

I. Problem solving process

We face problems in our day-to-day life. Problem in this sense is one that can be solved using computers.

E.g. Requirements for

- Handling and manipulating employee data
- Solving mathematical equations based on formula
- etc.

A problem may be real problem (existing problem), or anticipated (potential) problem and reasons for problems may be

- Directives (in organizations)
- Opportunities (to do things better)
- Or other things.

In solving a problem, we may get various solutions. But all solutions involve the following formal problem solving steps.

1. Defining the problem
2. Planning the solution
3. Coding the program
4. Testing the program
5. Documentation

1. **Defining the problem:**

Before starting any part of the solution, we have to be introduced with the problem. First, we must understand thoroughly what the problem is. This should lead us to complete specifications of the problem to be addressed.

- What part of the problem is going to be solved?
- What input data is required to solve the problem?
- What output data (result) is expected?
- What procedures are needed to achieve the result?

There are various methods to find these specifications, like interview, observations, or for simpler problems in written form, read the requirements carefully

This and the next steps of the problem solving process are computer independent works.

2. **Planning the Solution:-**

This step involves the following tasks:

- Divide the original problem into a number of sub problems. These sub- problems, being necessarily smaller than the original problem, are easier to solve and their solution will be the components of our final solution. This method, by which a complex problem is divided into smaller and easier sub-problems is called “divide and conquer”
- Associating each task with a precise method to accomplish it, which consists of a sequence of well defined steps which, when carried out, perform the corresponding task. This sequence of steps is called and Algorithm

Algorithms are usually developed in pseudo code or using flowchart. Pseudo code is a language used to describe the manipulations to be performed on data. This language is a simple mixture of natural language and mathematical notations and it is independent of programming language.

Example: the pseudo code to calculate tax deduction of employees based on the following requirement:
For salary more than 500 birr, tax is 5% of the salary, and for salary less than 500 birr, tax is 3% of the salary. The pseudo code for this requirement may look like the following

```
Input salary
If salary > 500 birr
    set tax to salary * 0.05
Else
    set tax to salary * 0.03
Output tax
```

Planning is sketching the step-by-step activities that lead to the solution. This is formulating a problem in terms of the steps to its solution.

3. coding the program

Pseudo code or flow chart algorithms cannot be executed (performed) directly by the computer. They must first be translated into programming language, a process referred to as coding.

It is expressing the solutions (in the previous step) in one of the various programming language.

There are many high-level programming languages like BASIC, COBOL, Pascal, FORTRAN, C, C++, VB, etc. To get our program work, we write it based on the *syntax rules* of the programming language.

Syntax rule: correct way of writing or expressing commands that direct the control of the computer system.

4. Testing the program

It is very rare that a program can be written correctly the first time. Most programmers get used to the idea that there are errors in their newly written programs, these errors are detected during testing the program, and appropriate revision must be made and the tests return.

The process of coding and testing is called implementation.

5. Documentation

This activity starts with the first step of defining the problem and continues. It is compiling related documents throughout the lifetime of the program development. The documentation must include

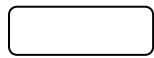
- Problem (requirement) definition
- Explanations of all the steps of the algorithms (the planning)
- The problems encountered during the writing & testing of the program
- Explanations of some important or ambiguous parts of the program code (comments)
- A printed copy of the program
- Users' manual that explains how to use the program, how to prepare the input data and how to interpret the program's results.

Though there can be many solutions (various programs) for one problem (requirement), we should select the best one based on its reliability, maintainability, portability, and efficiency (time & space).

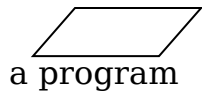
II. Algorithm Development

Algorithm is a finite set of well-defined rules (statements) for the solution of a problem in a finite number of steps. To design an algorithm for a problem first we break down the problem into simpler and manageable tasks. For one problem there may be a lot of algorithms that help to solve, but the algorithms that we select must be powerful, easy to maintain, and efficient (doesn't take too much space and time)

The most common ways to represent algorithms include pseudo code and flowchart. A flow chart consists of an ordered set of standard symbols (mostly geometrical shapes), which represent operations, data flow or decision.

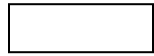
Major flowchart symbols**Terminal**

indicates start/end of a program

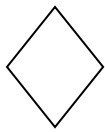
**Input/output**

a program

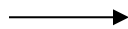
represent data input and data output of

**Process**

represents actions (operations) or group of operations (process)

**Decision**

represents decision to be made. It contains the conditions that determines which output path (or arrow) will be followed out of them. Usually two output paths based on the result of the condition (True /False)

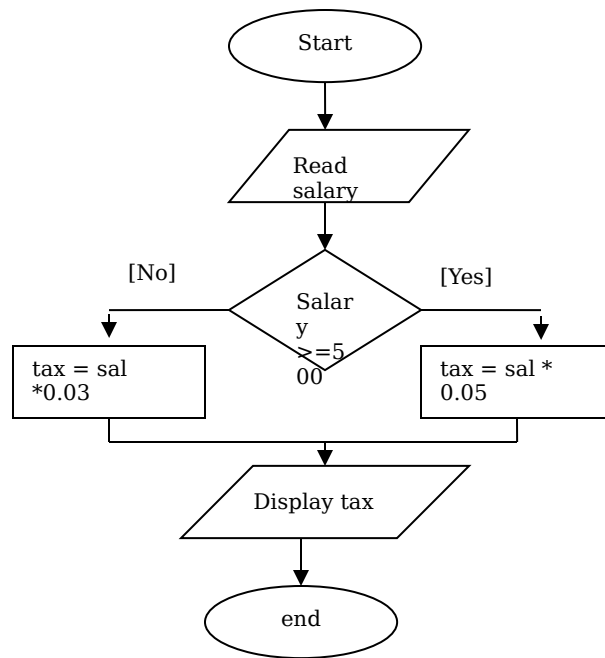
**Flow line**

indicates the direction of logical flow (a path from one operation to another)

Example: the flowchart to calculate tax deduction of employees based on the following requirement:

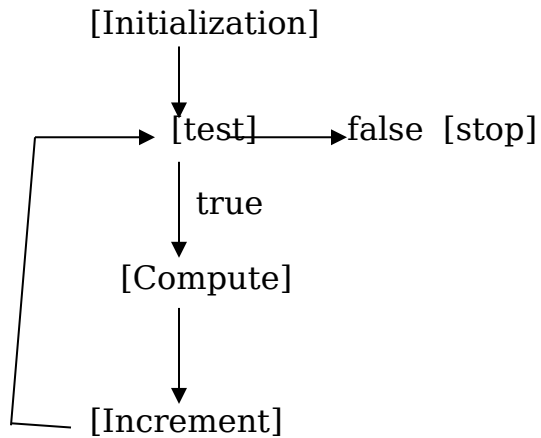
For salary more than 500 birr, tax is 5% of the

salary, and for salary less than 500 birr, tax is 3% of the salary. The flowchart for this requirement may look like the following



Loops

Some times there is a situation in which it is necessary to execute a group of statements repeatedly until some condition is satisfied. This situation is called a loop. Loop is a sequence of instruction, which is repeated until some specific condition occurs. The general execution flow structure of loops looks like the following.



The above simple structure tells that execution continues as far as the test holds true.

Example

Flowchart to find factorial of a given number

i.e. factorial of a given number n is: $n \times [n-1] \times [n-2] \times \dots \times 1$

