



Ch5 - Work Sheet (Practical Exercises)

Function

1. Write a function named *mult()* that accepts two floating-point numbers as a parameters, multiplies these two numbers and display the result.
2. Write a function that accepts a year as a user input and returns a one if the year is a leap year or a zero if it is not.
3. Write a modular program that reads a list of marks and display their range, mean, median.

Suggested functions and their prototypes:

```
Void readList(float myList[], unsigned & maxNo);  
Float findMax(const float myList [] , unsigned maxNo);  
Float findMin(const float myList [] , unsigned maxNo);  
Float findRange(const float myList [] , unsigned maxNo);  
Float findmean(const float myList [] , unsigned num);  
Void display(const float myList [] , unsigned maxNo);
```

4. For a function named **getProduct** and that has parameters namely **num1** and **num2** which their data type is integer and double respectively and that should multiply the integer value by the double number and then return the result as a double number.
 - Write a *function declaration (prototype)* for the getProduct function.
 - Write the *function call statement* to the getProduct function, and assigning its return value to a *product variable*.
 - Name the actual arguments firstNum and secondNum.
5. Write a function-based C++ statement that adds the cube of the number stored in the **num1** variable to the square root of the number stored in the **num2** variable. The program should prompt the user to enter the two number inside *main()* function. The statement should assign the result to the **answer** variable and return to the *main()* function. All of the variables have the double data type. (Tip: use built in library to find the cube and square of the numbers)



6. **(Hypotenuse Calculations)** Define a function `hypotenuse` that calculates the hypotenuse of a right triangle when the lengths of the other two sides are given. The method should take two arguments of type `double` and return the hypotenuse as a `double`. Incorporate this method into an application that reads values for `side1` and `side2` and performs the calculation with the `hypotenuse` method. Use `Math` methods `pow` and `sqrt` to determine the length of the hypotenuse for each of the triangles in the figure below. [Note: Class `Math` also provides method `hypot` to perform this calculation.]

Triangle	Side 1	Side 2
1	3.0	4.0
2	5.0	12.0
3	8.0	15.0

7. Write an application that calculates the product of a series of integers that are passed to function `product` using a variable-length argument list. Test your method with several calls, each with a different number of arguments.
8. Write a logical (i.e. Boolean) valued function which takes a single integer parameter, and returns "True" if and only if the integer is a prime number between 1 and 1000 (note that 1 is not usually counted as a prime number). Test your function by calling it from the main function with different inputs.

Hints:

- (i) if a number is not prime, it has at least one prime factor less than or equal to its square root,
- (ii) $(32 \times 32) = 1024$ and $1024 > 1000$.

9. Write a void function that takes **three int arguments by reference**. Your function should modify the values in the arguments so that the first argument contains **the largest value, the second the second-largest, and the third the smallest value**. Use the `swapValues` function to swap the values in each of the arguments if needed. Your function should print out the values before and after the swapping to show it works correctly.



10. Write a program that reads in a **length in feet and inches** and outputs the equivalent length in meters and centimeters. Use at least three functions: one for input, one or more for calculating, and one for output. Include a loop that lets the user repeat this computation for new input values until the user says he or she wants to end the program. There are 0.3048 meters in a foot, 100 centimeters in a meter, and 12 inches in a foot.
11. Write a program that combines the functions in the previous two Practice Programs. The program asks the user if he or she wants to convert from feet and inches to meters and centimeters or from meters and centimeters to feet and inches. The program then performs the desired conversion. Have the user respond by typing the integer 1 for one type of conversion and 2 for the other conversion. Include a loop that lets the user repeat this computation for new input values until the user says he or she wants to end the program.
12. Write the C++ program for a void function that receives **three double variables**: the first two *by value* and the last one *by reference*. Name the formal parameters **n1, n2, and answer**. The function should divide the n1 variable by the n2 variable and then store the result in the answer variable. Name the function calcQuotient. Also write an appropriate function prototype for the calcQuotient function. In addition, write a statement that invokes the calcQuotient function, passing it the actual parameters **num1, num2, and quotient** variables.
13. Write an overloaded function **max** that takes either two or three parameters of type double and returns the largest of them.
14. Write a function based C++ program that find the **Fibonacci series** of a given number. Define a function named **fibonacci** that has one argument with default argument and returns integer. The function is called both with parameter and without parameter.



15. In many mathematical problems, the parameters need to be **scaled**, either to convert between units or to standardize measurements to a certain range of values. Write a void function called `scale` which accepts a *double* passed by value and two *doubles* passed by reference. Your function should use the first parameter to scale the other two values. Write a program which tests scaling values up and down. Include a precondition which assumes the scale factor is not zero; use an assert case to terminate your program if this precondition is not satisfied. Your function should print out the values before and after scaling the values to show it works correctly.
16. Write a program that converts from 24-hour notation to 12-hour notation. For example, it should convert 14:25 to 2:25 PM. The input is given as two integers. There should be at least three functions, one for input, one to do the conversion, and one for output. Record the AM/PM information as a value of type `char`, 'A' for AM and 'P' for PM. Thus, the function for doing the conversions will have a call-by-reference formal parameter of type `char` to record whether it is AM or PM. (The function will have other parameters as well.) Include a loop that lets the user repeat this computation for new input values again and again until the user says he or she wants to end the program.
17. In cold weather, meteorologists report an index called the windchill factor, which takes into account the wind speed and the temperature. The index provides a measure of the chilling effect of wind at a given air temperature. Windchill may be approximated by the formula;

$$W = 13.12 + 0.6215 * t - 11.37 * v^{0.16} + 0.3965 * t * v^{0.016}$$

where

v = wind speed in m/sec

t = temperature in degrees Celsius: $t \leq 10$

W = windchill index (in degrees Celsius)

Write a function that returns the windchill index. Your code should ensure that the restriction on the temperature is not violated.



18. Write a function that calculates a discount applicable on the price of an item. Your function should have three arguments: the price as a reference to a *double*, the discount as a *double* value, and a *bool* to indicate if the discount is calculated as a percentage or a fixed amount. You should calculate the discount and modify the price of the item accordingly. The program for should also checks that the discount is not negative, and that the price of the item does not drop to zero after applying the discount. If it does violate these conditions, then your program should use an assert to exit the program
19. Determine the values displayed by each cout statement in the following program:

```
#include <iostream.h>
int firstnum=20;
void display(void);
int main() {
    int firstnum =20;
    cout<<"\nthe value of firstnum is<<firstnum<<endl;
    display();
    return 0;
}
void display(void) {
    cout<<"the value of firstnum is now<<endl;
    return;
}
```

20. Given the following code:

```
int sum(int 0) {
    if(n==0)    return 0;
    else       return(n+sum(n-1));
}
```

- a) show the terminating condition
- b) show the recursive part
- c) hand trace this piece of code for n=4 and find the output



21. Consider the following code fragment

```
#include <iostream.h>
char key;
long int number;
int main() {
    int a,b,c;
    double x,y;
    ...
    return 0;
}

double secnum;
int func1(int num1, int num2)
{
    int o,p;
    float q;
    ...
    return p;
}
double func2( float fisrt, float last)
{
    int a,b,c,o,p;
    floar r;
    double s,t,x;
    ...
    return s*t;
}
```

Draw a box around the appropriate section of the above code to enclose the scope of the variables key, secnum, y, and r.



Case study

1. Write a program that inputs a date (for example, July 4, 2008) and outputs the day of the week that corresponds to that date.

The implementation will require several functions.

➤ **bool isLeapYear(int year);**

This function should return true if year is a leap year and false if it is not. Here is pseudocode to determine a leap year:

leapYear = (year%400 ==0) or (year % 4 == 0 and year %100 != 0)

➤ **int getCenturyValue(int year);**

This function should take the first two digits of the year (that is, the century), divide by 4, and save the remainder. Subtract the remainder from 3 and return this value multiplied by 2. For example, the year 2008 becomes: (20/4) 5 5 with a remainder of 0. 3 2 0 5 3. Return 3 * 2 5 6.

➤ **int getYearValue(int year);**

This function computes a value based on the years since the beginning of the century. First, extract the last two digits of the year. For example, 08 is extracted for 2008. Next, factor in leap years. Divide the value from the previous step by 4 and discard the remainder. Add the two results together and return this value. For example, from 2008 we extract 08. Then (8/4) 5 2 with a remainder of 0. Return 2 1 8 5 10.

➤ **int getMonthValue(int month, int year);**

This function should return a value based on the table below and will require invoking the *isLeapYear* function.

Month	Return value	Month	Return value
January & October	0	June,	4
February, March, November	3	September, December	5
April, July	6	August	2
May	1		



Finally, to compute the day of the week, compute the sum of the date's day plus the values returned by `getMonthValue`, `getYearValue`, and `getCenturyValue`. Divide the sum by 7 and compute the remainder. A remainder of 0 corresponds to Sunday, 1 corresponds to Monday, etc., up to 6, which corresponds to Saturday.

For example, the date July 4, 2008 should be computed as (day of month) 1 (`getMonthValue`) 1 (`getYearValue`) 1 (`getCenturyValue`) 5 4 1 6 1 10 1 6 5 26. $26/7$ 5 3 with a remainder of 5. The fifth day of the week corresponds to Friday. Your program should allow the user to enter any date and output the corresponding day of the week in English. This program should include a void function named `getInput` that prompts the user for the date and returns the month, day, and year using pass-by-reference parameters. You may choose to have the user enter the date's month as either a number (1–12) or a month name