The Hashemite University

Prince Al-Hussein Bin Abdullah II Faculty for Information Technology
Department of Computer Sciences and Its application

Introduction to Programming Lab
LAB MANUAL

Updated on 21 March, 2023
By Majdoleen Hasan

Lab 1: **Basics of C++ Program**

Example 1: This program illustrates how to print text.

cout is an object that prints values or the string inside quotation marks " ". It is followed by the insertion operator **(<<)**.

```
#include<iostream>
using namespace std;
int main(){
cout<<"My first C++ program."<<endl;
return 0;
}
```

Output:

```
My first C++ program.
```

Example 2: This program illustrates how to read a values**.**
cin is an object that reads data from the keyboard with the extraction operator (>>).

```
#include<iostream>
using namespace std;
int main(){
int feet, inches;
cout<<"Enter two integers separated by spaces: ";
cin>> feet >> inches;
cout<< endl;
cout<<"Feet = "<< feet << endl; cout<<"Inches = "<< inches << endl;
return 0;
}
```

Output:

```
Enter two integers separated by spaces: 23 5

Feet = 23
Inches = 5
```

Exercises

1) Write a C++ Write a C++ program that prints your name, id and specialization on different lines.

2) Write a C++ program that contains two variables for floating-point numbers and initializes them with the values 123.456 and 76.543 Then display the sum and the difference of these two numbers on screen.

3) The following program contains several errors:

```
*/ Now you should not forget your glasses //
#include<stream>
int main
{
      a=5;
cout << " happy day!",
cout >> "Correct  the errors , ";
cout <<"a="a<<endl;
cout << 'Pass on what you have learned '<< " ." << endl.
return 0;
)
```

Correct the errors and run the program to test your changes.

Lab 2: **Arithmetic Operators**

Operator	Operation	Note
+	Addition	
-	Subtraction	
*****	Multiplication	asterisk is a star-shaped symbol (*).
/	Division	if an integer is divided by another integer, we will get the quotient. However, **if either divisor or dividend is a floating-point number, we will get the result in decimals.**
%	Modulus (remainder after division)	The % operator can only **be used with integers.**

Example 1: Write a C++ program that takes a length as an input in Feet and Inches. The program should then convert the length to Centimeters.

1 inch = 2.54 cm

1 foot = 12 inches

- **Problem analysis**

1- Input: length in feet and inches
2- Convert the length from feet to inches by multiply the number of feet with 12.
3- Calculate the total inches by adding the converted feet to the inches.
4- 1 inches is equal to 2.54 cm
5- Multiply total inches with 2.54.

6- Output: the equivalent length in centimeters
- Program computes the equivalent length in centimeters

- Needed variables

```
int feet; //variable to hold given feet
int inches; //variable to hold the given inches
int totalInches; //variable hold total inches
double centimeters; //variable to hold length in centimeters
```

- Needed Constants

```
const double CENTIMETERS_PER_INCH = 2.54;
const int INCHES_PER_FOOT = 12;
```

```
// The solution of example 1
#include<iostream>
using namespace std;
//declare constants
const double CENTIMETERS_PER_INCH = 2.54;
const int INCHES_PER_FOOT = 12;
int main(){
 int feet, inches, totalInches;// Step 1 declare variables
 double centimeter;
cout<<"Enter two integers, one for feet and one for inches: ";
 cin >> feet >> inches;    //Step 2
cout<< endl;
cout<<"The numbers you entered are "<< feet <<" for feet and "<<
inches<<" for inches. "<< endl;//Step 3
totalInches = INCHES_PER_FOOT * feet + inches;   //Step 4
cout<<"The total number of inches = "<<totalInches<< endl; //Step 5
centimeter = CENTIMETERS_PER_INCH * totalInches; //Step 6
cout<<"The number of centimeters = "<<centimeter<< endl;    //Step 7
return 0;
}
```

Output:

```
Enter two integers, one for feet and one for inches: 5 10

The numbers you entered are 5 for feet and 10 for inches.
The total number of inches = 70
The number of centimeters = 177.8
```

Exercises

1) Write C++ program that reads two integers (numb1, numb2) and find their:
   Summation (numb1+numb2)
   Multiplication (numb1*numb2)
   Subtraction (numb1-numb2)
   Division (numb1/numb2)
   Remainder (num1%num2)

2) Write a C++ program that reads the length of a square and calculate its Circumference and area.
   Hint: Circumference of a square = 4 × side
   Area of a square = length × length = length^2.

3) Write a C++ program that reads the radius of a circle and calculate its area.
   Hint : cir _area =22/7*radius^2 .

4) Write a C++ program that reads the temperature in Fahrenheit and convert it to Celsius. **Hint:** Celsius = ((5/9)*(fahrenhite-32).
   Example: 212 **°F** is equal to 100 **°C**.

5) Write a C++ program that reads a three-digit integer and prints the sum of all its digits.
   Hint: if the user enters 123, the result is 1+2+3=6. (Use mod and div in your program)

6) Write a C++ program to print the values of x and y, then swap the two values by using a third variable, reprint the new values of x and y.
   Consider the following two variables:
   X=10; Y=5;

Swap without using third variable

Re-type the previous program without using third variable.

7) Write a C++ program that prompts the user for a time in minutes and then displays the time in hours and minutes. For example, 184 minutes in hour: minute format is 3: 4 (Hint: use the division and modulus operators).

Strings in C++

- Strings are used for storing text.
- To use strings, you must include an additional header file in the source code, the **<string>** library: **#include <string>**
- A string variable contains a collection of characters surrounded by double quotes.
- To take input from the user which depend on the string. The most common way is to take input with **cin** keyword with the extraction operator (>>).

Example 1: Write a C++ program to create a variable of type string and assign it a value:

```
#include <iostream>
#include<string>
using namespace std;
int main() {
    string course="Welcome to introduction to programming lab";
      cout<<course <<endl;
    return 0;}
```

Output:

```
Welcome to introduction to programming lab
```

Example 2: Write a C++ program to create a variable of type string and take the value to it from the user:

```
#include <iostream>
#include<string>
using namespace std;
int main() {
      string name;
    cout<<"Enter your name"<<endl;
      cin>>name;
      cout<<"Your name is: "<<name<<endl;
    return 0;}
```

Output:

```
Enter your name
Majdoleen
Your name is: Majdoleen
```

Lab 3: **Selection Control Structures**

Example 1: Write a C++ program that reads a two integers and check whether they are equal or not.

```
#include<iostream>
using namespace std;
int main(){
int num1,num2;
cout<<"enter two integer numbers"<<endl;
cin>>num1>>num2;
if(num1==num2)
      cout<<"The two numbers are equal"<<endl;
else
      cout<<"The two numbers are not equal"<<endl;
return 0;
}
```

Output:

```
enter two integer numbers
7 8
The two numbers are not equal
```

Example 2: Write a C++ program that states whether an input number is positive, negative, or zero. Clearly, we have more than two options, so a single if statement will not work. Instead, use a nested if construction as follows:

```
#include<iostream>
using namespace std;
int main(){
double x; //the input number
cout<<"Enter the number:";
cin>> x;
if (x==0)
cout<<"iszero";
else
if (x>0)
cout <<"is positive"<<endl;
else
cout <<"is negative"<<endl;
return 0;
}
```

Output:

```
Enter the number: -8
is negative
```

Example 3: Switch statement

Write a C++ program to read a temperature in centigrade and display a suitable message according to temperature state below:

- Temperature 0-10 then Very Cold weather
- Temperature 10-20 then Cold weather
- Temperature 20-30 then Normal Temperature
- Temperature 30-40 then Its Hot
- Temperature >=40 then Its Very Hot

```
#include<iostream>
using namespace std;
int main(){
      int temperature;
      cout<<"enter a temperature in centigrade : "<<endl;
      cin>>temperature;
      switch(temperature/10) {//starts of switch statement
      case 0:
            cout<<"Very Cold weather"<<endl;
            break;
      case 1:
            cout<<"Cold weather"<<endl;
            break;
      case 2:
            cout<<"Normal Temperature"<<endl;
            break;
      case 3:
            cout<<"Its Hot"<<endl;
            break;
      default:
            cout<<"Its Very Hot"<<endl;
            break;

      }//ends of switch statement
return 0;
}
```

Output:

```
enter a temperature in centigrade:
25
Normal Temperature
```

If the **break** statement is omitted what the output will be?

Exercises

1) Write a C++ program that reads a floating-point number and prints "even" if the number is even and “odd “otherwise.

2) Write a C++ program reads angle measure, in degrees, and displays the type of angle corresponding to the degrees entered.
Hint: The angles are classified under the following types:

- Acute Angle – an angle measure less than 90 degrees
- Right Angle – an angle is exactly at 90 degrees
- Obtuse Angle – an angle whose measure is greater than 90 degrees and less than 180 degrees
- Straight Angle – an angle which is exactly at 180 degrees

3) Write a C++ program that asks the user to enter the employee basic salary, and then calculates the net salary according to the formula:

Net salary = basic salary –tax*basic salary

tax equals 0.16, if the basic salary is greater than 1000
tax equals 0.10, if the basic salary is between 500 and 1000
otherwise, the tax equals 0.08

4) Write a C++ program that prompts the user to enter an integer and determines whether it is divisible by both 5 and 6, whether it is divisible by 5 or 6 ,and whether it is not divisible by both 5 and 6.

5) Write a C++ program that asks the user to enter three integers and find:

1- The maximum among them
2- The minimum among them

6) Write C++ program that asks the user to enter two integers an operation (+,-,*, /), then find the result of entered operation (Hint: use switch statement)
Example: if the user entered: 35+ The output should be: 3+5=8

7) **Write a C++ program that do the following:**

A) Asks the user to enter a student mark, and prints "PASS” if the mark is greater than or equal to 50, and prints "FAIL! You must take the course again" otherwise.

B) if the student "PASS”, use nested if-else and logical operators to print the grade as follows:

"A”: if the mark was between (90-100)
"B”: if the mark was between (80-90)
"C”: if the mark was between (70-80)
"D”: if the mark was between (60-70)
"E”: if the mark was between (50-60)

C) Rewrite the code in "B” using switch statement.

Lab 4: **Repetition (Looping) Control Structures**

Counter controlled loop

In a Counter controlled loop, the number of iterations is known before the loop begins to execute.

Example 1: Write a C++ program to print all even numbers between 1 to 20.

```
// The solution of example 1
#include<iostream>
using namespace std;
int main(){
      cout<<"The even numbers from 1 to 20 are: "<<endl;
      for(int i=1;i<=20;i++)  // for loop
            if(i%2==0)
            cout<<i<<endl;
return 0;
}
```

Output:

```
The even numbers from 1 to 20 are:
2
4
6
8
10
12
14
16
18
20
```

Example 2: Write C++ program that prompts the user for five integers and prints the sum of them.

```
// The solution of example 2
#include<iostream>
using namespace std;
int main(){
      int i=1,num;
      double sum=0;
      while(i<=5){//starts while
            cout<<"enter an integer"<<endl;
            cin>>num;
     sum=sum+num;
       i++;

      }//ends while
      cout<<"The sum is: "<<sum<<endl;
return 0;
}
```

Output:

```
enter an integer
2
enter an integer
4
enter an integer
3
enter an integer
-2
enter an integer
8
The sum is: 15
```

Example 3: Write a C++ program that the prints the number of odd, even numbers and the number of zeros for N numbers.

```
// The solution of example 3
#include<iostream>
using namespace std;
int main(){
//Declare variables
int n;     //the number of numbers
int number;      //variable to store  number
int odds = 0,even=0,zeros=0;
cout<<"Please enter the number of numbers"<<endl;
cin>>n;
cout<<"Please enter "<< n <<" integers"<<endl;
for(int i=1;i<=n;i++){//starts for
      cin>>number;
       if(number==0)
            zeros++;
      else if(number%2==0)
            even++;
      else if(number%2!=0)
            odds++;

}//ends for
cout<< endl;
cout<<"The number of odd numbers is: "<< odds << endl;
cout<<"The number of even numbers is: "<< even << endl;
cout<<"The number of zeros  is: "<< zeros << endl;

return 0;
}
```

Output:

```
Please enter the number of numbers
10
Please enter 10 integers
1 2 9 8 -9 -16 20 0 3 0

The number of odd numbers is: 4
The number of even numbers is: 4
The number of zeros is: 2
```

Sentinel controlled loop

In a Sentinel controlled loop the number of iterations is not known before the loop starts executing. Also a special value called sentinel value is used to change the loop control expression from true to false in order to determine whether to execute the loop body.

Example 4:

Write a C++ program to keep asking for a number until you enter a negative number. At the end, print the sum of all entered numbers.

```
// The solution of example 4
#include<iostream>
using namespace std;
int main(){

int number;
int sum=0;
cout<<"Enter the number"<<endl;
cin>>number;
while(number>0)
{
      sum=sum+number;
cout<<"Enter the number"<<endl;
cin>>number;

}
cout<<"sum="<<sum<<endl;
return 0;
}
```

Output:

```
Enter the number
9
Enter the number
5
Enter the number
4
Enter the number
2
Enter the number
-5
sum=20
```

Example 5: Write a C++ program to create a simple menu-driven for calculation purposes that reads unspecified number of integers, and perform the following operations depending on the user's choice:

1. **Entered number 1 for addition.**

The user is asked to enter two integers. Then, the sum of these two integers is calculated and displayed on the screen.

2. **Entered number 2 for subtraction.**

The user is asked to enter two integers. Then, the difference of these two integers is calculated and displayed on the screen.

3. **Entered number 3 for multiplication.**

The user is asked to enter two integers. Then, the product of these two integers is calculated and displayed on the screen.

4. **Entered number 3 for Division.**

The user is asked to enter two integers. Then the first number is divided by the second and displayed the result on the screen.

5. **Entered number 4 to Quit.**
6. **Entered any other number** will display "Invalid choice!!! Please make a valid choice".

```
// The solution of example 5
#include<iostream>
using namespace std;
int main(){
      cout<<"Press 1 for Addition"<<endl;
      cout<<"Press 2 for Subtraction"<<endl;
      cout<<"Press 3 for Multiplication"<<endl;
      cout<<"Press 4 for Division"<<endl;
      cout<<"Press 5 to exit"<<endl;
int choice;
int number1,number2,result;
cout<<"Make your choice"<<endl;
cin>>choice;
while(choice!=5){//starts of while
switch(choice){//starts of switch
case 1:
cout<<"Enter the first  number"<<endl;
cin>>number1;
cout<<"Enter the second  number"<<endl;
cin>>number2;
result=number1+number2;
cout<<number1<<"+"<<number2<<"="<<result<<endl;
      break;
```

```
case 2:
cout<<"Enter the first  number"<<endl;
cin>>number1;
cout<<"Enter the second number"<<endl;
cin>>number2;
result=number1-number2;
cout<<number1<<"-"<<number2<<"="<<result<<endl;
      break;
case 3:
cout<<"Enter the first number"<<endl;
cin>>number1;
cout<<"Enter the second number"<<endl;
cin>>number2;
result=number1*number2;
cout<<number1<<"*"<<number2<<"="<<result<<endl;
      break;
case 4:
cout<<"Enter the first number"<<endl;
cin>>number1;
cout<<"Enter the second number"<<endl;
cin>>number2;
result=number1/number2;
cout<<number1<<"/"<<number2<<"="<<result<<endl;
      break;
default:
      cout<<"Invalid choice!!! Please make a valid choice"<<endl;

}//ends of switch
cout<<"Make your choice"<<endl;
cin>>choice;

}//ends of while

return 0;
}
```

Output:

```
Press 1 for Addition
Press 2 for Subtraction
Press 3 for Multiplication
Press 4 for Division
Press 5 to exit
Make your choice
1
Enter the first number
9
Enter the second number
3
9+3=12
Make your choice
2
Enter the first number
7
Enter the second number
2
7-2=5
Make your choice
5
Press any key to continue . . .
```

Example 6: Write a C++ program to display the cube of the numbers from 1 up to 10.

***Expected Output*:**

```
|------------|-----------|
| The number |  The Cube |
|------------|-----------|
|          1|          1|
|------------|-----------|
|          2|          8|
|------------|-----------|
|          3|         27|
|------------|-----------|
|          4|         64|
|------------|-----------|
|          5|        125|
|------------|-----------|
|          6|        216|
|------------|-----------|
|          7|        343|
|------------|-----------|
|          8|        512|
|------------|-----------|
|          9|        729|
|------------|-----------|
|         10|       1000|
|------------|-----------|
```

Using a simple cout stream we may not be able to format the output as shown above. Hence we can use the setw function from <iomanip> header, and we can set the specific width between the elements.

<iomanip>

The header <iomanip> consists of functions that are used to manipulate the output of the C++ program. We can make the output of any program neater and presentable based on where we want to show it or who is going to use it.

```
setw (int n);
```

Here the integer represents the number of characters that will be used as the width.

```
// The solution of example 6
#include<iostream>
#include<cmath>
#include<iomanip>
using namespace std;
int main(){
cout<<"|------------|-----------|"<<endl;
cout<<"| The number |  The Cube |"<<endl;
cout<<"|------------|-----------|"<<endl;
for(int i=1;i<=10;i++)
{
cout<<"|"<<setw(12)<<i<<"|"<<setw(11)<<pow(i,3.0)<<"|"<<endl;
cout<<"|------------|-----------|"<<endl;
}

return 0; }
```

Exercises

1) Write a C++ program to print the numbers from 1 to 10.

- Using while loop
- Using for loop

2) Write a C++ program that prints the following line of asterisks

****

- Using while loop
- Using do while loop
- Using for loop

3) Write a C++ the program that finds the sum of:

- All even numbers between 1 and 100
- All odd numbers between 1 and 100

4) Write a C++ program that reads the marks of unspecified number of students and calculates the average of the entered marks.
Hint: Marks Range: 0 -100.

5) Write a C++ program that reads the marks of N students. The program should display:

- Whether each student “PASS” or “FAIL”.
- The number of passed students and the number of failed students.
- The average.
- The minimum mark.
- The maximum mark.

6) Write a C++ program that asks the user to enter a number and check whether it is prime or not.

7) Write a C++ program that asks the user to enter a number and calculate the factorial of it.
Hint: The factorial of a number is the product of all the integers from 1 to that number.

8) Write a C++ program that allows the user to enter two numbers and finds the prime numbers in between.

9) Write a C++ program to print following:

A)	**B)**	**C)**
* ** *** **** *****	***** **** *** ** *	12345 1234 123 12 1
D)	**E)**	**F)**
* ** *** **** *****	***** **** *** ** *	1 00 111 0000 11111

10) Write a C++ program that calculates the factorials of the integers from 1 to 5. Print the results in tabular format.

Lab 5: **Functions**

A function is a group of statements that together perform a task which only runs when it is called.

- ✓ Dividing a complex problem into smaller chunks makes our program easy to understand and reusable.
- ✓ Every C++ program has at least one function, which is main().
- ✓ There are two types of function:

1. **Predefined (Built-in) Functions:**

   These are functions that are already present in C++, their definitions are already provided in the header files. The compiler picks the definition from header files and uses them in the program.

Example 1: A C++ program that demonstrates a set of mathematical functions.

```
#include<iostream>
#include<cmath>
using namespace std;

void main(){
double y=25;
cout << "Square root value of y=0.25 : " << sqrt(y) << endl;

int z = -10;
int a= 5;
double b =3;
cout << "Absolute value of z=-10 : " << abs(z) << endl;
cout << "Power value: a^b= (5,3.0) : " << pow(a,b) << endl;
}
```

Output:

```
Square root value of y=0.25 : 5

Absolute value of z=-10 : 10

Power value: a^b= (5,3.0) : 125
```

2. User-defined Functions:

These are functions that users create by themselves, wherein the user gives them own definition.

The syntax for creating a function is as follows:

```
return_type  function_name ( parameter list ) {
  //body of the function
}
```

- **Return Type**: A function may return a value. The **return_type** is the data type of the value the function returns such as double, int, …etc. But some functions perform the desired operations without returning a value. In this case, the **return_type** is the keyword **void**.

- **Function Name**: This is the actual name of the function.

- **Parameters**: The parameter list refers to the type, order, and number of the parameters of a function. Parameters are **optional**; that is, a function may contain no parameters.

- **Function Body** – The function body contains a collection of statements that define what the function does.

A C++ function definition consists of a function header and a function body and should be defined before the function call.

```
#include <iostream>
using namespace std;

return_type  function_name (parameter list) {
  //body of the function
}

void main()
{
function_name (parameter list);  // call the function
}
```

However, if we want to define a function after the function call, we need to use the function prototype. and the actual body of the function can be defined separately.

```
#include <iostream>
using namespace std;

return_type  function_name (parameter list); //function prototype

void main()
{
function_name (parameter list);  // call the function
}

return_type  function_name (parameter list) {
  //body of the function
}
```

Example 1: Write a C++ program that contains the following functions:

A. A function to find the sum of all numbers from 1 up to specific number.

B. A function that takes a number as a parameter and prints “The entered number is odd “, if the number is odd and “The entered number is even “, otherwise.

```
// The solution of example 1
#include<iostream>
using namespace std;
double findSum(int n);

 // function with single parameter and dosn't return a value
void checkEvenOdd(int num){
      if(num%2==0)
      cout<<"The entered number is even "<<endl;
      else
     cout<<"The entered number is odd "<<endl;

}
void main(){//starts main
      int number;
      cout<<"enter a number"<<endl;
      cin>>number;
```

```
cout<<"The sum="<<findSum(number)<<endl;//call the findSum() function.

checkEvenOdd(number);//call the checkEvenOdd() function.
}//ends main

 // function with single parameter and return a value
double findSum(int n){// starts function findSum
      double sum=0;
for(int i=1;i<=n;i++)
      sum+=i;
// return statement
return sum;
}// ends the function
```

Output:

```
enter a number
10
The sum=55
The entered number is even
```

Example 2: Suppose we need to create a program to calculate the maximum and the minimum number respectively among three numbers entered by the user. Use the following two functions to solve this problem:

- a function to compare the numbers and returns the maximum between two numbers and called get_max.
- a function to compare the numbers and returns the minimum between two numbers called get_min.

```
// The solution of example 2
#include<iostream>
using namespace std;
int get_max (int num1,int num2){
      if(num1>num2)
            return num1;
      return num2;
}
int get_min (int num1,int num2){
      if(num1<num2)
            return num1;
      return num2;
}
```

```
void main(){
      int n1,n2,n3;
      cout<<"Enter three integers"<<endl;
      cin>>n1>>n2>>n3;
      cout<<"The maximum number is:"<<get_max(n3,get_max(n1,n2))<<endl;
      cout<<"The minimum number is:"<< get_min(n3,
get_min(n1,n2))<<endl;
}
```

Output:

```
Enter three integers
10
3
50
The maximum number is:50
The minimum number is:3
```

Lab 6: **Function Overloading**

With **function overloading**, multiple functions can have the same name with different parameters. The parameters of the function should follow any one or more than one of the following conditions for Function:

1. They should have a different type
2. They should have a different number
3. They should have a different sequence of parameters.

Example 1: Write a C++ program to find Volume of Cube, Cylinder, Sphere using Function Overloading.

- The volume of a cube = a × a × a = a^3 , where 'a' is the length of the side of the cube.
- The volume of a cylinder = $\pi r^2 h$
- The volume of a sphere = **(4/3)** πr^3

```
/* C++ program to find Volume using Function Overloading  */
#include<iostream>
#include<cmath>
using namespace std;
int getVol(int);
double getVol(int,int);// different number
double getVol(double);// different type
double getVol(int,double);
double getVol(double,int);// different sequence
int main()
{
int r_cylinder,h,a;
float r_sphre;
cout<<"Enter side of cube:";
cin>>a;
cout<<"Enter radius and height of a cylinder:"<<endl;
cin>>r_cylinder>>h;
cout<<"Enter radius of sphere: ";
cin>>r_sphre;

cout<<"Volume of cube is "<<getVol(a)<<endl;
cout<<" Volume of cylinder is "<<getVol(r_cylinder,h)<<endl;
cout<<"Volume of sphere is "<<getVol(r_sphre)<<endl;
return 0;
}
int getVol(int length){//The volume of a cube
     return length*length*length;
}
```

```
double getVol(int radius,int height){//The volume of a cylinder
      return (22.0/7)*pow(radius,2.0)*height;
}
double getVol(double radius){//The volume of a sphere
      return (4.0/3)*(22.0/7)*pow(radius,3);
}
```

Exercise

Write the suitable code to implement the following functions:

```
double getVol(int,double);
double getVol(double,int);// different sequence
```

you have to overload them to calculate the volume of cylinder.

Lab 7: **Call by Value and Call by Reference**

If a function takes any arguments, it must declare variables that accept the values as arguments. These variables are called the formal parameters of the function.
Formal Parameters are the parameters that appear in the declaration of the function and **actual Parameters** are the parameters that appear in the function call statement which has been called.
There are two ways to pass value or data to function which is given below:

1) Call by Value

In call by value, original value cannot be changed or modified. In call by value, when you passed value to the function it is locally stored by the function parameter in stack memory location. If you change the value of function parameter, it is changed for the current function only but it not changes the value of variable inside the caller function

Example1 :

```
#include<iostream>
using namespace std;
  void increment(int s) {
            s=s+5000;
             cout << "salary after increment : " << s<<endl;
        }
void main() {
      int salary=4000;
        cout << "salary before function calling: " << salary<<endl;
        increment(salary);
        cout << "salary after function calling: " << salary<<endl;
}
```

Output:

```
salary before function calling: 4000
salary after increment: 9000
salary after function calling: 4000
```

2) Call by Reference

In call by reference, **original value is changed** or modified because we pass reference (address). Here, address of the value is passed in the function, so actual and formal parameters shares the same address space. Hence, any value changed inside the function, is reflected inside as well as outside the function.
In the call by reference, both formal and actual parameters share the same value. Calls by reference are preferred in cases where we do not want to make copies of objects or variables, but rather we want all operations to be performed on the same copy.

Example 2 :

```
#include<iostream>
using namespace std;
  void increment(int &s) {
           s=s+5000;
            cout << "salary after increment : " << s<<endl;
        }
void main() {
      int salary=4000;
        cout << "salary before function calling: " << salary<<endl;
        increment(salary);
        cout << "salary after function calling: " << salary<<endl;
}
```

Output:

```
salary before function calling: 4000
salary after increment: 9000
salary after function calling: 9000
```

Exercises

1) Write a C++ program that contains the following two functions:

- A function named **convertFtToM** that takes feet as double parameter and returns the conversion from feet to meter value.
- A function named **convertMToFt** that takes meterst as double parameter and returns the conversion from meter to feet value.

Hint: **1 Meter** = 3.280839895 feet **1 feet** = 0.3048 m

In the main function, prompts the user for feet and meters, and call the functions.

2) Write a C++ program that contains the following two functions:

A. A function named **isDivisbleBy10** that takes a number as a parameter and returns *true* if the number is divisible by 10 and *false* otherwise.

B. A function named **getFactorial** that takes an integer number as a parameter and returns the factorial of it. For example, if the number is **6** the factorial will be 1*2*3*4*5*6 = 720.

C. In the main function, call all functions you have declared.

3) Write a C++ program that contains the following function:

A. A function named **calculateAverage** that takes the sum of marks as double parameter, number of marks as integer and returns the average.

B. A function named **getGrade** that takes the mark as double and returns:
   A: (90<=mark <=100)
   B: (80<=mark <90)
   C: (70<=mark <80)
   D: (60<=mark <70)
   E: (50<=mark <60)
   F: otherwise ***Hint:*** *use switch statement.*

C. A function named **getResult** that takes the average of marks as double and returns 'P' if the average is greater than or equal 50 and 'F' otherwise.

D. A function named **printResult** that takes the average of marks as double and outputs the statement "Congratulations!!! You Passed!!!" if the result is "PASS", and the statement "Sorry!!! You Failed!!!" if the result is "Fail".

E. Implement the main() function as follows:

- Read unspecified number of marks for a student. ***Hint:*** *use a negative value to exit the loop.*
- Output the grade for each mark, the average and the final result of the student.

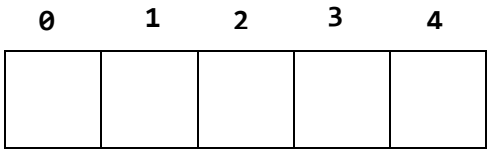
Lab 8: **Arrays**

An array is a series of elements of the same type placed in contiguous memory locations that can be individually referenced by adding an index to a unique identifier.

1) Declaring Arrays

```
datatype  arrayName[arraySize];
```

Note: The arraySize must be an integer constant greater than zero.
Example: double marks[5];



2) Initializing Arrays

You can initialize C++ array elements either one by one or using a single statement as follows:

int arr[3]={1,2,3};

The number of values between braces { } can not be larger than the number of elements that we declare for the array between square brackets [ ].

If you omit the size of the array, an array just big enough to hold the initialization is created.

int arr[]={1,2,3};

3) Access Elements in Array

In C++, each element in an array is associated with a number. The number is known as an array index. We can access elements of an array by using those indices.

```
arrayName[index];
```

4) Take Inputs from User and Store Them in an Array

5) Displaying Array Elements

Exercises

1) Write a C++ program that contains an array of size 5, the program will ask the user to fill the array with student's marks, and then find their average, number of passed students and the maximum mark. Declare also in the program another array that will contain 'P' if the corresponding mark is greater than or equal 50 and 'F', otherwise.

2) Write a C++ program that contains the following functions:

- A function named **fillSalaries** that fills an array with double values from the user. Use the header: void fillSalaries (double salaries[] , int size).

- A function named **getMaximum** that returns the maximum salary. Use the header: double **getMaximum** (double salaries [] , int size).

- A function named **getAverage** that finds the average of all salaries. Use the header: double **getAverage** (double salaries [] , int size).

- A function named **getGender** that finds the gender of the first salary greater than 500. char **getGender** (double salaries [] ,char gender[], int size).

- A function named **getGenderForMin** that finds the gender for the minimum salary. char **getGenderForMin** (double salaries [] ,char gender[], int size).

- **In the main function do the following:**
  - Declare an array named **gender** of **char** type, the size of it is 10, and fill it with characters from the user. *Hint: The user must enter **M** if the gender is male o r **F** if the gender is female.*
  - Declare an array named **salaries of** double type, the size of it is 10.
  - Call **all functions** you have declared before.

3) Write a C++ program that contains the following functions:

1) A function named **fillSales** that fills an array with double values from the user. Use the header: void **fillSales**(double sales[] , int size).

2) A function named **getMinimum** that returns the minimum sales in the sales array. Use the header: double **getMinimum** (double sales [] , int size).

3) A function named **getProductIndex** that searches for a specified product, if it presents in the **sales** array, then it returns the index of it. Otherwise, it returns -1. Use the header:

int **getProductIndex** (char products[], int size,char product).

4) **in the main function do the following:**

A. Declare an array named **sales** of double type, the size of it is 10.
B. Declare an array named **products** of char type and initialize it to 'a','b','c','d','e','f','g','h',I', and 'j'.
C. Call **all functions** you have declared before.