device Power Constraints • IoT PHY Connectivity	Lab#4 Lab#4, Lab#5 Lab#6 Mid_exam Project Progress Report

Week 9 – week 10	Chapter 9 & 10	• Need Of Internet Of Things (IoT) Security • Security Classification & Access Control	Lab#7 Lab#8,
Week 11 – week 12	Chapter 11 & 12	• Cryptology • Attack Surface And Threat Assessment	Lab#9 Lab#10, Lab#11 Project Progress Report Quiz #2
Week 13 – week 14	Chapter 13	• IoT Protocol Built-In Security Features	Lab#12 Lab#13 Lab#14 Final Project Report Lab exam
Week 15 – week 16	Exam		Final Exam

Teaching Methods

- The format of the course will be lecture, class discussion and presentation, lab exercise, assignments and Project.

Assessment Method

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. The Wireless Embedded Internet, Zach Shelby, Carsten Bormann, Wiley
2. Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems, Dr. Ovidiu Vermesan, Dr. Peter Friess, River Publishers

3. Interconnecting Smart Objects with IP: The Next Internet, Jean-Philippe Vasseur, Adam Dunkels, Morgan Kuffmann
4. The Internet of Things: From RFID to the Next-Generation Pervasive Networked LuYan, Yan Zhang, Laurence T. Yang, Huansheng Ning
5. Internet of Things (A Hands-on-Approach) , Vijay Madisetti , Arshdeep Bahga
6. Designing the Internet of Things , Adrian McEwen (Author), Hakim Cassimally
7. Practical Internet of Things Security (Kindle Edition) by Brian Russell, Drew Van Duren
8. Securing the Internet of Things Elsevier
9. Security and Privacy in Internet of Things (IoTs): Models, Algorithms,and Implementations
10. Daniel Minoli, "Building the Internet of Things with IPv6 and MIPv6:The Evolving World of M2M Communications",

Data Mining and Analytics

SEM IX

Fundamentals of Distributed Systems

Computer System Engineering : IT Stream	
Module Title: High Performance Computing	Module Code: **SITE-M4361**
	Module ECTS:
Course Tile: Fundamentals of Distributed Systems	Course Code: **SITE-4362**
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name: Office: Phone: Email: Office Hours:	Academic Year: Semester: Meeting day: Meeting time: Meeting location:

Course Description

Distributed systems aim at creating systems that are distributed over a network that can work autonomously, providing the required services as well as cooperating and collaborating to provide a general goal or service. Distributed systems appear to the user as a single, consistent and coherent system while in practice are distributed over the network providing or supporting all or some part of the services provided by the system.

This course covers the foundations of distributed systems including models of computing, communication, logical clocks and synchronization, distributed indexing, concurrency, consistency, failures, reliability and security. The course examines popular applications of distributed computing; analyze how the foundations manifest themselves in different ways depending on real-world constraints. Such applications include distributed file systems, peer-to-peer systems, distributed transactions, and web services.

Course Objectives

By the end of the course students should be able to accomplish the following:

- Understand the basic concepts of distributed systems and their difference with centralized systems.
- List the hardware and software environments of distributed systems and understand the architectures of distributed systems.
- List different distributed systems and their characteristics.
- Understand how communication takes place in distributed systems, and implement some applications, which perform communication between two systems.
- Define code migration and its importance in distributed systems. Describe how code migration can be performed in distributed systems.
- Describe the role of software agents in distributed systems and list characteristics of different types of software agents.
- Describe how different entities of a distributed system can be identified and located using names, identifiers and addresses. Describe how name resolution is performed and how unreferenced entities can be removed.
- Understand and describe the importance of synchronization in distributed systems and how it can be implemented.
- Understand the importance of mutual exclusion and election are in distributed systems, understand different methods and algorithms for mutual exclusion and election and implement them using a computer program.

- Describe how distributed transactions are carried out.
- List how we can ensure the consistency of distributed systems. Describe consistency and distribution protocols.
- Describe different cases in which a fault can arise in distributed system and discuss how distributed systems can be made fault tolerant and how a distributed system can recover from faults.
- Describe distributed system security.

Learning Outcomes

In Distributed Systems this course, you will learn a range of fundamental and applied techniques in distributed systems. The learning objectives for Distributed Systems are:

1. Apply knowledge of distributed systems techniques and methodologies.
2. Explain the design and development of distributed systems and distributed systems applications.
3. Use the application of fundamental Computer Science methods and algorithms in the development of distributed systems and distributed systems applications.
4. Discuss the design and testing of a large software system, and to be able to communicate that design to others.

Prerequisites: Computer Networks

Student Workload

Lecture:	**Tutorial**	**Home Study**	**Laboratory**

Course Content

Chapter 1: Introduction to Distributed System:

- **Introduction to Distributed systems**
- **Goals and distribution transparency**
- **Hardware concepts:** Multiprocessors, Homogeneous and heterogeneous Multi-computers
- **Software concepts:** Distributed and network Operating systems, Middleware
- **System architectures:** System architectures, Client server Model

Chapter 2: Communication:

- **Layered protocols**
- **Remote procedures call:** Basic RPC operations, Parameter passing, Extended RPC Models
- **Remote object invocation:** Distributed Objects, Binding, Dynamic vs. static RMI and Parameter passing.
- **Message-oriented communication:** Persistence and synchronicity in Communication, Message-oriented transient and persistent communication
- **Web Services:**

Chapter 3: Processes:

- **Threads, Clients and Servers**

- **Code Migration and Software agents:** Approaches, Migration and local resources, Software Agents

Chapter 4: Naming:

- **Naming entities and name resolution:** Names, Identifiers and Addresses, Namespace and resolution.
- **Locating mobile entities:** Naming vs. locating entities, Solutions for locating entities
- **Removing unreferenced entities:** Problem of un-referenced and unreachable objects and the solutions

Chapter 5: Synchronization:

- **Clock synchronization:** Physical clocks, Clock synchronization algorithms, Logical clocks and global state
- **Election algorithms**
- **Mutual exclusion**
- **Distributed transactions:** Models, Classification and implementation of transactions, Concurrency control.

Chapter 6: Consistency and Replication:

- **Introduction**
- **Consistency Models**
- **Distribution protocols**
- **Consistency protocols**

Chapter 7: Fault Tolerance and Security:

- **Introduction to fault tolerance**
- **Reliable communication**
- **Distributed commit and recovery.**
- **Introduction to Security**

Assessment Method:

- Assignments = 15%
- Projects = 25%
- Midterm = 25%
- Final Exam = 35%

Course Policies

Academic Honesty: Academic dishonesty including cheating, fabrication and plagiarism will not be tolerated in this course. It is expected that all work handed in by the student is their own. You will be reported to concerned bodies automatically including the center's academic council. The center deals with such academic dishonesty severely.

While we expect you to work individually, that doesn't mean that you cannot share ideas with other students or us. But you can only share ideas. No code or written work

should be shared amongst you. And such an activity is considered academic dishonesty.

You are expected to attend at least 85% of the classes according to the university senate legislation. We will not allow you to sit for the final if you don't meet the criteria.

References

Textbook:

- Distributed Systems: Concepts and Design, 5th Edition (Coulouris, Dollimore, Kindberg, Blair)
- Distributed Systems: Principles and Paradigms, 2nd Edition, (Tanenbaum, Van Steen)

Cloud Computing and Security

School: School of Information Technology and Engineering	
Module Title: Cloud Computing and IOT	Module Code: SITE-M5221
	Module ECTS: 10
Course Title: Cloud Computing and Security	Course Code: SITE-5222
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Cloud computing helps organizations realize cost savings and efficiencies without spending capital resources up front, while modernizing and expanding their IT capabilities. Cloud-based infrastructure is rapidly scalable, secure, and accessible over the Internet—you pay only for what you use. So, enterprises worldwide, big and small, are moving toward cloud-computing solutions for meeting their computing needs, including the use of Infrastructure as a Service (IaaS) and Platform as a Service (PaaS). We have also seen a fundamental shift from shrink-wrapped software to Software as a Service (SaaS) in data centers across the globe. Moreover, providers such as Amazon, Google, and Microsoft have opened their data centers to third parties by providing low-level services such as storage, computation, and bandwidth. This trend is creating the need for a new kind of enterprise architect, developer, QA, and operational professional—someone who understands and can effectively use cloud-computing technologies and solutions. The promise of significant cost savings and inherent flexibility of resources are an impetus for the adoption of cloud computing by many organizations. Cloud computing also introduces privacy and security risks that are not traditionally present in a siloed data center. This course focuses on these security concerns and countermeasures for a cloud environment.

An overview of cloud computing and virtualization, the critical technology underpinning cloud computing, provides the necessary background for these threats. Additional topics vary but may include access control, identity management, denial of service, account and service hijacking, secure APIs, malware, forensics, regulatory compliance, trustworthy computing, and secure computing in the cloud. This course follows a seminar-style format where students are expected to lead class discussions and write a publication-quality paper as part of a course project.

Course Objectives

- Compare and Contrast the various cloud service models, cloud delivery models, key cloud characteristics, roles and boundaries and important terminology
- Explain how virtualization technology has enabled cloud computing.
- Demonstrate how various cloud providers such as AWS, Google Compute and Microsoft Azure implement and offer IaaS, PaaS and SaaS services.
- Outline, Classify and categorize how infrastructure components (compute, storage, network, load balancers, DNS) are delivered as Infrastructure as a Service
- Develop, deploy, manage and scale applications running in platforms such as Java, Python Platform (PaaS).
- Consume, integrate and extend software services such as Productivity and business applications and other services like database services.
- Describe how cloud can be used to perform Big Data Analytics using distributed computing technologies like Hadoop.
- Cite cloud security concerns and illustrate how to build implement security across all layers.
- Explain the cloud automation concepts such as DevOps practices, Continuous Integration and Deployment and how it automates cloud deployment.
- Demonstrate implementing private cloud using OpenStack components and services.
- Identify key architectural principles while designing cloud applications including SLAs, pricing and preventing single-point-of-failures. related to cloud.
- Most importantly, hands on experience launching IaaS services such compute, storage and network services, PasS services, SaaS services, Security and Database services

using Amazon Web Services

- Compare and contrast the various cloud delivery and deployment models, particularly the security implications of each;
- Understand the basics of virtualization technology and current attacks against it;
- Distinguish between terms such as "cloud computing" and "big data" and explain their differences;
- Describe the mechanisms used to secure cloud computing platforms, including trustworthy computing, secure computation, and data security in cloud environments;
- Appreciate the challenges that cloud computing introduces for regulatory compliance and digital forensics.
- Students will gain an appreciation for ongoing research in the area of cloud computing security. Students will critically review research papers to articulate their contributions to the field and the limitations of that work.

Learning Outcomes

The Course provides a knowledge and understanding of the following:

- Develop and deploy cloud application using popular cloud platforms,
- Design and develop highly scalable cloud-based applications by creating and configuring virtual machines on the cloud and building private cloud.
- Explain and identify the techniques of big data analysis in cloud.
- Compare, contrast, and evaluate the key trade-offs between multiple approaches to cloud system design, and Identify appropriate design choices when solving real-world cloud computing problems.
- Write comprehensive case studies analyzing and contrasting different cloud computing solutions.
- Make recommendations on cloud computing solutions for an enterprise.
- articulate the main concepts, key technologies, strengths, and limitations of cloud computing and the possible applications for state-of-the-art cloud computing
- Identify the architecture and infrastructure of cloud computing, including SaaS, PaaS, IaaS, public cloud, private cloud, hybrid cloud, etc.
- Explain the core issues of cloud computing such as security, privacy, and interoperability.
- Choose the appropriate technologies, algorithms, and approaches for the related issues.
- Identify problems, and explain, analyze, and evaluate various cloud computing solutions.
- Provide the appropriate cloud computing solutions and recommendations according to the applications used.
- Attempt to generate new ideas and innovations in cloud computing.

- Collaboratively research and write a research paper, and present the research online.
- Effectively communicate course work in writing and oral presentation.

Prerequisites:

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48hrs**

Course Content

Part I: Cloud Computing

- Cloud Computing Overview
- DataCenter and Virtualization
- IaaS Compute
- IaaS Storage
- IaaS Networking
- Cloud Databases
- PaaS
- SaaS
- Big Data and Cloud
- Security
- DevOps and the Cloud
- Miscellaneous Topic in the Cloud

Part II: Cloud Security

- Fundamentals of Cloud Security
- Virtualization
- Trusthworthy computing
- Secure Computation
- Cloud Management
- Data Management
- Data Security
- Regulatory Compliance
- Forensics and Malware

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**

Week 1 – week 2	Part I: Cloud Computing	• Data Center and Virtualization • IaaS Compute • IaaS Storage • IaaS Networking	Assignment #1. Lab#1, Project Title Approval
Week 3 – week 4		• Cloud Databases • PaaS • SaaS • Big Data and Cloud	Lab#2 Lab#3, Quiz #1 Project Proposal
Week 5 – week 8		• DevOps and the Cloud • Miscellaneous Topic in the Cloud	Lab#4 Lab#4, Lab#5 Lab#6 Mid_exam Project Progress Report
Week 9 – week 10	Part II: Cloud Security	• Cloud Security • Virtualization • Trustworthy computing	Lab#7 Lab#8,
Week 11 – week 12		• Secure Computation • Cloud Management • Data Management	Lab#9 Lab#10, Lab#11 Project Progress Report Quiz #2
Week 13 – week 14		• Data Security • Regulatory Compliance • Forensics and Malware	Lab#12 Lab#13 Lab#14 Final Project Report Lab exam
Week 15 – week 16	Exam		Final Exam

Teaching Methods

- The format of the course will be lecture, class discussion and presentation, lab exercise, assignments and Project.

Assessment Method

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%

- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. "Cloud Computing: Concepts, Technology and Architecture" by Thomas Erl and Ricardo Puttini
2. Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, & IaaS)
3. "Cloud Computing Explained" by John Rhoton
4. "Cloud Computing for Programmers" by Daniele Casal
5. "Cloud Security: A Comprehensive Guide to Secure Cloud Computing" by Ronald L Krutz and Russell Dean Vines
6. "Cloud Computing: SaaS, PaaS, IaaS, Virtualization, Business Models, Mobile, Security and More" by Kris Jamsa
7. NIST Cloud Computing Security Reference Architecture: NiST SP 500-299
8. Principles of Security in Cloud Computing, Balamurugan S.,Charanyaa S.

Non Engineering Elective II

Enterprise Application Development

Project I,

SEM X

Ethical Hacking and Penetration Testing

Non Engineering Elective II

Project II

Selected Topics in Information Technology

Technology Entrepreneurship

Artificial Intelligence Stream

SEM VII

Numerical Optimization and ODE

Department: Software Engineering	
Module Title: Computational Analysis and Optimization	Module Code: SITE-M4281
	Module ECTS: 15
Course Title: Numerical Optimizations and ODE	Course Code: SITE-4281
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting Day:
Email:	Meeting Time:
Office Hours:	Meeting Location:

Course Description
This course studies fundamental concepts of optimization from two viewpoints: theory and algorithms. It will cover ways to formulate optimization problems (e.g. in the primal and dual domains), study feasibility, assess optimality conditions for unconstrained and constrained optimization, and describe convergence. Moreover, it will cover numerical methods for analyzing and solving linear programs (e.g. simplex), general smooth unconstrained problems (e.g. first-order and second-order methods), quadratic programs (e.g. linear least squares), general

smooth constrained problems (e.g. interior-point methods), as well as, a family of nonsmooth problems.

Course Objectives

The course gives insight into the mathematical formulation of optimization problems and deals with advanced methods and algorithms to solve these problems. The knowledge of the possibilities and shortcomings of these algorithms should lead to a better understanding of their applicability in solving concrete engineering problems. In the course, an overview of existing software for optimization will also be given, this software will be used in the practical exercise sessions. The student learns to select the appropriate solving methods and software for a wide range of optimization problems and learns to correctly interpret the results.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Be able to formulate problems in their fields of research as optimization problems by defining the underlying independent variables, the proper cost function, and the governing constraint functions.
- Be able to transform an optimization problem into its standard form as outlined in the course. understand how to assess and check the feasibility and optimality of a particular solution to a general constrained optimization problem.
- Be able to evaluate whether the cost function and the constraints are convex, thus defining a convex problem with strong guarantees on optimality and convergence.
- Be able to use the optimality conditions to search for a local or global solution from a starting point. be able to formulate the dual problem of some general optimization types and assess their duality gap using concepts of strong and weak duality. Understand the computational details behind the numerical methods discussed in class, when they apply, and what their convergence rates are.
- Be able to implement the numerical methods discussed in class and verify their theoretical properties in practice.
- Be able to apply the learned techniques and analysis tools to problems arising in their own research.
- Implement different optimization algorithms in machine learning, robotics and other topics in software engineering fields

Prerequisites:

Applied Mathematics II, Introduction to Computer Science

Student Workload

Lecture:32 hrs.	**Tutorial: 32hrs**	**Home Study:65 hrs.**	**Laboratory: 48hrs**

Course Content

Chapter 1 - Basic Concepts

- Formulating an Optimization Problem
- Mathematical Description
- Local and Global Optimality
- Existence of an Optimal Solution
- Level Sets and Gradients
- Convex Sets, Functions, and Problems
- First-order characterization of a convex function
- Second-order characterization of a convex function

Chapter 2 - Complexity Issues

- Algorithms and Complexity
- Average Running Time
- Randomized Algorithms
- Basics of Computational Complexity Theory
- Complexity of Local Optimization
- Optimal Methods for Nonlinear Optimization

Chapter 3 - Introduction to Linear Programming

- Formulating a Linear Programming Model
- Examples of LP Models
- Practical Implications of Using LP Models
- Solving Two-Variable LPs Graphically
- Classification of LPs

Chapter 4 - The Simplex Method for Linear Programming

- The Standard Form of LP
- The Simplex Method
- Geometry of the Simplex Method
- The Simplex Method for a General LP
- The Fundamental Theorem of LP
- The Revised Simplex Method
- Complexity of the Simplex Method

Chapter 5 - Duality and Sensitivity Analysis in Linear Programming

- Defining the Dual LP
- Correspondence between the Primal and Dual LP Types
- Complementary Slackness
- Economic Interpretation of the Dual LP
- Sensitivity Analysis

Chapter 6 - Unconstrained Optimization

- Optimality Conditions
- Optimization Problems with a Single Variable
- Algorithmic Strategies for Unconstrained Optimization
- Method of Steepest Descent

- Newton's Method
- Conjugate Direction Method
- Quasi-Newton Methods
- Rank-one correction formula
- Inexact Line Search

Chapter 7 - Constrained Optimization

- Optimality Conditions
- Duality
- Projected Gradient Methods
- Sequential Unconstrained Minimization

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Chapter 1		
Week 2	Chapter 2		
Week 4 – Week 5	Chapter 3		
Week 5 - 7	Chapter 4		
Week 8	Mid Exam		
Week 9 - 10	Chapter 5		
Week 11 - 13	Chapter 6		
Week 14 - 15	Chapter 7		
Week 16	Final Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Exams – 60%
- Labs – 15%
- Assignments – 25%

Course Policies
• **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course. • **Assignments:** No Late Assignment will be accepted • **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation. • **Cheating/Plagiarism:** No second Chance or excuses.
References
• Numerical Optimization *by Jorge Nocedal Stephen J. Wright* • Numerical methods and optimization - An Introduction, 2nd Edition *by Sergiy Butenko, Panos M. Pardalos*

Social Network Analysis

Department: Software Engineering	
Module Title: Computational Analysis and Optimization	Module Code: SITE-M4281
	Module ECTS: 14
Course Title: *Social Network Analysis*	Course Code: SITE - 4328
	Course ECTS: 4
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Social networks pervade our social and economic lives. They play a central role in the transmission of information about job opportunities and are critical to the trade of many goods and services. They are important in determining which products we buy, which languages we speak, how we vote, as well as whether or not we decide to become criminals, how much education we obtain, and our likelihood of succeeding professionally. The countless ways in which network structures affect our well-being make it critical to understand how social network structures impact behavior, which network structures are likely to emerge in a society, and why we organize ourselves as we do.
Course Objectives
This course provides an overview and synthesis of research on social and economic networks, drawing on studies by sociologists, economists, computer scientists, physicists, and mathematicians.

The course begins with some empirical background on social and economic networks,

and an overview of concepts used to describe and measure networks.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Prerequisites:

Probability and Statistics

Algorithm and Data Structure

Linear Algebra

Student Workload

Lecture:32 hrs.	**Tutorial: 32hrs**	**Home Study:65 hrs.**	**Laboratory: 48hrs**

Course Content

Chapter 1: Graph Theory and Social Networks

- Social networks and their impact
- Graph definition
- Paths and connectivity
- Distance

Chapter 2: Strong and Weak Ties

- Triadic Closure
- Strength of weak ties
- Tie Strength and Network structure
- Betweenness Measure and Graph partitioning

Chapter 3: Random Network models

- Erdos-Renyi random graphs
- Model insights and realism
- Poisson Random Networks
- Growing random Networks
- Preferential attachment and Power laws

Chapter 4: Network centrality

- Degree
- closeness, betweenness
- Directed and eigenvector centrality

Chapter 5: Community structure

- Community

- Cliques, k-cores
- Detecting communities

Chapter 6: Small world network models

- Small world experiments
- Clustering and motifs
- Small world models
- Strategic network formation and search

Chapter 7: Games and Networks

- Network Games
- Network structure and behavior
- Nash Equilibrium
- Linear quadratic game

Chapter 8: Information networks

- Structure of the web
- Information networks, Hypertext and associative memory
- Link analysis and web search
- The problem of ranking: PageRank

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- Programming Assignments 35%
- Lab Exercises 15%
- 2 Exams – 50%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Networks, Crowds and Markets
- Social and Economic Networks, Matthew O. Jackson

Machine Learning

Department: Software Engineering	
Module Title: *Adaptive Computation and Machine Learning*	Module Code: SITE-M4291
	Module ECTS: 33
Course Title: *Machine Learning*	Course Code: SITE - 4292
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Machine learning techniques enable us to automatically extract features from data so as to solve predictive tasks, such as speech recognition, object recognition, machine translation, question-answering, anomaly detection, medical diagnosis and prognosis, automatic algorithm configuration, personalization, robot control, time series forecasting, and much more. Learning systems adapt so that they can solve new tasks, related to previously encountered tasks, more efficiently. This course will introduce the field of Machine Learning, in particular focusing on the core concepts of supervised and unsupervised learning. In supervised learning we will discuss algorithms which are trained on input data labelled with a desired output, for instance an image of a face and the name of the person whose face it is, and learn a function mapping from the input to the output. Unsupervised learning

aims to discover latent structure in an input signal where no output labels are available, an example of which is grouping web-pages based on the topics they discuss. Students will learn the algorithms which underpin many popular Machine Learning techniques, as well as developing an understanding of the theoretical relationships between these algorithms. The practical's will concern the application of machine learning to a range of real-world problems.

Course Objectives

The course aims to provide an introduction to the basic principles, techniques, and applications of Machine Learning. Programming assignments are used to help clarify basic concepts.

The course covers the principles, design and implementation of learning programs that improve their performance on some set of tasks with experience. Upon successful completion of the course, students will have a broad understanding of machine learning algorithms and their use in data-driven knowledge discovery and program synthesis. Students will have designed and implemented several machine learning algorithms in Python. Students will also be able to identify, formulate and solve machine learning problems that arise in practical applications. Students will have a knowledge of the strengths and weaknesses of different machine learning algorithms (relative to the characteristics of the application domain) and be able to adapt or combine some of the key elements of existing machine learning algorithms to design new algorithms as needed. Students will have an understanding of the current state of the art in machine learning and be able to begin to conduct original research in machine learning.

Learning Outcomes

The program provides a knowledge and understanding of the following:

- Be able to formulate machine learning problems corresponding to different applications.
- Understand a range of machine learning algorithms along with their strengths and weaknesses.
- Understand the basic theory underlying machine learning.
- Be able to apply machine learning algorithms to solve problems of moderate complexity.
- Be able to read current research papers and understand the issues raised by current research.
- Apply machine learning algorithm for classification and problems in health-related problem domains like medicine, genetics, medical diagnosis

Prerequisites:			
Student Workload			
Lecture:32 hrs.	**Tutorial: 32hrs**	**Home Study:65 hrs.**	**Laboratory: 48hrs**

Course Content

1. Introduction to Machine Learning
2. Bayesian Decision Theory
3. Parametric Methods
4. Multivariate Methods
5. Dimensionality Reduction
6. Clustering
7. Nonparametric Methods
8. Decision Trees
9. Linear Discrimination
10. Multilayer Perceptron
11. Kernel Machines
12. Graphical Models
13. Hidden Markov Models
14. Bayesian Estimation
15. Combining Multiple Learners
16. Reinforcement Learning

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Topic 1		
Week 2	Topic 2	Lab 1	
Week 3	Topic 3		
Week 4	Topic 4 - 5	Lab 2	Assignment I
Week 5	Topic 6		
Week 6	Topic 7	Lab 3	
Week 7	Mid Exam		Assignment II
Week 8	Chapter 8	Lab 4	
Week 9	Chapter 9		
Week 10	Chapter 10	Lab 5	

Week 11	Chapter 11		Assignment III
Week 12	Chapter 12	Lab 6	
Week 13	Chapter 13		
Week 14	Chapter 14	Lab 7	Assignment IV
Week 15	Final Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 7 Labs – 10%
- 1 Final Project – 20%
- 2 Exams – 50%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Pattern Recognition and Machine Learning Book by Christopher Bishop
- Machine Learning: A Probabilistic Perspective Textbook by Kevin R Murphy

Department: Software Engineering	
Module Title**: High performance Computing**	Module Code: **SITE-M4361**
	Module ECTS: 10
Course Tile: Fundamentals of Parallel Computing	Course Code: **SITE - 4191**
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course will cover many aspects of parallel systems development. Starting initially by looking at different types of computer architecture and hardware capabilities for parallel programs as well as software. It will also investigate different performance measures for parallel programs.
The course will discusses different parallel processing models include theoretical approaches (Flynn taxonomy), as well as practical aspects (programming for shared memory machines, message passing paradigm). Lectures present technologies and related standards of parallel and concurrent computing: Pthreads, OpenMP, MPI. The subject of design and analysis (especially concerning correctness and efficiency) of concurrent and parallel programs is also covered. Students will also be able to implement some parallel algorithms using the technologies presented.

Course Objectives

By the end of the course students should be able to accomplish the following:

- Understand the need for parallel computing and different hardware and software used for parallel processing.
- Critically evaluate approaches to concurrent and parallel systems

development

- Compare and contrast the use of different parallel systems techniques to solve particular problem types.
- Design and construct parallel algorithms to solve particular problem types
- Analyze the performance of parallel systems undertaking particular work types

Learning Outcomes	
Practical Skills	**Transferable Skills**

Prerequisites: Operating systems

Student Workload

Lecture:	**Tutorial**	**Home Study**	**Laboratory**

Course Content

Chapter 1: Introduction:

- **Introduction:** parallel, concurrent and distributed systems, the need for parallelism.
- **Parallel Hardware:** Flynn's taxonomy of computer architecture, Parallelism in sequential machines, Abstract model of parallel computer, Multiprocessor architecture, Pipelining, Array processors.
- **Parallel Software:** Overview, Operating system support, Types of operating systems, parallel programming models, Processes and threads, shared memory, distributed memory, hybrid systems, Software tools.
- **Performance:** Speedup and efficiency, Scalability, Amdahl's law, Gustafson-Barsis's law.
- **Parallel Programming languages:** Fortran90, nCUBE C, Occam, C-Linda

Chapter 2: Data Dependency Analysis:

- Types of dependencies loop and array dependences, Loop dependence analysis, Program transformations.

Chapter 3: Shared Memory Programming:

- General model of shared memory programming, Process model under UNIX. Shared memory programming using Pthreads. Shared memory programming using OpenMP

Chapter 4: Algorithms for Parallel Machines:

- Speedup, Complexity and cost, Histogram computation, Parallel reduction, Quadrature problem, Matrix multiplication, Parallel sorting algorithms, Solving linear systems, Probabilistic algorithms.

Chapter 5: Message Passing Programming:

- Introduction, Model, Interface, Introducing collective, benchmarking parallel performance. Message passing programming using MPI.

Chapter 6: Debugging Parallel Programs:

- Debugging techniques, debugging message passing parallel programs, Debugging shared memory parallel programs.

Schedule – optional

Week	Topics and Sub Topics

Assessment Method:

- Assignments = 15%
- Projects = 25%
- Midterm = 25%
- Final Exam = 35%

Course Policies

Academic Honesty: Academic dishonesty including cheating, fabrication and plagiarism will not be tolerated in this course. It is expected that all work handed in by the student is their own. You will be reported to concerned bodies automatically including the center's academic council. The center deals with such academic dishonesty severely.

While we expect you to work individually, that doesn't mean that you cannot share ideas with other students or us. But you can only share ideas. No code or written work should be shared amongst you. And such an activity is considered academic dishonesty.

You are expected to attend at least 85% of the classes according to the university senate legislation. We will not allow you to sit for the final if you don't meet the criteria.

References

Textbook:

1. *A*dvanced computer architecture and parallel processing, John willy and sons inc publishing (Hesham El-Rewini Mostafa Abd-El-Barr)
2. An introduction to parallel programming (Peter pacheco)
3. Parallel Programming: For Multicore and Cluster Systems, Springer (Thomas Rauber · Gudula Ru¨nger)

SEM VIII

Refer to Internship Guideline

SEM IX

Non Engineering Elective II
Project - I

Deep Learning

Department: Software Engineering	
Module Title: Adaptive Computation and Machine Learning	Module Code: SITE-M4291
	Module ECTS: 33
Course Title: *Deep Learning*	Course Code: SITE - 5293
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
"Deep Learning" systems, typified by deep neural networks, are increasingly taking over all AI tasks, ranging from language understanding, and speech and image recognition, to machine translation, planning, and even game playing and autonomous driving. As a result, expertise in deep learning is fast changing from an esoteric desirable to a mandatory prerequisite in many advanced academic settings, and a large advantage in

the industrial job market.

Course Objectives

In this course we will learn about the basics of deep neural networks, and their applications to various AI tasks. By the end of the course, it is expected that students will have significant familiarity with the subject, and to be able to apply to them to a variety of tasks. They will also be positioned to understand much of the current literature on the topic and extend their knowledge through further study.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- To understand how deep learning algorithms, work and how to train them.
- To review recent state-of-the-art applications of deep learning to problems in computer vision and machine perception.
- To gain experience using deep learning to solve problems in computer vision and machine perception.
- Implement major deep learning algorithm from scratch
- Scientifically explain the training process, model complexity, regularization and many crucial topics in Deep Learning
- Apply deep learning in computer vision for Robotics
- Apply deep learning for image classification in health-related problems like cancer treatment, pathogen identification
- Apply deep learning on astronomical object classification when studying the object's properties

Prerequisites: Machine Learning

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study:65 hrs.**	**Laboratory: 48 hrs**

Course Content

Chapter 1 - Deep Feedforward Networks

Chapter 2 - Regularization for Deep Learning

Chapter 3 - Optimization for Training Deep Models

Chapter 4 - Convolutional Networks

Chapter 5 - Sequence Modeling: Recurrent and Recursive Networks

Chapter 6 - Practical Methodology

Chapter 7 - Applications

Chapter 8 - Linear Factors

Chapter 9 - Autoencoders

Chapter 10 - Representation Learning

Chapter 11 - Structured Probabilistic Models for Deep Learning

Chapter 12 - Monte Carlo Methods

Chapter 13 - Confronting the Partition Function

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Chapter 1		
Week 2	Chapter 2	Lab 1	
Week 3	Chapter 3		
Week 4	Chapter 4	Lab 2	Assignment I
Week 5	Chapter 5	Lab 3	
Week 6	Chapter 6	Lab 4	
Week 7	Chapter 7		Assignment II
Week 8	Mid Exam		
Week 9	Chapter 8	Lab 5	
Week 10	Chapter 9		
Week 11	Chapter 10	Lab 6	Assignment III
Week 12	Chapter 11	Lab 7	
Week 13	Chapter 12	Lab 8	
Week 14	Chapter 13		Assignment IV
Week 15	Chapter 14		
Week 16	Mid Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab

exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 3 Homework – 15%
- 3 Exams – 65%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Deep Learning, by *Ian Goodfellow, Yoshua Bengio, Aaron Courville Online book, 2017*
- Deep Learning with Python by *J. Brownlee*
- Deep Learning Step by Step with Python: A Very Gentle Introduction to Deep Neural Networks for Practical Data Science by *N. D. Lewis*

Computer System Engineering : IT Stream	
Module Title: Adaptive Computation and Machine Learning	Module Code: SITE-M4291
	Module ECTS: 33
Course Title: Data Mining and Analytics	Course Code: SITE-4291
	Course ECTS: 5
	Course Duration: 16 weeks

Course Description

The field of computer science is experiencing a transition from computation-intensive to data-intensive problems, wherein data is produced in massive amounts by large sensor networks, new data acquisition techniques, simulations, and social networks. Efficiently extracting, interpreting, and learning from very large datasets requires a new generation of scalable algorithms as well as new data management technologies. Further how knowledge is created, captured, represented, stored and reused so as to fully leverage the intellectual assets of a firm has a become organizations strategic focus.

In this course we explore key data analysis and management techniques, which is applied to massive datasets as the cornerstone that enables real-time decision making in distributed environments, business intelligence in the Web, and scientific discovery at large scale. In particular, we examine the map-reduce parallel computing paradigm and associated technologies such as distributed file systems, no-sql databases, and stream computing engines. Additionally, we review machine learning methods that make possible the efficient analysis of large volumes of data in near real time. Besides we go through the tools and techniques for knowledge acquisition, assessment,

evaluation, management, organization and dissemination for managing and tracking organizations Knowledge.

Course Objectives

At the end of this course, the student will become familiar with the fundamental concepts of Big Data management and analytics; will become competent in recognizing challenges faced by applications dealing with very large volumes of data as well as in proposing scalable solutions for them; and will be able to understand how Big Data impacts business intelligence, scientific discovery, and our day-to-day life. The course will also enable you to learn about the practice in the context of managing the design, development and operation of information technologies that can facilitate Knowledge management

Learning Outcomes

The program provides a knowledge and understanding of the following:

- Solve organizational knowledge discovery , management and knowledge modeling issues
- Solving the Big Data problems, Current challenges, and trends
- Applying big data Algorithms for Big Data analysis.
- Apply machine learning approaches to deal with large datasets
- Apply Neural Nets to extract patterns
- Apply Big data analytics like Map-Reduce, Hadoop, Spark..etc to handle big data
- Implementation of big Data Technologies for Big Data management.
- Students will understand what big data and knowledge management means in the current IT era
- Students will be capable of applying Big data analytics tools, technologies and approaches in in order to extract knowledge for organization owning big data
- Understand what constitutes organizational knowledge and how to acquire and model them

Prerequisites: Machine Learning

Student Workload

Lecture:32 hrs.	**Tutorial: 32hrs**	**Home Study:65 hrs.**	**Laboratory: 48hrs**

Course Content

Chapter 1: Introduction

- Meaning of Data Mining
- Essence of Data Mining

- Relationship between Data Mining, Data Base, Data Warehousing and On-line Analytical Processing
- Issues in Data Mining
- The KDD/DM Process Model; Prediction vs. Description modeling

Chapter 2: Data warehousing and OLAP Technology for data mining

- What is a data warehouse?
- A multidimensional data models
- data cube computation
- Data warehouse architecture
- Data warehouse implementation
- From data warehouse to data mining

Chapter 3: Data preprocessing

- Why preprocesses the data?
- Major Tasks in Data Preprocessing
- Data Exploration
- Data understanding
- Data cleaning and reduction
- Data Integration and Transformation
- Discretization and concept hierarchy generation

Chapter 4: Handling and Processing Big Data: The MapReduce paradigm & Hadoop and HDFS overview

- MapReduce
- Mining of Large Datasets
- Introduction to Hadoop
- Installation of Hadoop
- MapReduce Algorithms
- Big Data Analytics using R
- Introduction to Apache Spark With
- Analysis of Streaming and real time Data with Apache Spark 2.0

Chapter 5: Social media analysis

- Sentiment Analysis
- Social Network Analysis
- SNA Theory and Application

- Chapter 6: Big Data Analytics and Machine Learning
- Clustering and classification
- Neural Networks and Deep Learning for Large scale analysis
- Self-organizing map(SOM)
- large scale classification and pattern recognition problems Using Tensor Flow

Chapter 7: Mining association rules in large databases

- Overview of Pattern Discovery
- Pattern finding and association rules discovery techniques
- aPriori algorithm
- Pattern-Growth Approach
- Mining single-dimensional Boolean association rules from transactional databases

Chapter 8: Web Mining

- Web content mining
- Web structure mining
- Web usage mining

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 - 2	Chapter 1		
Week 3 - 5	Chapter 2	Lab 1	
Week 6 - 7	Chapter 3		
Week 8	Exam	Lab 2	Assignment I
Week 9 - 10	Topic 6		
Week 6	Topic 7	Lab 3	
Week 7	Mid Exam		Assignment II
Week 8	Chapter 8	Lab 4	
Week 9	Chapter 9		
Week 10	Chapter 10	Lab 5	
Week 11	Chapter 11		Assignment III
Week 12	Chapter 12	Lab 6	

Week 13	Chapter 13		
Week 14	Chapter 14	Lab 7	Assignment IV
Week 15	Final Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 7 Labs – 10%
- 1 Final Project – 20%
- 2 Exams – 50%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Pattern Recognition and Machine Learning Book by Christopher Bishop
- Machine Learning: A Probabilistic Perspective Textbook by Kevin R Murphy

Machine Intelligence

Department: Software Engineering	
Module Title: Perception and NLP	Module Code: SITE-M5261
	Module ECTS: 15
Course Title: Machine Intelligence	Course Code: SITE - 5261
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Probabilistic and machine learning techniques are now an essential part of building robots (or embedded systems) designed to operate in the real world. These systems must contend with uncertainty and adapt to changes in the environment by learning from experience. Uncertainty arises from many sources: the limitations inherent in modeling a complex environment, noise and perceptual limitations in sensor measurements, and the approximate nature of algorithmic solutions. Building intelligent machines also requires that they adapt to their environment. Few things are more frustrating than machines that repeat the same mistake over and over again. We will explore modern statistical learning techniques that are effective at modeling nonlinear systems and learning online (i.e. throughout the robots operation). We will see how the twin ideas of uncertainty and adaptation are closely tied in both theory and implementation
Course Objectives

By the end of this course, students will understand the basic concepts and theory governing the programming of robots that perform autonomous tasks such as navigation and manipulation

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Describe the different physical forms of robot architectures.
- Kinematically model simple manipulator and mobile robots.
- Mathematically describe a kinematic robot system.
- Analyze manipulation and navigation problems using knowledge of coordinate frames, kinematics, optimization, control, and uncertainty.
- Compute forward and inverse kinematics for a small serial kinematic chain.
- Consider trade-offs among position control, velocity control, and force control when solving a robot control problem.
- Perform stability analysis of a controller-robot system, and describe why it is important.
- Model uncertainty in robot processes.
- Program robotics algorithms related to kinematics, control, optimization, and uncertainty.

Prerequisites: Algorithm and Data Structure,Probability and Statistics

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study:65 hrs.**	**Laboratory: 48hrs**

Course Content

Chapter 1 – Introduction

- Introduction and Overview
- Robot Mechanisms
- Robot Operating System

Chapter 2 – Kinematics

- Rotations and Rotation Matrices
- Euler Angles, Axis-Angle, Quaternions
- Properties of Rotations
- Rigid Bodies and Homogeneous Transforms
- Kinematic Chains
- Mobile Robot Kinematics
- Inverse Kinematics
- Linear and Angular Velocities, Moving Coordinate Frames
- The Jacobian

Chapter 3 – Filtering

- Kalman Filters (EKF)
- Kalman Filters continued
- Particle Filters
- Unscented Kalman Filter (UKF)
- Discrete Bayes Filters
- Chapter 4 – Planning
- Occupancy grids
- EKF-SLAM
- GraphSLAM
- RGB-D SLAM

Chapter 4 – Planning and Control

- Markov Decision Processes (MDPs)
- POMDPs, Belief Road Maps
- POMDP slides, BRM slides
- Fast (re-)planning
- Multi-robot mapping
- Gaussian Processes
- Reinforcement Learning
- Inverse RL
- Object recognition

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1 – Week 2	Chapter 1	Lab 1	
Week 3 – Week 5	Chapter 2	Lab 2	Assignment I
Week 6	Mid Exam		
Week 7 – Week 10	Chapter 3	Lab 3	Assignment II
Week 11 – Week 15	Chapter 4	Lab 4	Assignment III
Week 16	Final Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab

exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 3 Assignments – 30%
- 4 Labs – 10%
- 1 Final Project – 20%
- 2 Exams – 40%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Mark W. Spong, Seth Hutchinson, and M. Vidyasagar, Robot Modeling and Control, Wiley, 2006.
- Kevin M. Lynch and Frank C. Park, Modern Robotics: Mechanics, Planning, and Control, Cambridge University Press, 2017.
- John J. Craig, Introduction to Robotics, Addison-Wesley Publishing, 1989.*
- Alonzo Kelly, Mobile Robotics: Mathematics, Models, and Methods, Cambridge University Press, 2013.*
- Sebastian Thrun, Wolfram Burgard, and Dieter Fox, Probabilistic Robotics, MIT Press, 2005.*
- Bruno Siciliano and Oussama Khatib, eds. Springer Handbook of Robotics, Springer, 2008.

Selected Topics in AI

SEM X

Computer Vision

Department: Software Engineering	
Module Title: *Perception and NLP*	Module Code: SITE-M5261
	Module ECTS: 15
Course Title: *Computer Vision*	Course Code: SITE - 5262
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: _____	Academic Year: 2016
Office: _____	Semester: I/II
Phone: _____	Meeting day:
Email: _____	Meeting time:
Office Hours:_____	Meeting location:

Course Description
This course provides a comprehensive introduction to computer vision. The areas that the course will cover are image processing, Statistical Methods, Object Recognition, Tracking and Optimization Functions and highlights recent research technologies in computer vision.
Course Objectives
To introduce students with practice and theory of computer vision To give basics of pattern recognition concepts with applications to computer vision To provide students with necessary theory and skills for analysis of digital images, and thereby to construct representations of physical objects and scenes, and to make useful decisions.

Prerequisites:

Applied Mathematics I

Linear Algebra

Student Workload

Lecture:32 hrs	Tutorial: 32 hrs	Home Study:65 hrs	Laboratory: 48hrs

Course Content

Part 1: Image Formation and DIP

- Computer Imaging Systems and Lenses
- Image formation and sensing
- Digital Image Processing
    - 1D Signal Processing
    - Image Analysis and Preprocessing
        - Point Operation
            - Arithmetic operations
            - complements
            - Histograms
            - Contrast Stretching
            - Histogram Equalization
        - Neighbourhood operation
            - Convolution
            - Image Spatial Frequencies
            - Filtering
                - Low Pass Filtering
                    - Mean Filter
                    - Median filter
                    - Gaussian Filter
                - High Pass Filtering
                    - Laplacian Filter
                    - Laplacian of Gaussian Filter
                    - Sobel Filter
        - Segmentation
        - Fourier Transforms
        - Morphological operations of image
            - Dilation
            - Erosion

Part 2: Statistical Methods

Principal Component Analysis

Feature Detection (Blobs and SIFT)

- Object Detection
- Camshift and Mean Shift based object Tracking
- Optical Flow
- Background Subtraction Techniques
- Face Detection using Haar cascades

Part 4: Object Recognition, Tracking and Optimization Functions

Image Classification

The data-driven approach

K-nearest neighbor

Linear classification I

Loss Functions and Optimization

Linear classification II

Higher-level representations, image features

Optimization, stochastic gradient descent

Introduction to Neural Networks

Backpropagation

Multi-layer Perceptrons

The neural viewpoint

Convolutional Neural Networks

History

Convolution and pooling

ConvNets outside vision

Training Neural Networks

Update rules, ensembles, data augmentation, transfer learning

Deep Learning Software

Caffe, Torch, Theano, TensorFlow, Keras, PyTorch, etc

CNN Architectures

AlexNet, VGG, GoogLeNet, ResNet, etc

Part 4: Researches in Computer Vision

- Article review / Survey Research / Basic and/or Applied Research

Schedule

Weeks	**Topics and Subtopics**

	Lecure	Laboratory	Assessment
Week 1- Week 2	Part I		
Week 3- Week 4	Part I		Assignment I (Practical)
Week 5- Week 6	Part II		Mid Exam
Week 7- Week 8	Part II		Assignment II
Week 9- Week 10	Part III		
Week 11- Week 12	Part III		Assignment III: Mini Project/ Research
Week 13- Week 14	Part III		
Week 15- Week 16	Research Week		Final Exam and Seminar

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Mid Exam 20%
- Assignments 40%
- Final Exam 40%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

Reinforcement Learning

Department: Software Engineering	
Module Title: Adaptive Computation and Machine Learning	Module Code: SITE-M4291
	Module ECTS: 33
Course Title: Reinforcement Learning	Course Code: SITE - 5294
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Reinforcement learning is a framework for modeling the an autonomous agent's interaction with an unknown world. The agent's objective is to learn the effects of it's actions, and modify its policy in order to maximize future reward. The study of Reinforcement learning emphasizes a learning approach to Artificial Intelligence. Unlike supervised learning, the agent is not explicitly told the correct answers (labels), rather an RL agent must learn only from reward and trial and error interaction with the world. This general framework has been used to optimize helicopter flight, schedule elevators, and achieve super-human level performance in many games (e.g., Backgammon, GO, and Atari). Ideas from reinforcement learning has also be used to explain learning in animals, and model dopamine activity in the human brain.
Course Objectives
This course provides an introduction to some of the foundational ideas on which modern reinforcement learning is built, including Markov decision processes, value functions, Monte

Carlo estimation, dynamic programming, temporal difference learning, eligibility traces, and function approximation. This course will develop an intuitive understanding of these concepts (taking the agent's perspective), while also focusing on the mathematical theory of reinforcement learning. Programming assignments and projects will require implementing and testing complete decision-making systems.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Solving System of Linear Equations
- Manipulation of Vectors and Matrices in Vector Spaces
- Decomposition of matrices using different techniques for different purposes
- Transformation of Vectors and Matrices to different basis Vector
- Application of Linear Algebra to different numerical optimization methods
- Implementation of Linear Algebra topics using a computer programming language
- Applying computer program to solve problems
- Students will have knowledge of Linear algebra that can be applied to wide variety of engineering fields
- Student will acquire practical knowledge of application number crunching for applications like Machine Learning, Computer Graphics

Prerequisites: Machine Learning

Student Workload

Lecture:32 hrs.	**Tutorial: 32h rs**	**Home Study:65 hrs.**	**Laboratory: 48 hrs**

Course Content

Topic 1. Introduction and course overview
Topic 2. Supervised learning and imitation
Topic 3. Reinforcement learning introduction
Topic 4. Policy gradients introduction
Topic 5. Neural networks review
Topic 6. Actor-critic introduction
Topic 7. Value functions introduction
Topic 8. Advanced Q-learning algorithms
Topic 9. Optimal control and planning
Topic 10. Learning dynamical systems from data
Topic 11. Learning policies by imitating optimal controllers
Topic 12. Advanced model learning and images
Topic 13. Connection between inference and control
Topic 14. Inverse reinforcement learning
Topic 15. Advanced policy gradients (Natural Gradient, Importance Sampling)
Topic 16. Exploration and transfer learning
Topic 17. Multi-task learning and transfer
Topic 18. Meta-learning and parallelism

Topic 19. Advanced imitation learning and open problems

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Topic 1		
Week 2	Topic 2 – Topic 3		Homework 1
Week 3	Topic 4 – Topic 5		
Week 4	Topic 6 – Topic 7		Homework 3
Week 5	Exam		
Week 6	Topic 8 – Topic 9		
Week 7	Topic 10 – Topic 11		Homework 3
Week 8	Topic 12 – Topic 13		
Week 9	Topic 14		
Week 10	Topic 15		
Week 11	Topic 16		Homework 4
Week 12	Topic 17		
Week 13	Topic 18		
Week 14	Topic 19		
Week 15	Exam		
Week 16			

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 3 Homework – 15%
- 3 Exams – 65%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Deep Learning by *Ian Goodfellow*
- Reinforcement Learning: An Introduction by *Sutton & Barto*
- Algorithms for Reinforcement Learning by *Szepesvari*
- Dynamic Programming and Optimal Control, Vols I and II, by *Bertsekas*
- Markov Decision Processes: Discrete Stochastic Dynamic Programming, by *Puterman*
- Approximate Dynamic Programming by *Powell*

Project II

Non Engineering Elective II

Technology Entrepreneurship

Cyber Security Engineering

SEM VII

Enterprise Systems and Network Security

Department:	
Module Title: Enterprise Systems Security	Module Code: SITE-M4211
	Module ECTS: 10
Course Title: Enterprise system and network security	Course Code: SITE - 4211
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
This course introduces a series of advanced and current topics in cyber security, many of which are especially relevant in modern enterprise and infrastructure settings. The basics of enterprise compliance frameworks are provided with introduction to NIST and PCI. Hybrid cloud architectures are shown to provide an opportunity to fix many of the security weaknesses in modern perimeter local area networks. Emerging security issues in blockchain, blinding algorithms, Internet of Things (IoT), and critical infrastructure protection are also described in the context of cyber risk. Mobile security and cloud security hyper-resilience approaches are also introduced. This course completes with some practical advice for learners on how to

plan careers in cyber security.

Course Objectives

After completion of the course the student must be able to know and understanding

- Information and risk models including confidentiality, integrity and availability (CIA)
- Threats, attacks and how to manage them
- Cybersecurity architecture and operations
- Secure systems and products
- Cybersecurity management

Learning Outcomes

The program provides a knowledge and understanding of the following:

- Application security measures
- How to identify operating system holes
- The important interplay of privacy and digital rights management
- Trends in malware, privacy and security for mobile devices
- Ways to prevent network attacks and gaps in security policy

Prerequisites: None

Student Workload

Lecture:	**Lab:**	**Tutorial:**	**Home:**

Course Content

Topic 1: Introduction

Topic 2: Security Awareness, Compliance, Assessments, and Risk

- Social Engineering
- Phishing
- Security Awareness
- Security Assessment and Audit
- Different Frameworks Overview(NIST)
- Challenges of Compliance versus Security

Topic 3: Hybrid Cloud Security

- Enterprise Shift to Hybrid Cloud
- Workload Micro-Segmentation
- Cloud Access Security Brokers
- Advanced Hybrid Cloud Security Architecture

- Security of Isolated Servers

Topic 4: Blockchain, Anonymity, and Critical Infrastructure Protection

- Hashing Algorithms
- Blockchain
- Cyber Attribution
- Onion Routing and Tor
- Critical Infrastructure (Requirements and protection methods)

Topic 5: Mobility Security and Deception

- Mobility Security
- IMSI Catching
- IoT Security
- Using Deception and Honey Pots for Security

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1			•
Week 2- Week 8			•
Week 9- Week 15			•
Week 16			•

Teaching Methods	**Teaching Methods**

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 3 Homework – 15%
- 3 Exams – 65%

Course Policies	**Course Policies**

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.

- **Cheating/Plagiarism:** No second Chance or excuses.

References

- *"Framework for Improving Critical Infrastructure Cybersecurity," Version 1.0, NIST, Feb 12, 2014.*
- *" Introduction to Cyber Security", Edward Amoroso and Matthew Amoroso*
- *"Practical Methods for Securing the Cloud," IEEE, Ed Amoroso, 2014*
- *"Common Threats and Vulnerabilities of Critical Infrastructures," Robles, Choi, Cho, Kim, Park, and Lee, Dongseo University, Korea.*
- *"Cyber Security Deception," Mohammed Almeshekah and Eugene Spafford, Springer.com*
- *"Anonymous Connections and Onion Routing," Michael Reed, Paul Syverson, and David Goldschlag, onion-router.net.*

Cryptographic Concepts and Applications *

School: School of Information Technology and Engineering	
Module Title: Cryptographic Concepts and Applications	Module Code: **SITE-M4321**
	Module ECTS: 5
Course Tile: *Cryptographic Concepts and Applications*	Course Code: **SITE - 4321**
	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course will provide practical knowledge on a wide range of cryptography mechanisms and will explore their relationship with today's modern communications and networks. It includes the fundamentals of cryptography, classic and modern encryption, decryption, public and private key structures, digital signature and secure hash functions.

Course Objectives

The course provides the student with the mathematical background and the theory of cryptographic mechanisms and an introduction to their application in networks and communications links. After finishing this course student will be able to

- explain the importance of Cryptography as the workhorse in cyber security.
- identify and compare the classical encryption techniques and exploit their vulnerabilities.
- design and employ block ciphers such as DES, 3DES, AES and measure their strengths based on their key size.
- relate the number theory into key generation for asymmetric cryptography and explain RSA.
- Compare symmetric and symmetric encryption mechanisms and select appropriate method for each application.
- explain the application of encryption in RSA and authentication methods, HASH, MAC, SHA, and digital signatures.

- implement Suite B algorithms, Data at rest security measure, EC and ID based Crypto techniques.

Learning Outcomes

In this course, you will progress toward the following Learning Outcomes:
An ability to apply knowledge of computing and mathematics appropriate to the program's student outcomes and to the discipline.
Students will use statistical concepts to model and interpret data.
Students will apply discrete mathematics concepts and algorithms.
An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs.
Students will implement a computer-based system, process, component, or program from a given specification.
An understanding of professional, ethical, legal, security and social issues and responsibilities.
Students will recognize and describe current issues in security.
An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.

Prerequisites:

Course Content

Chapter One: Overview of Security

- Threats, risks, consequences
- Sources of threats
- Attacks classification
- Preventive measures, remedial measures

Chapter Two: Cryptography for Security

- Historical perspective
- Confusion vs. diffusion
- Stream ciphers vs. block ciphers
- Keys and key management
- Key exchange (peer to peer, peer - keyserver - peer)
- Diffie Helman key sharing scheme
- Symmetric key cryptography vs asymmetric key cryptography
- Trapdoor functions

Chapter Three: Hash Digests

- Properties of cryptographic hash functions
- Merkle Damgard construction
- MD family
- SHAfamily
- Digital signatures
- SHA3

Chapter Four: GPG

- Overview of GPG
- Commands and CLI
- GPG trust model

- GUI – KGPG, Seahorse
- Frontends – Kleopatra, enigmail

Chapter Five: Block ciphers

- Block cipher principles
- Feistel networks
- S boxes and P boxes
- Block cipher modes of operation
- DES
- 3DES
- AES

Chapter Six: Elementary number Theory

- Prime numbers, Factoring
- Modular arithmetic
- Fermat's & Euler's theorems
- gcd, Euclid's algorithm
- Discrete logarithm problem

Chapter Seven: Public key encryption

- Public key cryptosystems
- RSA algorithm
- Elliptic Curve cryptography

Chapter Eight: Practical applications

- PKI, CA. X509 certificates
- SSL/TLS, HTTPS
- IPV6 and IPSEC
- Proxies and Firewalls

Chapter Nine: Misc. Techniques

- Encryption using non-cryptographic tools (vi, zip)
- Authentication principles and methods
- Passwords, two-factor authentication
- Steganography
- Hamming
- Chaffing and Winnowing

Chapter Ten: Management aspects

- System Administration policies
- Security audit
- Penetration testing and ethical hacking
- Mandatory Access control, Discretionary Access Control
- Monitoring and logging tools
- Legal aspects

"Content of Laboratory Manuals"

Session 1: Encryption, decryption, signing, verification using GPG and OpenSSL
Session 2: Key signing party. Creation of a web of trust
Session 3: Computation of hash digests (md5, sha1)
Session 4: Building a fortress

Session 5: Encryption of a whole drive/partition

Schedule – optional

Week	Topics and Sub Topics

Assessment Method:

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. William Stallings, Cryptography and network security, Pearson Education.
2. Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone , Hand-book of Applied Cryptography, CRC Press.
3. Margaret Cozzens, Steven J Miller, The mathematics of encryption, American Mathematical Society
4. Bruce Schneier Applied Cryptography, John Wiley and Sons Mark Stamp,
5. Information Security: Principles and Practice, John Wiley and Sons
6. Matt Bishop, Computer Security, Art and Science, Pearson Education

Systems and Network Security Administration

School: School of Information Technology and Engineering	
Module Title: **Security Analysis and Administration**	Module Code: **SITE - M4341**
	Module ECTS: 5
Course Tile: *System and Network Security*	Course Code: **SITE - 4341**

Administration	Course ECTS: 5
	Course Duration: 16 Weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course gives students a broad overview of the administration of computing systems with an emphasis on the security of the systems and networks. Students will study computer security law and ethics, the development of an organizational computer security program, computer security policy development, computer security planning, contingency planning, risk management, and the certification and accreditation of computing systems.

Course Objectives

By the end of the semester, you will be able to:
• Describe how risk relates to a system security policy
• Describe various risk analysis methodologies
• Describe different types of attacks and their characteristics
• Describe appropriate measures to be taken, should a system compromise occur
• Describe the DoD system certification and accreditation processes
• Describe your responsibilities related to the handling of information about vulnerabilities
• Describe how the type of legal dispute (civil, criminal, private) affects the evidence used to resolve it
• Identify the bad actors in cyberspace and conduct a threat assessment of those bas actors
• Identify significant vulnerabilities, risks, and points at which specific security technologies/methods should be employed,
after examining the a typically, complex system
• Evaluate, categorize, and recommend appropriate responses for risk 1) with respect to technology; 2) with respect to individuals, and 3) in the enterprise
• Compare the advantages and disadvantages of various risk assessment methodologies
• Select the optimal risk assessment methodology, based on needs, advantages, and disadvantages • Define certification and accreditation • List the applicable laws and policies related to cyber defense and describe the major components of each, pertaining to the storage and transmission of data • Demonstrate the most important ethical and legal issues with regard to computer security solutions • Enhance technical writing skills by analyzing case studies in writing • Enhance speaking skills by analyzing case studies in class

Prerequisites:	
Schedule – optional	
Week	**Topics and Sub Topics**

Assessment Method:

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

Machine Learning

Software Project Management

SEM VIII

Refer to Internship guideline

SEM IX

Non Engineering Elective II

Project I

Ethical Hacking and Penetration Testing

School: School of Information Technology and Engineering	
Module Title: *Security Engineering*	Module Code: **SITE-M5331**
	Module ECTS:
Course Tile: *Ethical Hacking and Penetration Testing*	Course Code: **SITE- 5331**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course addresses how to evaluate the security stance of an organization IT infrastructures with intention of improving it. The course shows cases the acts, techniques, and tools used by attackers to probe security flaws with the permission of the owners of the target system as security assurance mechanism for improving the target's security. With the philosophy that to prevent a thief think like a thief.

Course Objectives

This course teaches students the underlying principles and many of the techniques associated with the cybersecurity practice known as penetration testing or ethical hacking. This course enables the students to:

- Understand the basic of information security
- Identify the components of ethical hacking / penetration testing
- Compromise target systems and getting access to information
- Master the skill required to recon, scan, exploit a system/network
- Understand how to write a pen test report for different stakeholders
- Provide a security countermeasure for the flaws found during pen test
- Identify the techniques to maintain access to a compromised system

Prerequisites:

Student Workload

Lecture: 32hrs	**Tutorial: 32hrs**	**Home Study:65hrs**	**Laboratory:48hrs**

Course Content

Chapter One: Overview of Computer Network
Chapter Two: Overview of Cyber Security
Chapter Three: What is an Ethical Hacking?

- Introduction
- Terminologies (Asset, Threat, Vulnerability, Exploit, Risk …)
- Why Ethical Hacking? Objectives of Ethical Hacking
- Testing Methodologies (Phases of Ethical Hacking)
- Categories and Types of Ethical Hacking
- Ethical Hacking Process
- Determining Scope
- Rules of Engagement
- Ethical and Legal issues
- Report Preparation

Chapter Four: Intelligence Gathering and Reconnaissance

- Whois Lookups
- Website Searches (email, people, Culture, Metagoofile, SHODAN, maltego…)
- DNS Recon (Lookups, Dig…)
- Document Metadata Analysis (exiftool, strings)
- Search Engines Vuln-Findings (Google Hacking –GHDB)

Chapter Four: Scanning

- Introduction
- Packet Sniffing (Filtering traffic, MITM, Hijacking Session , ARP Attacks …)
- Network Sweeping
- Network Tracing
- Port Scanning
- OS Fingerprinting
- Version Scanning
- Vulnerability Assessment
- Enumeration
- Netcat for Ethical Hacking

Chapter Five: Exploitation and Post Exploitation

- Introduction
- Gaining Access
- Remote and Client Side Exploitations
- Metasploit Kung fu (shell/bind, reverse listening, meterpreter…)
- Post Exploitation (moving files, pilfering from target, pivoting …)
- Maintaining Access (Backdoors and Rootkits, Covering the tracks …)
- Password Attacks (Guessing, brute force, cracking, sniffing, pass-the-hash …)
- Social Engineering (phishing, Web attack vectors…)

Chapter Six: Recommendation

- Recommending Countermeasures

"Content of Laboratory Manuals" –optional

Session 1:

Schedule – optional

Week	Topics and Sub Topics

Assessment Method:

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. Hacking Exposed Network Security Secrets and Solutions Sixth-Edition, Stuart Mcclure, Joel Scambray George Kurtz, McGraw-Hill , 2009
2. Penetration Testing: A Hands-On Introduction to Hacking, Georgia Weidman, No Starch Press, 2014
3. Hands-On Ethical Hacking and Network Defense, Michael T. Simpson, Kent Backman, and James E. Corley, 2011
4. The Basics of Hacking and Penetration Testing Ethical Hacking and Penetration Testing Made Easy, Patrick Engebretson, 2011

Computer Crime and Digital Forensics

School: School of Information Technology and Engineering	
Module Title: Cyber Forensics	**Module Code: SITE - M5311**
	Module ECTS: 5
Course Tile: Computer Crime and Digital Forensics	**Course Code:SITE-5311**
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course is designed to give students the tools and techniques for investigating crime involving digital evidence. The Course is designed as an introductory course in computer forensics. Students will first understand the need for computer forensics. Students will learn best practices for general incidence response. The course will then focus on the tools and techniques to perform a full computer forensic investigation.

Course Objectives

The course covers both the principles and practice of digital forensics. Societal and legal impact of computer activity: computer crime, intellectual property, privacy issues, legal codes; risks, vulnerabilities, and countermeasures; methods and standards for extraction, preservation, and deposition of legal evidence in a court of law. This course provides hands-on experience in different computer forensics situations that are applicable to the real world. Students will learn different aspects of digital evidence: ways to uncover illegal or illicit activities left on disk and recovering files from intentionally damaged media with computer forensics tools and techniques.

Learning Outcomes

- Determine which network systems are alive (reconnaissance)
- Discover which operating system software is used on a network, which patches have not been updated, etc. (enumeration)
- Identify which TCP and UDP services are running, listening, or established on the network (port scan)
- Create IDS sensors and attach them to network (snort)
- Detect, identify, resolve and document network intrusions
- Detect privilege escalation, remote control, keyloggers, rootkits, etc.

- Discover rouge wireless access points, sniffers, or VPNs on network
- Monitor and evaluate audit logs and set administrative alerts
- Deploy, implement and test IDS security plan

Prerequisites:

Student Workload

Lecture: 32hrs	**Tutorial: 32hrs**	**Home Study:65hrs**	**Laboratory:48hrs**

Course Content

Chapter One: Introduction to Computer Forensics

- Introduction: the problem of computer crime
- Understanding the need for computer forensics
- Defining computer forensics

Chapter Two: Computer Hardware

- Understanding computer components
- Digital Media
- Hard disk basics

Chapter Three: Forensic Tools

- Forensic hardware
- Hardware write/blockers
- Hard drive acquisitions
- Processing the scene

Chapter Four: Files and File Systems

- Windows file systems
    - FAT32
    - NTFS
- Forensic file images

Chapter Five: Forensic Software

- Overview of different software packages
- EnCase Introduction

Chapter Six: Bookmarking and Searching

- Creating basic search queries
- Hexadecimal, and Binary
- ASCII
- Unicode

Chapter Seven: Searching Evidence

- GREP
- Understanding GREP
- Building Regular Expressions
- Creating GREP keywords
- Viewing and managing keywords and cases

Chapter Eight: Forensic Reports

- Creating a forensic report
- Proper report writing
- Explaining forensics to the uneducated

Chapter Nine: Email Analysis

- Viewing
- mail
- Webmail
- POP
- IMAP

Chapter Ten: File Signature Analysis

- File signatures
- File extensions
- Differences between
- Identifying differences

Chapter Eleven: Hash Analysis

- Understanding hash algorithms
- Hashing files
- Hash libraries

"Content of Laboratory Manuals" –optional

Session 1: Hard drive acquisition
Session 2: Case Preparation
Session 3: EnCase Introduction
Session 4: Searching evidence
Session 5: GREP Lab
Session 6: Email analysis lab
Session 7: Detecting File Manipulation
Session 8: Hash Analysis Lab

Schedule – optional

Week	**Topics and Sub Topics**		
Week 1 - Week 4			
Week 5 - Week 8			
Week 9 - Week 12			
Week 13 - Week 16			

Assessment Method:

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

1. Hacking Exposed: Computer Forensics, Second Edition. Davis, Philipp, and Cowen ISBN: 0071626778
2. Thomas K. Clancy, Cyber Crime and Digital Evidence: Materials and Cases, First Edition 2011, LexisNexis, ISBN: 9781422494080
3. Digital Forensics with Open Source Tools. Cory Altheide and Harlan Carvey, ISBN: 978-1 59749-586-8, Elsevier publication, April 2011
4. 2. Computer Forensics and Cyber Crime: An Introduction (3rd Edition) by Marjie T. Britz, 2013.
5. EnCase Computer Forensics., 2014
6. File System Forensic Analysis. By Brian Carrier. Addison-Wesley Professional, March 27, 2005.
7. Computer Forensics: Hard Disk and Operating Systems, EC Council, September 17, 2009
8. Network Forensics: Tracking Hackers Through Cyberspace, Sherri Davidoff, Jonathan Ham Prentice Hall, 2012

School: School of Information Technology and Engineering	
Module Title: **Enterprise Systems Security**	Module Code:**SITE-M4211**
	Module ECTS: 9
Course Tile: *Web Security*	Course Code: SITE - 4211
	Course ECTS: 4
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year:
Office:	Semester:
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description

This course is designed to give students an overview of the most common security vulnerabilities found in Web Applications. It will cover Web Application Security architecture and the most common vulnerabilities found in site hosted at web application and how to protect against them.

Course Objectives

The course covers both the principles and practice of web application security. This course enables students to understand:

- Web Application Data Security Overview
- The Web Server
- Common Threats
- Securing the web server
- TLS and SSL
- Patches and Security Updates
- Firewalls and Port forwarding

Learning Outcomes

Get hands-on experience on web programming
Critically audit web applications for security flaws.
Design and implement exploits for real security bugs.
Develop secure web applications.

Prerequisites:

Student Workload			
Lecture: 32hrs	**Tutorial: 32hrs**	**Home Study:65hrs**	**Laboratory:48hrs**

Course Content

Chapter One: Web Application Data Security Overview

- How Data is stored in a Web Application
- Types of Data that need to be secured
- Overview of common security practices

Chapter Two: The Web Server

- Overview of IIS
  - System Requirements
  - Common Security Threats
- Overview of Apache
  - System Requirements
  - Common Security Threats

Chapter Three: Common Threats

List of common threats/security issues in web applications (brief description and common scenarios only)

- Rootkits / Bootkits
- Cross site scripting
- Alternate data streams
- Hidden Object(Services/Files/Users)
- Web App backdoors (C99.php, etc)
- Vulnerability scan
- DDOS/Application attack
- Profiling OS/Application
- Spoofing Dns, SSL, etc
- Man in the middle attack
- Mirroring server to remote location
- Phishing
- Key logging
- SQL Injection
- MAC Spoofing
- Mail form exploits
- Defacement
- Xftping
- social engineering

Chapter Four: Securing the web server

- Securing IIS
  - Unused Services
  - Patching
- Securing Apache
  - Unused Services
  - Patching

Chapter Five: TLS/SSL

- Transport Layer Security Overview

- Secure Socket Layer Overview
- SSL Termination
- SSL in the hosted environment
- Load Balanced SSL
- Setting up SSL in a load balanced environment

Chapter Six: Patches and Security Updates

- Installing patches and security updates
- Recent patches and security updates for IIS/Apache

Chapter Seven: Firewalls and Port forwarding

- Firewall configuration for Web Application Hosting
- Forwarding necessary ports for Web Application Hosting

"Content of Laboratory Manuals" –optional

Session 1: Apply OWASP to a web security assessment
Session 2: Exploit known Web Vulnerabilities on a Live Web Server

Schedule – optional

Week	**Topics and Sub Topics**		
Week 1 -Week 4			
Week 5 -Week 8			
Week 10 -Week 12			
Week 13 -Week 16			

Assessment Method:

- Quiz#1 – 5%
- Quiz#1 – 5%
- Assignments#1 – 5%
- Assignments#1 – 5%
- Mid Exam – 15%
- Each Lab and Final Lab exam – 20%
- Final Project - 20%
- Final Exam – 25%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.
- **Project:** The project consists of report and presentation on a topic of your choice that is relevant to the course. It is group-based

References

9. Hacking Exposed: Computer Forensics, Second Edition. Davis, Philipp, and

Cowen ISBN: 0071626778

10. Harwood, M., Goncalves, M., and Pemble, M. (2010) Security Strategies in Web Applications and Social Networking (Information Systems Security & Assurance), Sudbury, MA. Jones & Bartlett Learning. ISBN: 9780763791957

Electives

Natural Language Processing

Department: Software Engineering	
Module Title: *Perception and NLP*	Module Code:SITE-M5261
	Module ECTS: 22
Course Title: *Natural Language Processing*	Course Code: SITE-5264
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name: _____	Academic Year: 2016
Office: _____	Semester: I/II
Phone: _____	Meeting day:
Email: _____	Meeting time:
Office Hours:_____	Meeting location:

Course Description
Natural Language Processing addresses fundamental questions at the intersection of human languages and computer science. How can computers acquire, comprehend and produce English? How can computational methods give us insight into observed human language phenomena? How can you get a job at Google? In this interdisciplinary introductory course, you will learn how computers can do useful things with human languages, such as translate from French into English, filter junk email,

extract social networks from the web, and find the main topics in the day's news. You will also learn about how computational methods can help linguists explain language phenomena, including automatic discovery of different word

senses and phrase structure. Over the past decade, natural language processing has been revolutionized by statistical and probabilistic methods; you will learn about robust approaches to parameter estimation and inference. Our work will include learning new methods, discussions, and hands-on laboratories.

Prerequisites:

Applied Mathematics I

Applied Mathematics II

Basic probability Theory

Linear Algebra

Student Workload

Lecture:32 hrs	**Tutorial: 32 hrs**	**Home Study:65 hrs**	**Laboratory: 48 hrs**

Course Content

NLP: Overview and Background

Morphological Analysis

Syntax and Parsing

Semantic Analysis

Disclosure and Pragmatics

Disambiguation

Applications of NLP

Related Fields

Schedule

Weeks	**Topics and Subtopics**		
	Lecure	**Laboratory**	**Assessment**
Week 1 - Week 2			
Week 3 - Week 4			Assignment I (Practical)

Week 5 - Week 6			Mid Exam
Week 7 - Week 8			Assignment II
Week 9 - Week 10			
Week 11 - Week 12			Assignment III: Mini Project/ Research
Week 13- Week 14	Part III		
Week 15- Week 16	Research Week		Final Exam and Seminar

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, classroom discussion and classroom action research.

Assessment Method

The mark breakdown is as follows

- Mid Exam 20%
- Assignments 40%
- Final 40%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

Computational Genomics

Department: Software Engineering	
Module Title: Adaptive Computation and Machine Learning	Module Code: SITE-M4291
	Module ECTS: 37
Course Title: *Computational Genomics*	Course Code: SITE - 5296
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Research in biology and medicine is undergoing a revolution due to the availability of high-throughput technology for probing various aspects of a cell at a genome-wide scale. The next-generation sequencing technology is allowing researchers to inexpensively generate a large volume of genome sequence data. In combination with various other high-throughput techniques for epigenome, transcriptome, and proteome, we have unprecedented opportunities to answer fundamental questions in cell biology and understand the disease processes with the goal of finding treatments in medicine. The challenge in this new genomic era is to develop computational methods for integrating different data types and extracting complex patterns accurately and efficiently from a large volume of data. This course will discuss computational issues arising from high-throughput techniques recently introduced in biology, and cover very

recent developments in computational genomics and population genetics, including genome structural variant discovery, association mapping, epigenome analysis, cancer genomics, and transcriptome analysis.

Course Objectives

This course aims to present some of the most basic and useful algorithms for sequence analysis, together with the minimal biological background necessary for a computer science student to appreciate their application to current genomics research. Sequence alignments, hidden Markov models, multiple alignment algorithms and heuristics such as Gibbs sampling, and the probabilistic interpretation of alignments will be covered. Applications of these tools to sequence analysis will be presented: comparing genomes of different species, gene finding, gene regulation, whole genome sequencing and assembly. Whenever possible, examples will be drawn from the most current developments in genomics research.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Acquire a fundamental understanding of the core algorithms, techniques and data structures used in the specialized area of computational biology and bioinformatics.
- Acquire an ability to formulate and/or model a biological problem into a well-defined computer science problem.
- Acquire an ability to design and analyze efficient algorithms and data representations for problems in computational biology and bioinformatics.
- Obtain an interdisciplinary vocabulary that encompasses basic and emerging research themes within bioinformatics and the life sciences.
- Obtain an ability to read/comprehend research literature in the emerging area of computational life sciences.

Transferable Skills

- Acquire an ability to apply and relate the algorithmic concepts/techniques and data structures used for solving computational biology problems into other scientific domains - particularly relating to text mining, pattern matching and string algorithmic.
-

Prerequisites:

Probability and Statistics

Algorithm and Data Structure

Student Workload

Lecture:32 hrs.	Tutorial: 32hrs	Home Study:65 hrs.	Laboratory: 48hrs

Course Content

- **Chapter 1 – Sequence statistics**
  - Genomic era, year zero
  - The anatomy of a genome
  - Probabilistic models of genome sequences
  - Annotating a genome: statistical sequence analysis
  - Finding data: GenBank, EMBL, and DDBJ
- **Chapter 2 - Gene finding**
  - The human genome sweepstakes
  - An introduction to genes and proteins
  - Genome annotation: gene finding
  - Detecting spurious signals: hypothesis testing
- **Chapter 3 - Sequence alignment**
  - Eye of the tiger
  - On sequence alignment
  - On sequence similarity
  - Sequence alignment: global and local
  - Statistical analysis of alignments
  - BLAST: fast approximate local alignment
  - Multiple sequence alignment
  - Computing the alignments
- **Chapter 4 – Hidden Markov models**
  - The nose knows
  - Hidden Markov models
  - Profile HMMs
  - Finding genes with hidden Markov models
  - Case study: odorant receptors
  - Algorithms for HMM computations
- **Chapter 5 – Variation within and between species**
  - Variation in DNA sequences
  - Mitochondrial DNA: a model for the analysis of variation
  - Variation between species
  - Estimating genetic distance
  - Case study: are Neanderthals still among us?

- **Chapter 6 - Natural selection at the molecular level**
  - A mysterious disease
  - Evolution and natural selection
  - HIV and the human immune system
  - Quantifying natural selection on DNA sequences
  - Estimating KA/KS
  - Case study: natural selection and the HIV genome
- **Chapter 7 - Phylogenetic analysis**
  - Outbreak
  - On trees and evolution
  - Inferring trees
  - Case study: phylogenetic analysis of the SARS epidemic
  - The Newick format
- **Chapter 8 – Whole genome comparisons**
  - Uninvited guests
  - By leaps and bounds: patterns of genome evolution
  - Beanbag genomics
  - Synteny
- **Chapter 9 - Analysis of gene expression**
  - Chateau Hajji Feruz Tepe
  - Monitoring cellular communication
  - Microarray technologies
  - Case study: the diauxic shift and yeast gene expression
  - Bonus case study: cell-cycle regulated genes
- **Chapter 10 - Identification of regulatory sequences**
  - The circadian clock
  - Basic mechanisms of gene expression
  - Motif-finding strategies
  - Case study: the clock again

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Chapter 1		
Week 2	Chapter 2	Lab 1	
Week 3	Chapter 3		
Week 4	Chapter 4	Lab 2	Assignment I
Week 5	Chapter 5		
Week 6	Chapter 5	Lab 3	
Week 7	Mid Exam		

Week 8	Chapter 6		
Week 9	Chapter 6	Lab 4	Assignment II
Week 10	Chapter 7		
Week 11	Chapter 8	Lab 5	
Week 12	Chapter 9		
Week 13	Chapter 9	Lab 6	Assignment III
Week 14	Chapter 10		
Week 15	Final Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 3 Homework – 15%
- 3 Exams – 65%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- Introduction to Computational Genomics A Case Studies Approach by *Nello Cristianini and Matthew W. Hahn*
- Handbook of Computational Molecular Biology by *Srinivas Aluru*
- Biological Sequence Analysis: Probabilistic Models of Protein and Nucleic Acids, by *Durbin, Eddy, Krogh, Mitchison*
- Algorithms on strings, trees and sequences: Computer Science and Computational Biology by *Dan Gusfield*
- Genome-Scale Algorithm Design by *Makinen, Belazzougui, Cunial, Tomescu*

Introduction to Cognitive Science

Department: Software Engineering	
Module Title: Adaptive Computation and Machine Learning	Module Code: SITE-M4291
	Module ECTS: 37
Course Title: *Introduction to Cognitive Science*	Course Code: SITE - 5297
	Course ECTS: 5
	Course Duration: 16 weeks

Instructor's Contact Information:	**Course Information:**
Name:	Academic Year: 2016
Office:	Semester: I/II
Phone:	Meeting day:
Email:	Meeting time:
Office Hours:	Meeting location:

Course Description
Cognitive Science is an interdisciplinary field of inquiry that is concerned with the acquisition, representation, and use of knowledge by individual minds, brains, and machines, as well as groups, institutions, and other social entities. Because the fundamental purpose of the university, as a social institution, is the preservation, generation, and transmission of knowledge, cognitive science speaks to the heart of the university's mission. By engaging faculty from Psychology, Philosophy, Linguistics, Computer Science, Neuroscience, and Anthropology, Sociology, and other social sciences in common purpose, Cognitive Science constitutes a microcosm of the university as a whole.

Course Objectives

The objectives of the course are to introduce the basic concepts hypotheses, models, methods, issues, and debates in cognitive science. The course will cover the main information-processing paradigms in cognitive science as well as the main critiques of the paradigms.

By the end of the course, a student should know enough about cognitive science that the student can take advanced courses in cognitive science. Further, the knowledge and understanding acquired through this course should inform student's subsequent work on any application related to cognitive science, including human-centered computing, information system design, digital media, educational technology, design computing, human-robot interaction, etc.

Learning Outcomes

The program provides a knowledge and understanding of the following:

Practical Skills

- Understand and critically evaluate research and theory in cognitive psychology, including perception, attention, learning, memory, reasoning, problem-solving, judgment, and decision-making;
- Understand and critically evaluate research and theory in cognitive linguistics, with special attention to the relation between language and thought;
- Understand and critically evaluate various approaches to Artificial Intelligence, and the computational modeling of cognitive processes;
- Understand and critically evaluate the biological bases of cognitive functions, as uncovered by cognitive neuroscience;
- Understand and critically evaluate classic and contemporary work on the philosophy of mind, including the mind-body problem, mental causation, freedom of the will, and the nature of consciousness;
- Understand and critically evaluate the sociocultural context of individual cognition, including the social construction and organization of knowledge, cultural differences in cognition, the history of information, etc.

Prerequisites:

Student Workload

Lecture:32 hrs.	**Tutorial: 32 hrs**	**Home Study:65 hrs.**	**Laboratory: 48hrs**

Course Content

Chapter 1 – Introduction to Cognitive Sciences

Chapter 2 – Computational Model of Mind

Chapter 3 – Philosophy of Mind

Chapter 4 – Language and Thought

Chapter 5 – Vision and Color Perception

Chapter 6 – Language Processing

Chapter 7 - Metaphor

Chapter 8 - Conversation

Chapter 9 - Reasoning

Chapter 10 – Cognitive Development

Chapter 11 – Evolution of Mind

Schedule

Weeks	**Topics and Subtopics**		
	Lecture	**Laboratory**	**Assessment**
Week 1	Chapter 1		
Week 2	Chapter 2		
Week 3	Chapter 3		
Week 4	Chapter 3		Assignment I
Week 5	Chapter 4		
Week 6	Chapter 5		
Week 7	Chapter 5		Assignment II
Week 8	Mid Exam		
Week 9	Chapter 6		
Week 10	Chapter 6		
Week 11	Chapter 7		Assignment III
Week 12	Chapter 8		
Week 13	Chapter 9		
Week 14	Chapter 10		Assignment IV

Week 15	Chapter 11		
Week 16	Final Exam		

Teaching Methods

The teaching methods that shall be employed for this course are lecturing, demonstration, Lab exercises, classroom discussion and classroom action research and assignments and homework.

Assessment Method

The mark breakdown is as follows

- 4 Assignments – 20%
- 3 Homework – 15%
- 3 Exams – 65%

Course Policies

- **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course.
- **Assignments:** No Late Assignment will be accepted
- **Test/Quizzes:** Rarely reexamination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation.
- **Cheating/Plagiarism:** No second Chance or excuses.

References

- https://ocw.mit.edu/courses/brain-and-cognitive-sciences/9-10-cognitive-neuroscience-spring-2006/readings/
- MIND, An Introduction to Cognitive Science, *Paul Thagard, MIT Press,2nd edition, 2005*.
- Cambridge Handbook of Intelligence, *R. Sternberg & S. Kauffman (editors), Cambridge university Press, 2011*

Appendix 1C: Summary of SECT Curricula

Software Engineering Stream	
Degree Nomenclature	"Bachelor of Science in Software Engineering and Computing Technology (Software Engineering Stream)" "□□□□□ □□□□ □□□ □□□□□□ □□□□□□ □□ □□□□□□□ □□□□□ (□□□□□ □□□□□□ □□□)"
SECT base ECTS	211 ECTS credits
Software Engineering Stream ECTS	211 ECTS credits
Total ECTS	309 ECTS credits
Program Duration	5 Years
Revision Year	2010, 2018, 2021

Information Technology Stream	
Degree Nomenclature	"Bachelor of Science in Software Engineering and Computing Technology (Information Technology Stream)" "□□□□□ □□□□ □□□ □□□□□□ □□□□□□ □□ □□□□□□□ □□□□□ (□□□□□□□ □□□□□ □□□)"
SECT base ECTS	209 ECTS credits
Information Technology Stream ECTS	209 ECTS credits
Total ECTS	307 ECTS credits
Program Duration	5 Years
Revision Year	2010, 2018,2021

Artificial Intelligence Stream	
Degree Nomenclature	"Bachelor of Science in Software Engineering and Computing Technology (Artificial Intelligence Stream)" "□□□□□ □□□□ □□□ □□□□□□ □□□□□□ □□ □□□□□□□ □□□□□ (□□□□□□ □□□□□□□ □□□)"
SECT base ECTS	210 ECTS credits
Artificial Intelligence Stream ECTS	210 ECTS credits
Total ECTS	308 ECTS Credits
Program Duration	5 Years
Revision Year	2010, 2018,2021

Cyber Security Engineering Stream	
Degree Nomenclature	"Bachelor of Science in Software Engineering and Computing Technology (Cyber Security Engineering Stream)" "□□□□□ □□□□ □□□ □□□□□□ □□□□□□ □□ □□□□□□□ □□□□□ (□□□□ □□□□□ □□□□□□ □□□)"
SECT base ECTS	207 ECTS credits
Cyber Security Engineering Stream ECTS	207 ECTS credits
Total ECTS	305 ECTS credits
Program Duration	5 Years
Revision Year	2010, 2018, 2021

Probability and Statistics for Engineers (Stat-M2173)

Module Guidebook

Module Name	**Probability and Statistics for Engineers**					
Module Code	Stat-M2173					
Total ECTS of the module	**5**					
Total study hours in the module	**Home study**	**Lecture**	**Tutorial**	**Lab**	**Assessment**	**Total**
	48	48	32	-	7	135
Module category	**Service**					
Rationale of the module	Although in areas like engineering the thinking and arguments rely heavily on deterministic models engineering students must acquire basic knowledge and skills about probability models by way of introductory statistics and probability theory so that they will be able to statistical inference..					
Module description	The module is designed for non-statistics major students. and covers the topics: Meaning of statistics; methods of data collection; methods of data presentation; measures of central tendency; measures of variation; elementary probability; random variables; probability distributions, cumulative distribution function; common discrete probability distributions: Binomial, Poisson and Geometric; common continuous distributions: uniform, Normal and exponential; one and two dimensional random variables and their distributions, expectation, covariance and correlation; moments and moment generating functions: simple linear regression and correlation.					
Learning outcomes of the module	Upon completion of the module, students will be able to: □□demonstrate statistical techniques through principles of data collection, descriptive statistics, present data and give interpretations; □□make statistical inferences using the knowledge they have acquired about probability and distributions.					

Clustered courses in the module		
Course number	**Course name**	**ECTS**
Stat2171	Probability and Statistics for Engineers	5

Module Contents and Guidebook

Probability and Statistics for Engineers (Stat2171) Course Guidebook
Course Title/Code: Probability and Statistics for Engineers (Stat2171)
Credit: 5 ECTS
Old Code: Stat 276
Module title/code: Probability and Statistics for Engineers (Stat-M2173)
Course Type: Service
Prerequisite(s): None
Academic Year: _____
Semester: _____
Student's College/Faculty: EiABC
Department: _____
Program: Undergraduate
Enrollment: _____
Module Team Leader's Name: _____
Address: _____
Instructor's Name: _____
Address: _____
Class Hours: _____
Length of time to complete the course: 16 Weeks

Student's work Load					
Lecture	**Tutorial**	**Assessment**	**Lab**	**Home study**	**Total**
48	32	7	-	48	135

Course Outline

1. Basic Concepts, methods of data collection and presentation

1.1 Introduction
- 1.1.1 Definition and classification of Statistics
- 1.1.2 Stages in statistical investigation
- 1.1.3 Definition of Some Basic terms
- 1.1.4 Applications, uses and limitations of statistics
- 1.1.5 Types of variables and measurement scales

1.2 Methods of data collection and presentation
- 1.2.1 Methods of data collection
- 1.2.2 Sources and types of data
- 1.2.3 Methods of data presentation
  - 1.2.3.1 Frequency distributions
  - 1.2.3.2 Diagrammatic and/or graphical presentation of data: bar charts, pie-chart, pictogram, histogram, frequency polygon, ogive, Stem and leaf plot

2. Summarizing of Data

2.1 Measures of central tendency: objectives of measuring central tendency

2.2 Types of measures of central tendency
- 2.2.1 mean (arithmetic, weighted, geometric and harmonic), mode, median

2.3 Measures of location: quantiles (quartiles, deciles and percentiles)

2.4 Measures of dispersion/variation
- 2.4.1 range, variance, standard deviation and coefficient of variation

2.5 Standard scores

3. Elementary Probability

3.1 Deterministic and non-deterministic models

3.2 Review of set theory: sets, union, intersection, complementation, De Morgan's rules

3.3 Random experiments, sample space and events

3.4 Finite sample spaces and equally likely outcomes

3.5 Counting techniques

3.6 Definitions of probability

3.7 Derived theorems of probability

4. Conditional Probability and Independence

4.1 Conditional probability

4.2 Multiplication theorem, Bayes' Theorem, total probability theorem

4.3 Independent events

5. One-dimensional Random Variables

3.1 Random variable: definition and distribution function

3.2 Discrete random variables

3.3 Continuous random variables

3.4 Cumulative distribution function and its properties

6. Functions of Random Variables

6.1 Equivalent events

6.2 Functions of discrete random variables and their distributions

6.3 Functions of continuous random variables and their distributions

7. Two dimensional Random Variables

7.1 Two dimensional random variables

7.2 Joint distributions for discrete and continuous random variables

7.3 Marginal and conditional distributions

7.4 Independent random variables

7.5 Distributions of functions of two dimensional random variables

8. Expectation

8.1 Expectation of a random variable

8.2 Expectation of a function of a random variable

8.3 Properties of expectation

8.4 Variance of a random variable and its properties

8.5 Moments and moment generating function

8.6 Chebychev's Inequality

8.7 Covariance, correlation Coefficient

9. Common Probability distributions

9.1 Common Discrete Distributions and their Properties

9.1.1 Binomial distribution

9.1.2 Poisson distribution

9.1.3 Geometric distribution

9.2 Common Continuous Distributions and their Properties

9.2.1 Uniform distribution

9.2.2 Normal distribution

9.2.3 Exponential distribution

10. Simple Linear Regression and Correlation

9.5 Introduction

9.6 Fitting simple linear regression

9.7 Covariance and the correlation coefficient

9.8 Rank correlation coefficient

Suggested textbooks

Ross, S. (2006). A First Course in Probability (7th Ed.). Prentice-Hall, New Jersey.

Meyer L. P. (1970). Introductory Probability and Statistical Applications (2nd Ed.). Addison-Wesley Publ. Co., Massachusetts.

References

Cheaffer, R.L. and McClave, J.T (1994). Probability and Statistics for Engineers (4th Ed.). Duxbury Press, New York.

Lipschutz, S. and Schiller, J. (1998). Introduction to Probability and Statistics.

Mendenhall, W., Beaver, R.J. and Bearer, B.M. (2008). Introduction to Probability and Statistics (13th Ed.). Duxbury Press, New York.

Walpole, R. E., Myers, S.L. and Ye, K. (2006). Probability and Statistics for Engineers and Scientists (6th Ed.). Prentice-Hall, New Jersey.

Suhov, Y. and Kelbert, M. (2005). Probability and Statistics by Examples. Cambridge University Press.

Teaching and learning methods

Lectures, tutorials and assignments

Mode of Assessment

Tests and/or assignments 50%

Final Examination 50%

Total 100%