



HIMALAYA COLLEGE OF ENGINEERING

Advanced C++ Programming Lab Report

Lab 1: Basics of C++ programming

Prepared By: Aditya Man Shrestha

Subject: Object Oriented Programming (OOP)

Program: Bachelors of Electronics and Computer Engineering

Institution: Himalaya College of Engineering

Date: May 25, 2025

Objectives:

- To apply control structures, functions, and standard libraries in C++ programming.
- To carry out calculations for quadratic equations using the standard formula.
- To develop logic that verifies the validity of a triangle and identifies its type.
- To perform string operations and character analysis to assess the strength of a password.

Tools and Libraries Used:

- Programming Language: C++
- IDE: Code::Blocks
- Libraries: `#include <iostream>`, `include <string>`, `#include <math>`

Theory:

Basics of C++ Programming

C++ is a powerful and flexible language used to create efficient applications.

Beginners typically begin by understanding variables, conditional statements, and loops to tackle basic problems.

Structure of a C++ Program

A simple C++ program begins with header files (such as `<iostream>`) and defines the **main ()** function as the starting point. It often includes **using namespace std;** to simplify access to standard library features. The main function holds the program's logic and typically concludes with **return 0;** to signal successful execution.

Example:

```
#include<iostream>

using namespace std;

int main() {
    cout << "Hello world!";
    return 0;
}
```

Variables and Data Types

A **variable** is a named storage location in memory used to store data that can change during program execution. You must declare a variable before using it, specifying its **data type**.

Common Data Types in C++

Data Type	Description	Example
int	Stores integers	int age = 25;
float	Stores decimal numbers	float pi = 3.14;
double	Stores decimal numbers (double precision)	double salary = 4567.89;
char	Stores a single character	char grade = 'A';
string	Stores a sequence of characters	string name = "Alex";
bool	Stores true or false values	bool isPassed = true;

Variable Naming Rules

- Must start with a **letter** (A–Z or a–z) or an underscore (`_`)
- Can contain letters, numbers, and underscores
- Cannot start with a number
- Cannot use C++ reserved words (like `int`, `while`, `return`)
- Case-sensitive (e.g., `value`, `Value`, and `VALUE` are different)
- Use meaningful names (e.g., `age`, `totalScore` instead of `a`, `x`)

Conditional Statements in C++

Used to make decisions in a program.

1. if statement

Executes code if a condition is true.

```
if (condition) {  
    // code  
}
```

2. if...else statement

Executes one block if the condition is true, another if false.

```
if (condition) {  
    // code if true  
} else {  
    // code if false  
}
```

3. else if ladder

Checks multiple conditions one by one.

```
if (condition1) {  
    // code  
} else if (condition2) {  
    // code  
} else {  
    // default code  
}
```

4. switch statement

Selects code to run based on a value.

```
switch (value) {  
    case option1:  
        // code  
        break;  
    case option2:  
        // code  
        break;  
    default:  
        // default code  
}
```

Loops in C++

Repeat a block of code.

1. for loop

Used when the number of repetitions is known.

```
for (initialisation; condition; update) {  
    // code  
}
```

2. while loop

Used when the condition is checked before looping.

```
while (condition) {  
    // code  
}
```

3. do...while Loop

Executes the code **at least once**, then repeats as long as the condition is true.

```
do {  
    // code  
} while (condition);
```

//qn1. Solve $ax^2 + bx + c = 0$ and handle all discriminant cases.

Codes:

```
#include<iostream>
#include<math.h>
using namespace std;
int main(){
    float d,x1,x2,a,b,c;
    cout<<"The given equation is ax^2+bx+c : "<<endl;
    cin>>a>>b>>c;
    if(a==0) {
        cout<<"Error: The given equation is not quadratic.;
    }
    else {
        d=(b*b)-4*a*c;
        if(d==0) {
            cout<<"There exists one common root. "<<endl;
            x1=-b/(2*a);
            cout<<"The root is: "<<x1;
        }
        else if(d>0) {
            cout<<"There exists two distinct roots. "<<endl;
            x1=(-b+sqrt(d))/(2*a);
            x2=(-b-sqrt(d))/(2*a);
            cout<<"The roots are: "<<x1<<" and "<<x2;
        }
        else {
            cout<<"There exists two complex roots. "<<endl;
            x1=(-b)/(2*a);
            x2=sqrt(-d)/(2*a);
            cout<<"The roots are: "<<x1<<" +i"<<x2<<" and "<<x1<<" -i"<<x2;
        }
    }
}
```

Output:

```
The given equation is  $ax^2+bx+c$  :  
a : 0  
b : 4  
c : 5  
Error: The given equation is not quadratic.  
The given equation is  $ax^2+bx+c$  :  
a : 1  
b : -4  
c : 4  
There exists one common root.  
The root is: 2  
  
The given equation is  $ax^2+bx+c$  :  
a : 1  
b : 4  
c : 3  
There exists two distinct roots.  
The roots are: -1 and -3  
  
The given equation is  $ax^2+bx+c$  :  
a : 2  
b : 3  
c : 4  
There exists two complex roots.  
The roots are: -0.75+i1.19896 and -0.75-i1.19896
```

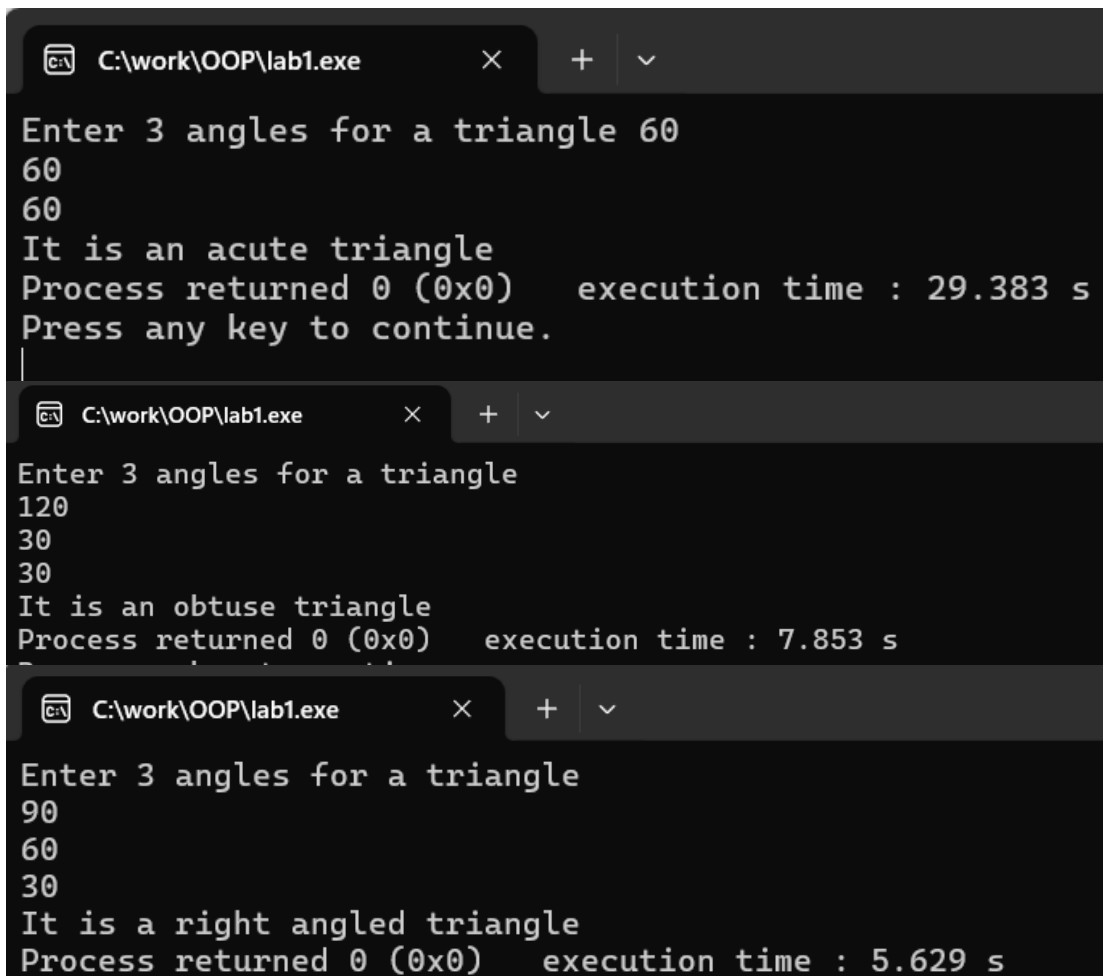
//qn2. WAP To check if input triangle is acute, obtuse or right angled triangle.

Codes:

```
#include<iostream>  
using namespace std;  
int main() {  
    float angle1, angle2, angle3;  
    cout<<"Enter 3 angles for a triangle";  
    cin>>angle1>>angle2>>angle3;  
    if(angle1+angle2+angle3 != 180)  
        cout<<"Invalid triangle";  
    else {  
        if(angle1>90||angle2>90||angle3>90)  
            cout<<"It is an obtuse triangle";  
        else if(angle1<90 && angle2<90 && angle3<90)  
            cout<<"It is an acute triangle";  
    }  
}
```

```
        else if (angle1==90||angle2==90||angle3==90)
            cout<<"It is a right angled triangle";
        else
            cout<<"not a triangle";
    }
}
```

Output:



The image displays three sequential screenshots of a Windows command prompt window running a C++ program. Each window has a title bar showing 'C:\work\OOP\lab1.exe'. The first screenshot shows the user entering three 60-degree angles, resulting in the output 'It is an acute triangle'. The second screenshot shows the user entering angles of 120, 30, and 30 degrees, resulting in the output 'It is an obtuse triangle'. The third screenshot shows the user entering angles of 90, 60, and 30 degrees, resulting in the output 'It is a right angled triangle'. Each output is followed by 'Process returned 0 (0x0)' and the execution time in seconds.

```
C:\work\OOP\lab1.exe
Enter 3 angles for a triangle
60
60
It is an acute triangle
Process returned 0 (0x0)    execution time : 29.383 s
Press any key to continue.

C:\work\OOP\lab1.exe
Enter 3 angles for a triangle
120
30
30
It is an obtuse triangle
Process returned 0 (0x0)    execution time : 7.853 s

C:\work\OOP\lab1.exe
Enter 3 angles for a triangle
90
60
30
It is a right angled triangle
Process returned 0 (0x0)    execution time : 5.629 s
```


//qn3. Check password strength based on length and character rules.

Codes:

```
#include <iostream>
#include <string>
using namespace std;
int Uppercase(string password) {
    for (int i = 0; i < password.length(); i++) {
        if (password[i] >= 'A' && password[i] <= 'Z') {
            return 1;
        }
    }
    cout << "Password MUST include at least one Capital letter.\n";
    return 0;
}
int Lowercase(string password) {
    for (int i = 0; i < password.length(); i++) {
        if (password[i] >= 'a' && password[i] <= 'z') {
            return 1;
        }
    }
    cout << "Password MUST include at least one Small letter.\n";
    return 0; }
int Digit(string password) {
    for (int i = 0; i < password.length(); i++) {
        if (password[i] >= '0' && password[i] <= '9') {
            return 1;
        }
    }
    cout << "Password MUST include at least one Number.\n";
    return 0; }
int SpecialChar(string password) {
    for (int i = 0; i < password.length(); i++) {
        char ch = password[i];
        if ((ch >= 33 && ch <= 47) || (ch >= 58 && ch <= 64) ||
            (ch >= 91 && ch <= 96) || (ch >= 123 && ch <= 126)) {
            return 1;
        }
    }
}
```

```

    }}
    cout << "Password MUST include at least one Special character.\n";
    return 0; }

int main() {
    string password;
    here:
    cout << "Enter your password: ";
    cin >> password;
    while (password.length() < 8) {
        cout << "Password must be at least 8 characters long.\n";
        cout << "Enter a longer password: ";
        cin >> password;    }
    int upper = Uppercase(password);
    int lower = Lowercase(password);
    int digit = Digit(password);
    int special = SpecialChar(password);
    if (upper == 1 && lower == 1 && digit == 1 && special == 1) {
        cout << "Password is strong.\n";
    } else {
        cout << "Please make a stronger password.\n";
        goto here;
    }
}

```

Output:

```

C:\work\OOP\password.exe
Enter your password: aditya
Password must be at least 8 characters long.
Enter a longer password: 123aditya
Password MUST include at least one Capital letter.
Password MUST include at least one Special character.
Please make a stronger password.
Enter your password: 123Adity@
Password is strong.

Process returned 0 (0x0)   execution time : 41.408 s
Press any key to continue.

```

Conclusion & Discussion:

In this lab, we applied fundamental C++ concepts by solving a variety of practical problems. The focus was on using control structures, functions, and string manipulation techniques. This hands-on experience helped reinforce core programming skills and laid a strong foundation for more advanced topics, such as C++ classes and object-oriented programming (OOP). Some difficulties occurred while conducting this lab experiment. The file was not running due to multiple 'main' being included in the same file. Logical errors and syntax errors were quite often as we were new to C++. Despite these challenges, working through them deepened our understanding of C++ and improved our problem-solving skills — preparing us for more complex programming paradigms.