Himalaya College of Engineering

Advanced C++ Programming Lab Report

Lab 1: Quadratic Equations, Triangle Classification, and Password Strength Checking

Prepared By: Krishnanand Badu
Subject: Object-Oriented Programming (OOP)
Program: Bachelor of Computer Engineer
Institution: Himalaya College of Engineering
Date: May 19, 2025

Objectives

- Understand and apply C++ control structures, functions, and standard libraries.
- Implement mathematical computations using the quadratic formula.
- Develop logic for triangle validation and classification.
- Apply string handling and character checking for password strength.
- Practice working with loops and conditions for prime number generation.

Tools and Libraries Used

- Programming Language: C++
- IDE: Any C++ compiler (G++, Code::Blocks, etc.)
- Libraries: `#include <iostream>`, `#include <cmath>`, `#include <string>`

1 Quadratic Equation Solver

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

Code

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

int main() {
    double a, b, c, discriminant, root1, root2;
    cout << "Enter coefficients a, b, and c: ";
    cin >> a >> b >> c;

    discriminant = b*b - 4*a*c;

    if (discriminant > 0) {
        root1 = (-b + sqrt(discriminant)) / (2*a);
        root2 = (-b - sqrt(discriminant)) / (2*a);
        cout << "Real and distinct roots: " << root1 << " and "
           << root2;
    } else if (discriminant == 0) {
        root1 = -b / (2*a);
        cout << "Real and equal roots: " << root1;
    } else {
        double realPart = -b / (2*a);
        double imagPart = sqrt(-discriminant) / (2*a);
        cout << "Complex roots: " << realPart << "    " <<
           imagPart << "i";
    }

    return 0;
}
```

2 Triangle Classification

Problem: Check if three angles form a triangle and classify it.

Code

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

int main() {
    int angle1, angle2, angle3;
    cout << "Enter three angles: ";
    cin >> angle1 >> angle2 >> angle3;

    int sum = angle1 + angle2 + angle3;

    if (sum == 180 && angle1 > 0 && angle2 > 0 && angle3 > 0) {
        cout << "Valid Triangle. Type: ";
        if (angle1 < 90 && angle2 < 90 && angle3 < 90)
            cout << "Acute Triangle";
        else if (angle1 == 90 || angle2 == 90 || angle3 == 90)
            cout << "Right Triangle";
        else
            cout << "Obtuse Triangle";
    } else {
        cout << "Not a valid triangle.";
    }

    return 0;
}
```

3 Password Strength Checker

Problem: Check password strength based on length and character rules.

Code

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

bool isSpecial(char ch) {
    string special = "!@#$%^&*";
    return special.find(ch) != string::npos;
}

int main() {
    string password;
    cout << "Enter password: ";
    cin >> password;

    bool hasUpper = false, hasLower = false;
    bool hasDigit = false, hasSpecial = false;

    for (char ch : password) {
        if (isupper(ch)) hasUpper = true;
        if (islower(ch)) hasLower = true;
        if (isdigit(ch)) hasDigit = true;
        if (isSpecial(ch)) hasSpecial = true;
    }

    if (password.length() >= 8 && hasUpper && hasLower &&
       hasDigit && hasSpecial) {
        cout << "Strong Password";
    } else {
        cout << "Weak Password. Try adding uppercase, lowercase,
           digits, and special characters.";
    }

    return 0;
}
```

4 Prime Number Generator

Problem: Generate all prime numbers up to a given number n.

Code

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

bool isPrime(int num) {
    if (num < 2) return false;
    for (int i = 2; i*i <= num; i++) {
        if (num % i == 0) return false;
    }
    return true;
}

int main() {
    int n;
    cout << "Enter upper limit: ";
    cin >> n;

    cout << "Prime numbers up to " << n << ":\n";
    for (int i = 2; i <= n; i++) {
        if (isPrime(i))
            cout << i << " ";
    }

    return 0;
}
```

Conclusion

In this lab, we practiced implementing core C++ logic through a variety of practical problems. The use of control structures, functions, and string handling techniques was reinforced. This lab builds a strong foundation for further studies in C++ and object-oriented programming.