THE UNIVERSITY OF LAHORE
ISLAMABAD CAMPUS

Object Oriented Programming (CS02203)

Lab Report

Name:	Noor Baho
Registration #:	CSU-F12-116
Lab Report #:	01
Dated:	29-10-2018
Submitted To:	Mr. Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 1
Intoduction to Programming in C++

Objective
The objective of this lab is to introduce the students with basic syntax and structure of C++ programming language which includes variables, data types, and operators, types of operators and basic input and output from user.

Software Tool

- **OS Windows 10**
- **Editor DEV C++**
- **Language C++**

1 Theory

1.1 C++ Program

A computer program is a sequence of instructions that tell the computer what to do

1.2 Statements and Expressions

The most common type of instruction in a program is the statement. A statement in C++ is the smallest independent unit in the language. Statements in C++ are terminated by a semicolon (;).
There are many different kinds of statements in C++. The following are some of the most common types of simple statements:

```
int z;
z = 5;
```

```
std::cout << z;
int z is a declaration statement.
z = 5 is an assignment statement
std::cout << z is an output statement.
```

1.3 Functions

A function is a collection of statements that executes sequentially. Every C++ program must contain a special function called main. When the C++ program is run, execution starts with the first statement inside of main. Functions are typically written to do a very specific job.

1.4 Libraries and C++ Standard Library

A library is a collection of precompiled code (e.g. functions) that has been "packaged up" for reuse in many different programs. C++ also comes with a library called the C++ standard library that provides additional functionality for your use. One of the most commonly used parts of the C++ standard library is the iostream library, which contains functionality for writing to the screen and getting input from a console user.

```
#include <iostream>
int main()
{
std::cout << "Hello UOL!";
return 0;
}
```

Line 1: is a special type of statement called a preprocessor directive. Preprocessor directives tell the compiler to perform a special task. In this case, we are telling the compiler that we would like to add the contents of the iostream header to our program. The iostream header allows us to access functionality in the iostream library, which will allow us to write to the screen.
Line 2: declares the main () function, which is mandatory. Every program

must have a main () function.
Lines 3 and 7: tell the compiler which lines are part of the main function. Everything between the opening curly brace on line 3 and the closing curly brace on line 7 is considered part of the main () function.
Line 4: is our first statement and it is an output statement. std::cout is a special object that represents the console/screen. The ¡¡ symbol is an operator (much like + is an operator) called the output operator. std::cout understands that anything sent to it via the output operator should be printed on the screen. In this case, we're sending it the text "Hello world!".
Line 5: is a new type of statement, called a return statement. When an executable program finishes running, the main() function sends a value back to the operating system that indicates whether it was run successfully or not.

1.5 Variables

A variable in C++ is a name for a piece of memory that can be used to store information. In order to define a variable, we generally use a declaration statement.
Int z%;

1.6 Initialization of Variables

When declaring a variable, its value is by default undetermined. It is possible to store a concrete value in a variable at the same moment that the variable is declared. This is called initializing a variable.
int z$\bar{1}$0; Here variable z is initialized with value of 10

1.7 Fundamental Data Types

Fundamental data types are basic types implemented directly by the language that represent the basic storage units supported natively by most systems. They can mainly be classified into
Character types: They can represent a single character, such as 'A' or ´$; The most basic type is char, which is a one-byte character. Other types are also provided for wider characters.

- **Numerical integer types** : They can store a whole number value, such as 7 or 1024. They exist in a variety of sizes, and can either be signed or unsigned, depending on whether they support negative values or not.
- **Floating-point types** : They can represent real values, such as 3.14 or 0.01, with different levels of precision, depending on which of the three floating-point types is used.
- **Boolean type** : The boolean type, known in C++ as bool, can only represent one of two states, true or false.
- **Constants** : "Constants are expressions with a fixed value."
  A variable which does not change its value during execution of a program is known as a constant variable. Any attempt to change the value of a constant will result in an error message. A constant in C++ can be of any of the basic data types, const qualifier can be used to declare constant as shown below:
  const float PI = 3.1415;
  The above declaration means that PI is a constant of float types having a value 3.1415. Examples of valid constant declarations are:

  ```
  const int RATE = 50;
  const float PI = 3.1415;
  const char CH = 'A';
  ```

1.8 Operators

Once introduced to variables and constants, we can begin to operate with them by using operators. Operators are special symbols used for specific purposes. C++ provides six types of operators. Arithmetical operators, Relational operators, Logical operators, Unary operators, Assignment operators, Conditional operators, Comma operator
Arithmetical operators
Arithmetical operators +, -, *, /, and % are used to performs an arithmetic (numeric) operation. You can use the operators +, -, *, and / with both integral and floating-point data types. Modulus or remainder % operator is used only with the integral data type. Operators that have two operands are called binary operators.

Assignment operator
The assignment operator '=' is used for assigning a variable to a value. This operator takes the expression on its right-hand-side and places it into the variable on its left-hand-side. For example:
m = 5;
The operator takes the expression on the right, 5, and stores it in the variable on the left, m.
x = y = z = 32;
This code stores the value 32 in each of the three variables x, y, and z. In addition to standard assignment operator shown above, C++ also support compound assignment operators.
Relational operators
The relational operators are used to test the relation between two values. All relational operators are binary operators and therefore require two operands. A relational expression returns zero when the relation is false and a non-zero when it is true. The following table shows the relational operators.
Logical operators
The logical operators are used to combine one or more relational expression. The logical operators are
Unary operators: C++ provides two unary operators for which only one variable is required. For Example
a = - 50;
a = + 50;
Here plus sign (+) and minus sign (-) are unary because they are not used between two variables.
Increment and Decrement Operators: C++ provides two special operators '++' and '–' for incrementing and decrementing the value of a variable by 1. The increment/decrement operator can be used with any type of variable but it cannot be used with any constant. Increment and decrement operators each have two forms, pre and post.
Pre-increment: ++variable.
Post-increment: variable++
Pre-decrement: – – variable
Post-decrement: variable – –
In Prefix form first variable is first incremented / decremented, then evaluated whereas In Postfix form first variable is first evaluated, then incremented/decremented

```
int x, y;
int i = 10, j = 10;
x = ++i; //add one to i, store the result back in x
y = j++; //store the value of j to y then add one to j
cout << x; //11
cout << y; //10
```

Conditional operator
The conditional operator ?: is called ternary operator as it requires three operands. The format of the conditional operator is:
Conditiona_ expression ? expression1 : expression2;
If the value of conditional expression is true then the expression1 is evaluated, otherwise expression2 is evaluated.

```
int a = 5, b = 6;
big = (a > b) ? a : b;
```

The condition evaluates to false, therefore biggest the value from b and it becomes 6.
The comma operator
The comma operator gives left to right evaluation of expressions. When the set of expressions has to be evaluated for a value, only the rightmost expression is considered.

```
int a = 1, b = 2, c = 3, i;      // comma acts as separator, not as an oper
i = (a, b);                      // stores b into i
```

Would first assign the value of a to i, and then assign value of b to variable i. So, at the end, variable i would contain the value 2.
The sizeof operator (function sizeof())
The sizeof operator can be used to find how many bytes are required for an object to store in memory. For example
sizeof (char) returns 1
sizeof (float) returns 4
the sizeof operator determines the amount of memory required for an object at compile time rather than at run time.

1.9 Basic Input / Output

C++ uses a convenient abstraction called streams to perform input and output operations in sequential media such as the screen or the keyboard. The standard C++ library includes the header file iostream, where the standard input and output stream objects are declared.

- **cout console output**
- **cin console input**

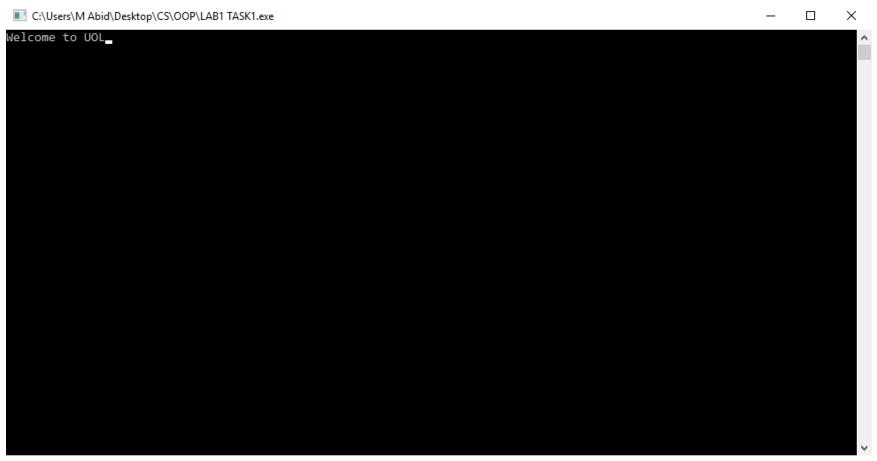
Standard Output (cout)
By default, the standard output of a program is the screen, and cout is used in conjunction with the insertion operator, which is written as ¡¡ (two "less than" signs).

```
cout << "Output sentence";       // prints Output sentence on screen
cout << 120;                     // prints number 120 on screen
cout << z;                       // prints the content of z on screen
To print more than one thing on the same line, the output operator
(<<) can be used multiple times. For example:
cout << "z is : " << z;          // prints z is: 4
std::endl;
If we want to print things to more than one line, we can do that by using
When used with cout, endl inserts a newline character (causing the cursor
 to the start of the next line).
cout << "Hi!" << endl;
cout << "My name is Alex." << endl;
```

output
Hi!
My name is Alex.
Standard Input (cin)
The standard input device is usually the keyboard. input in C++ is done by operator of extraction (¿¿) on the cin stream. The operator must be followed by the variable that will store the data that is going to be extracted from the stream. For example: int age;
cin ¿¿ age;

You can also use cin to request more than one datum input from the user:
cin ¿¿ a ¿¿ b;
is equivalent to
cin ¿¿ a;
cin ¿¿ b;
cin and strings
We can use cin to get strings with the extraction operator (¿¿) as we do with fundamental data type variables:
cin ¿¿ mystring;
However, cin extraction stops reading as soon as if finds any blank space character, so in this case we will be able to get just one word for each extraction. If we want to get a sentence from the user, this extraction operation would not be useful. In order to get entire lines, we can use the function getline(), which is the more recommendable way to get user input with cin:

```
int main ()
{
string name;
cout << ”Enter your name”;
getline (cin, name);
cout << ”Hello ” << name << ”!\n”;
return 0;
}
```


C:\Users\M Abid\Desktop\CS\OOP\LAB1 TASK1.exe

```
Welcome to UOL
```

Figure 1: Welcome to UOL

2 Task

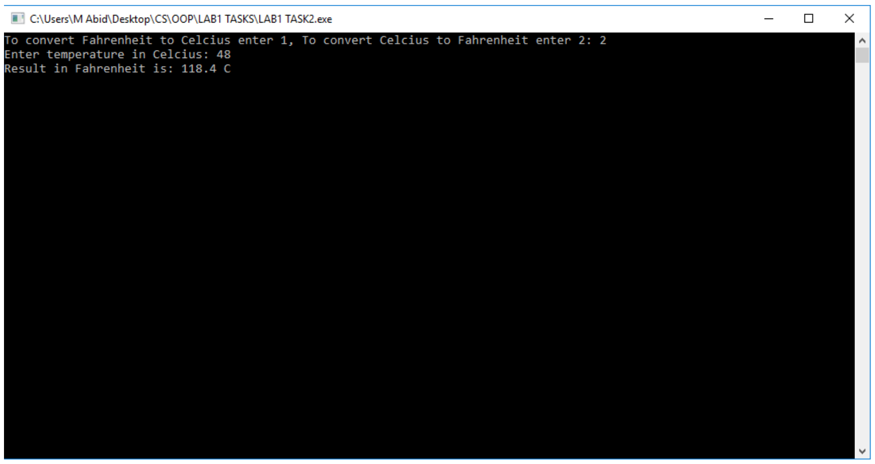
2.1 Procedure: Task 1

Write a program that displays “Welcome to UOL” in C++ on screen.
Solution:

```
#include <iostream>
#include <conio.h>
using namespace std;
int main ()
{
        cout<<"Welcome to UOL";

        getche();
        return 0;
}
```

Explaination: In above lines of codes, we were needed simply to print ”Welcome to UOL” on console screen, so we used standart output cout to print this.



```
To convert Fahrenheit to Celcius enter 1, To convert Celcius to Fahrenheit enter 2: 2
Enter temperature in Celcius: 48
Result in Fahrenheit is: 118.4 C
```

Figure 2: Temperature Converter

2.2 Procedure: Task 2

Write a program that converts temperature from Centigrade to Fahrenheit.
Solution:

```
#include <iostream>
#include <conio.h>
//temperature converter: Fahrenheit to Celcius and Celcius to Fahrenheit
using namespace std;

float f2c(float f)
{
        return (f-32)*5/9;
}

float c2f(float c)
{
        return (c*9/5)+32;
}

int main()
{
        int mod;
        float f, c;

        cout<<"To convert Fahrenheit to Celcius
         enter 1, To convert Celcius to Fahrenheit
         enter 2: ";
        cin>>mod;

        switch (mod)
        {
                case 1:
                        {
                                cout<<"Enter temperature in
                                 Fahrenheit: ";
                                cin>>f;
                                cout<<"Result in Celcius: "<<
                                f2c(f)<<" F";
                                break;
                        }
                case 2:
                        {
                                cout<<"Enter temperature in Celcius: ";
                                cin>>c;
                                cout<<"Result in Fahrenheit is: "
                               <<c2f(c)<<" C";
                                break;
                        }
        }

        getch();
        return 0;
}
```

Explaination: Required was to covert the temperature from celcius to fahrenheit and fahrenheit to celcius whereas user can input anyone from both as input. So we delcared a variable over which we used switch. We asked user to first select the mode of his input either celcius or fahrenheit. 1 for celcius and 2 for fahrenheit. Furtherly, we defined two functions, one is converting celcius input into fahrenheit output and other is performing fahrenheit into celcius output which will proceed according to the switch statement which will decide that which one function should be called. In functions we passed the input of user as argument and each function is returning the answer that is directly printing on screen without saving that input into any variable.

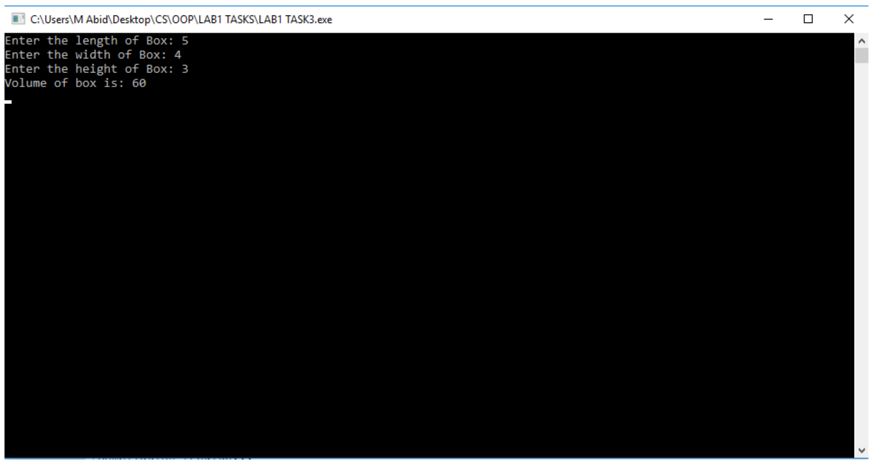

```
C:\Users\M Abid\Desktop\CS\OOP\LAB1 TASKS\LAB1 TASK3.exe
Enter the length of Box: 5
Enter the width of Box: 4
Enter the height of Box: 3
Volume of box is: 60
```

Figure 3: Volume of Box

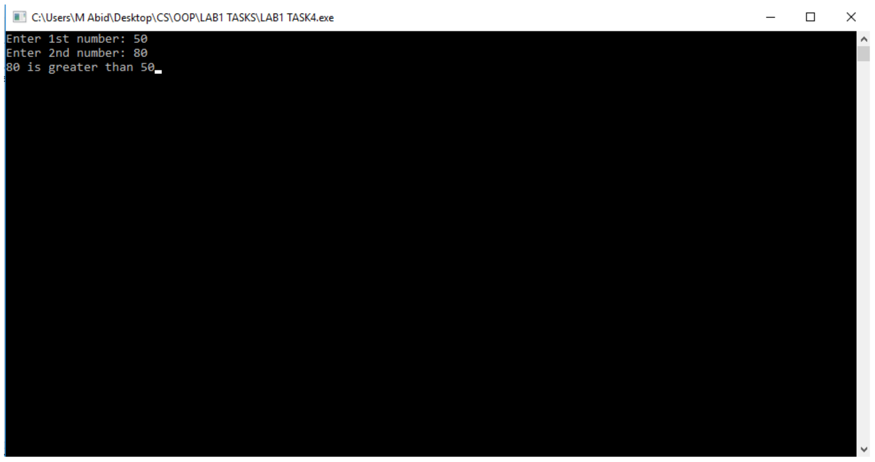
2.3 Procedure: Task 3

Write a C++ program to find the volume of box.
Solution:

```
#include<iostream>
#include<conio.h>
//finds volume of cube
using namespace std;
int main()
{
        float w, h, l;
        cout<<"Enter the length of Box: ";
        cin>>l;
        cout<<"Enter the width of Box: ";
        cin>>w;
        cout<<"Enter the height of Box: ";
        cin>>h;

        cout<<"Volume of box is: "<<l*w*h<<endl;
        getch();
        return 0;
}
```

Explaination: Required is to find volume of any box whose parameters will be provided by user. So for a box there are three parameters; length, width and height. So we declared three variables w, h and l for width, height and lenght respectively. Took input from user for the parameters of box, stored them into respective variables. As volume of box is the width*length*height, so using the same formula we directly cout the result of these three variables to the screen that shows the volume of desired box.



```
Enter 1st number: 50
Enter 2nd number: 80
80 is greater than 50
```

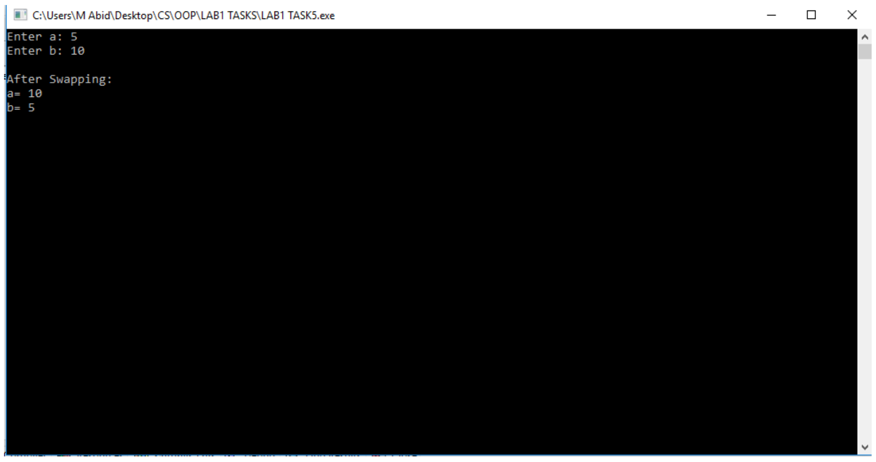
Figure 4: Greater Number Finder

2.4 Procedure: Task 4

Write a program that finds the greater number by using conditional operator. **Solution:**

```
#include <iostream>
#include <conio.h>
//finds greater number
using namespace std;
int main()
{
        float a,b;
        cout<<"Enter 1st number: ";
        cin>>a;
        cout<<"Enter 2nd number: ";
        cin>>b;
        if(a>b)
        cout<<a<<" is greater than "<<b;
        else if(b>a)
        cout<<b<<" is greater than "<<a;
        else
        cout<<"Both "<<b<<" & "<<a<<" are equal ";
        getch();
        return 0;
}
```

Explaination: Here required is to find the greater number. We took two integers from user as input and saved them into two variables a and b respectively. To compare them we sued if-else-if statement, first if is comparing is the number a is greater than b, then this block will execute and shows the 'a' 1st number is greater. If first if goes false then 2nd if connected with else will execute and will be comparing, is the number b is greater than a, then this block will excute and shows 'b' 2nd number is greater and if both ifs don't execute that automatically means that both numbers are equal that we executed in simple else that both the numbers are equal.



```
C:\Users\M Abid\Desktop\CS\OOP\LAB1 TASKS\LAB1 TASK5.exe
Enter a: 5
Enter b: 10

After Swapping:
a= 10
b= 5
```

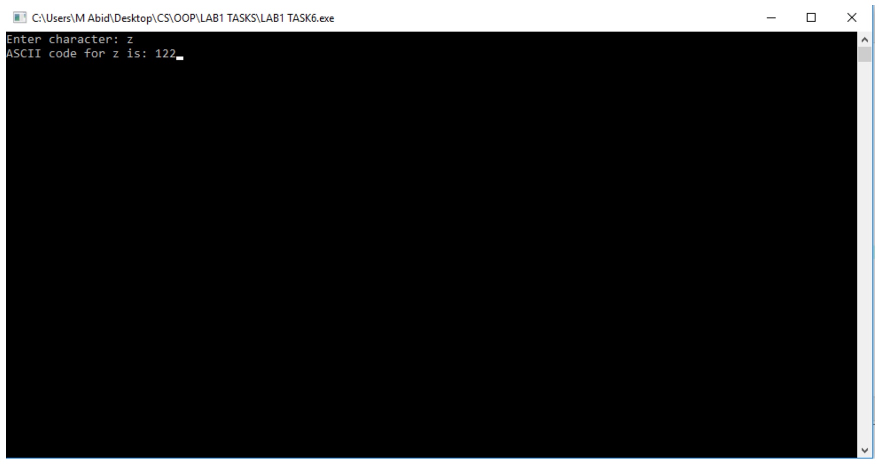
Figure 5: Swapping Variable Values without involving 3rd Variable

2.5 Procedure: Task 5

Write a program that swaps two numbers without involving third variable.
Solution:

```
#include <iostream>
#include<conio.h>
//swaps values of two variables without involving third variable
using namespace std;
int main()
{
        float a,b;
        cout<<"Enter a: ";
        cin>>a;
        cout<<"Enter b: ";
        cin>>b;
        a+=b;
        b=a-b;  //b got swapped value
        a=a-b;  //a got swapped value
        cout<<endl<<"After Swapping: "<<endl;
        cout<<"a= "<<a<<endl;
        cout<<"b= "<<b<<endl;
        getch( );
        return 0;
}
```

Explaination: We are required to swap to numbers between two variables without involving any third variable. So we declared only two integer type variables and took input into them from user. Now this swaping point is little tricky, we summed up number a and b into a by overwriting value of variable a. now a is the sum and b contains its value. then took difference of a and b and stored into b. As from sum if we subtract value of b, it will automatically result the value of initial a. 1st step done, we succeeded to swap value of a into b. now it comes to the value of b. Still a is containing the sum and b is containing the value of initial a. Now we will subtract b(that is actually initial a) from sum (a) that will give us the value of initial a and we will store it into a. So at the end we have value of initial a into b and value of initial b into a, means swapped done without involving any third variable.



```
Enter character: z
ASCII code for z is: 122
```

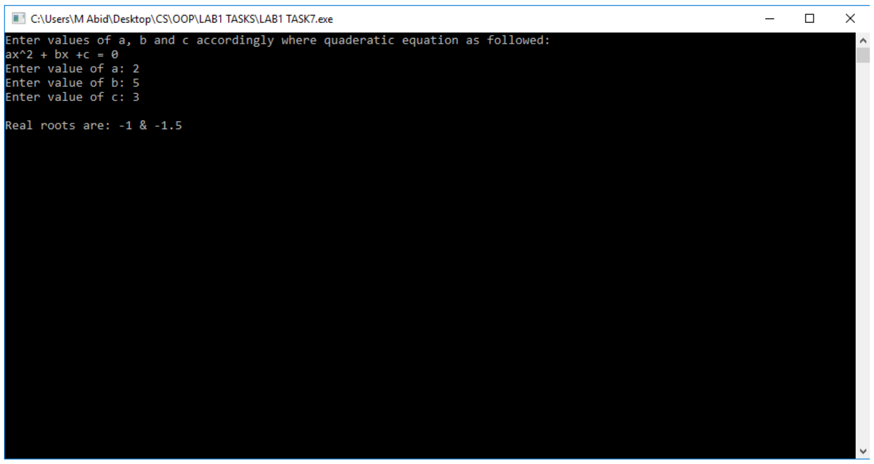
Figure 6: Character to ASCII Converter

2.6 Procedure: Task 6

Write a program that reads character and print their ASCII code.
Solution:

```
#include <iostream>
#include <conio.h>
//prints ASCI codes
using namespace std;
int asc(char x)
{
        return x;
}
int main()
{
        char a;
        cout<<"Enter character: ";
        cin>>a;
        cout<<"ASCII code for "<<a<<" is: "<<asc(a);
        getch();
        return 0;
}
```

Explaination: ASCII codes are required to be shown to user whatever character he inputs. So we declared a character type variable which is storing character input from user. We defined a function named asc(), which is taking argument as character type (user input) and returning the value into integer type, which will return the ASCII value of the character inserted. So when we called the function, it returned the ASCII value of that charecter inserted by user and we directly printed that returned value infront of user to show him his required ASCII value.



```
C:\Users\M Abid\Desktop\CS\OOP\LAB1 TASKS\LAB1 TASK7.exe
Enter values of a, b and c accordingly where quaderatic equation as followed:
ax^2 + bx +c = 0
Enter value of a: 2
Enter value of b: 5
Enter value of c: 3

Real roots are: -1 & -1.5
```

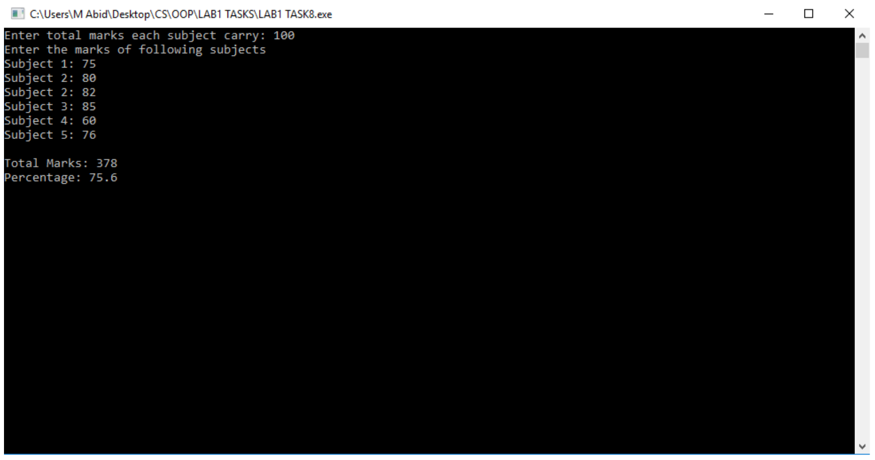
Figure 7: Real Root Calculator of Quadratic Equation

2.7 Procedure: Task 7

Write a program to find real roots of a quadratic equation.
Solution:

```
#include<iostream>
#include<conio.h>
#include<cmath>
//finds real roots of quadratic equation
using namespace std;
int main()
{
        float x, y, s;  //roots
        float a,b,c;    //quaderatic equation values
        cout<<"Enter values of a, b and c accordingly
         where quaderatic equation as followed: "<<endl
        <<"ax^2 + bx +c = 0"<<endl;
        cout<<"Enter value of a: ";
        cin>>a;
        cout<<"Enter value of b: ";
        cin>>b;
        cout<<"Enter value of c: ";
        cin>>c;
        s=(b*b)-(4*a*c);
        x=(-b+sqrt(s))/(2*a);
        y=(-b-sqrt(s))/(2*a);
        cout<<endl<<"Real roots are: "<<x<< " & "<<y;
        getch();
        return 0;
}
```

Explaination: Required is a Mathematic problem to be solved, to find the real roots of a quadratic equation. So we will simply consult Mathematics and find out the quadratic formula for a quadratic equation. We will declare two float variables x and y for root, the result to be saved and a, b and c to take input from user. user will provide us values of a, b and c as per the equation. using quaderatic formula we will find out the roots. As we know quadratic formula can be split into two, with positive and other one with negative. so in x we considered positive and in y we stored the results of negative side. Square root is also involved so we used library of "cmath" which will help us to find square root easily.At the end, both the roots are shown seperately.



```
Enter total marks each subject carry: 100
Enter the marks of following subjects
Subject 1: 75
Subject 2: 80
Subject 2: 82
Subject 3: 85
Subject 4: 60
Subject 5: 76

Total Marks: 378
Percentage: 75.6
```

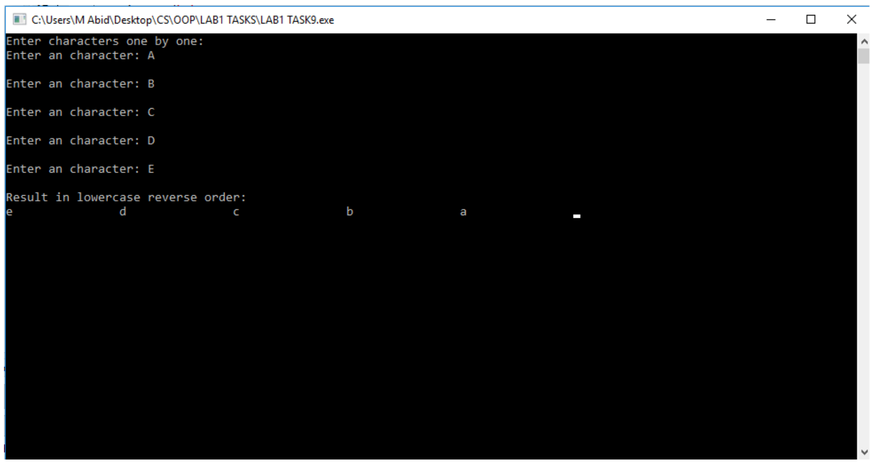
Figure 8: Total Marks & Percentage Calculator

2.8 Procedure: Task 8

Write a program that calculates total marks and percentage of a student in 5 different subjects.
Solution:

```
#include <iostream>
 #include<conio.h>
 //finds total marks & percentage of subjects
 using namespace std;
 int main()
 {
        float st, s1, s2, s3, s4, s5, total, perc;
        cout<<"Enter total marks each subject carry: ";
        cin>>st;
        cout<<"Enter the marks of following subjects"<<endl;
        cout<<"Subject 1: ";
        cin>>s1;
        cout<<"Subject 2: ";
        cin>>s2;
        cout<<"Subject 2: ";
        cin>>s2;
        cout<<"Subject 3: ";
        cin>>s3;
        cout<<"Subject 4: ";
        cin>>s4;
        cout<<"Subject 5: ";
        cin>>s5;
        st*=5;
        total= s1+s2+s3+s4+s5;
        perc= total/st*100;
        cout<<endl<<"Total Marks: "<<total<<endl;
        cout<<"Percentage: "<<perc;
        getch();
        return 0;
}
```

Explaination: Two results are required to be achieved from the user provided data of a student's marks. So we declared 5 variables for subjects' marks, 'total' for sum of marks, 'perc' for pecentage accumulation and st to count the number of subjects that will help us in calculating percentage. Took input from user related to the marks of subjects. First of all summed them up. the using percentage formula, accumulated percentage and stored into relevant variables. Finally shown to user the relevant data he requierd.



```
Enter characters one by one:
Enter an character: A

Enter an character: B

Enter an character: C

Enter an character: D

Enter an character: E

Result in lowercase reverse order:
e               d               c               b               a
```

Figure 9: Upper Case Character to Lower Case Character Printer

2.9 Procedure: Task 9

Write a program that accept five characters in uppercase and print it in lowercase in reverse order.
Solution:

```
#include <iostream>
#include <conio.h>
//Takes five uppercase characters as input and gives lowercase characters r
using namespace std;
int main()
{
        int asc;
        char word[4]={0};

        //taking input
        cout<<"Enter characters one by one: "<<endl;
        for(int a=0; a<=4; a++)
        {
                cout<<"Enter an character: ";
                cin>>word[a];
                cout<<endl;
        }
        //converting into lowercase
        for(int a=0; a<=4; a++)
        {
                if(word[a]!=0)
                {
                        asc=word[a];
                        asc+=32;
                        word[a]=asc;
                }
        }



        //giving output
        cout<<"Result in lowercase reverse order: "<<endl;
        for(int a=4; a>=0; a--)
        {
                        cout<<word[a];
                        cout<<"          ";
        }
        getch();
        return 0;
}
```

Explaination: Required is to take 5 characters input from user in Upper case and show that stream of characters into lower case but in reverse order. So we declared an array of size 4 which will be containing 5 indexes 0-4. Took input from user by using for loop and storing each character entered by user into each respective index. Then we used for loop with nested if statement to convert these characters into lower case. for loop is choosing each index (lower case character) and passing it to if statement that is converting it by getting ASCII of uppercase, adding number 32 into that ASCII and again converting that ASCII into character type which is stored back to the same index of array. So lower case conversion is done here. Now its time to reverse there order. So we used a for loop that is printing the indexes of array in reverse order form 4 to 3 to 2 to 1 to 0.

3 Conclusion

We got revised the basic sytax of C++ and also got fimiliar to the use of data types (char, int, float etc) which one should be used where to get the desired results, use of conditional operators; how to compare through if statements, use of if-else and switches as well as also practiced the use of increment/ decrement operators in the loops to make loop running for required number of times.