ETE 1151 Lab Report: Laboratory #7

Eli Holmes
Class section: 76010
Instructor: Professor Qayyum

Date: October 15, 2024

Objective:

This lab is designed to enforce the concept of functions. We will learn about:

- Writing functions.
- Tracing through a program.
- Passing the parameters to functions.
- Writing functions to return data

****Code comments in **green**

Function 1: Solving a Quadratic Equation

Function Statement:

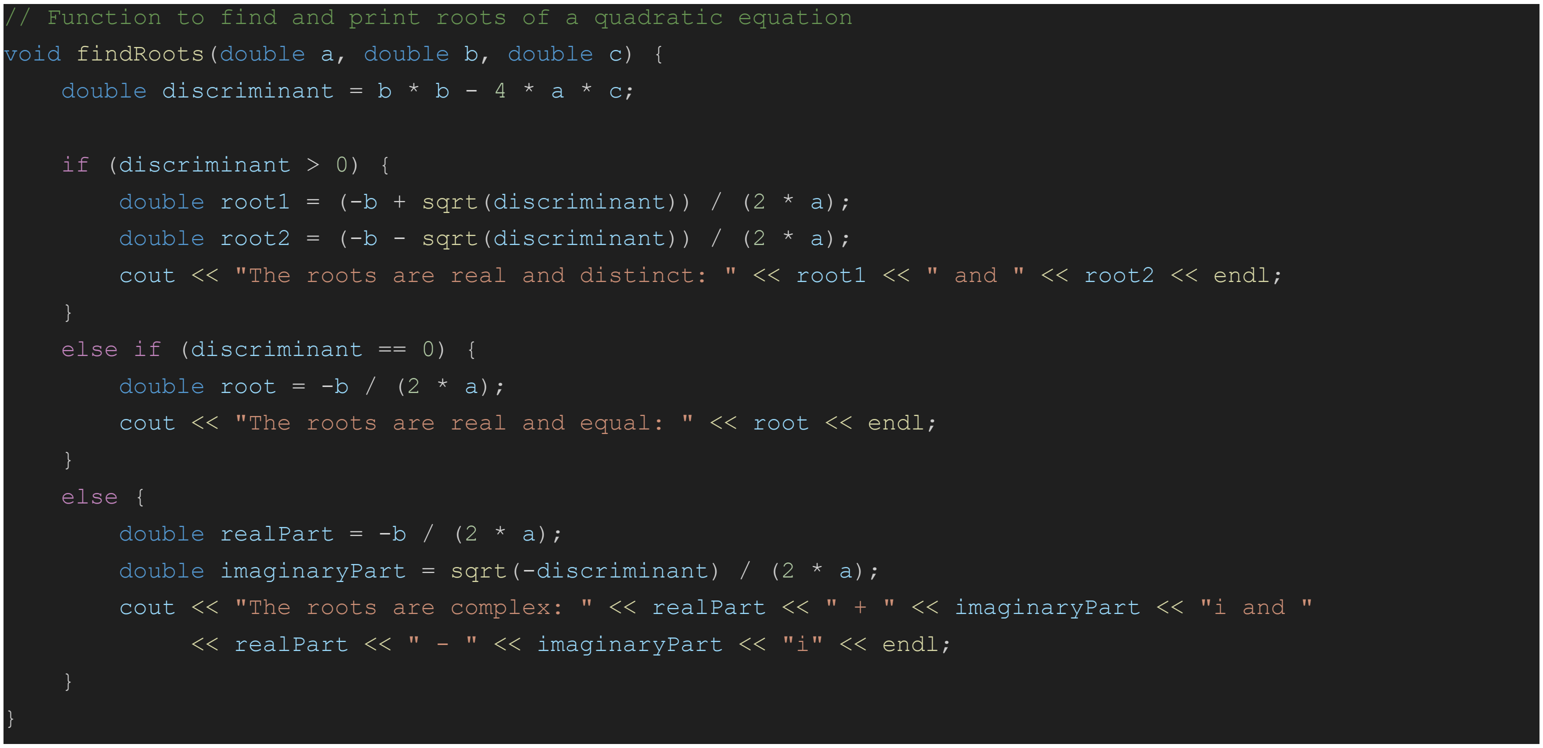
Write a program that inputs three coefficients of a quadratic equation and calls a function to print the roots of the equation. The function should handle and print the roots for all possible cases:

$$\frac{-b\pm\sqrt{b^{2}-4ac}}{2a}$$

1. Two real roots.
2. One real root (repeated).
3. Complex roots.

—-------------------------------------- PROBLEM # 1 SOURCE CODE ---

#include <iostream>
#include <cmath> // for sqrt()
using namespace std;



```cpp
// Function to find and print roots of a quadratic equation
void findRoots(double a, double b, double c) {
    double discriminant = b * b - 4 * a * c;

    if (discriminant > 0) {
        double root1 = (-b + sqrt(discriminant)) / (2 * a);
        double root2 = (-b - sqrt(discriminant)) / (2 * a);
        cout << "The roots are real and distinct: " << root1 << " and " << root2 << endl;
    }
    else if (discriminant == 0) {
        double root = -b / (2 * a);
        cout << "The roots are real and equal: " << root << endl;
    }
    else {
        double realPart = -b / (2 * a);
        double imaginaryPart = sqrt(-discriminant) / (2 * a);
        cout << "The roots are complex: " << realPart << " + " << imaginaryPart << "i and "
             << realPart << " - " << imaginaryPart << "i" << endl;
    }
}
```

—-------------------------------------- PROBLEM # 2 SOURCE CODE ---

Function 2: Evaluating e^x Using Series Expansion

Function Statement:

Write a function to evaluate the value of exe^xex using the series expansion:

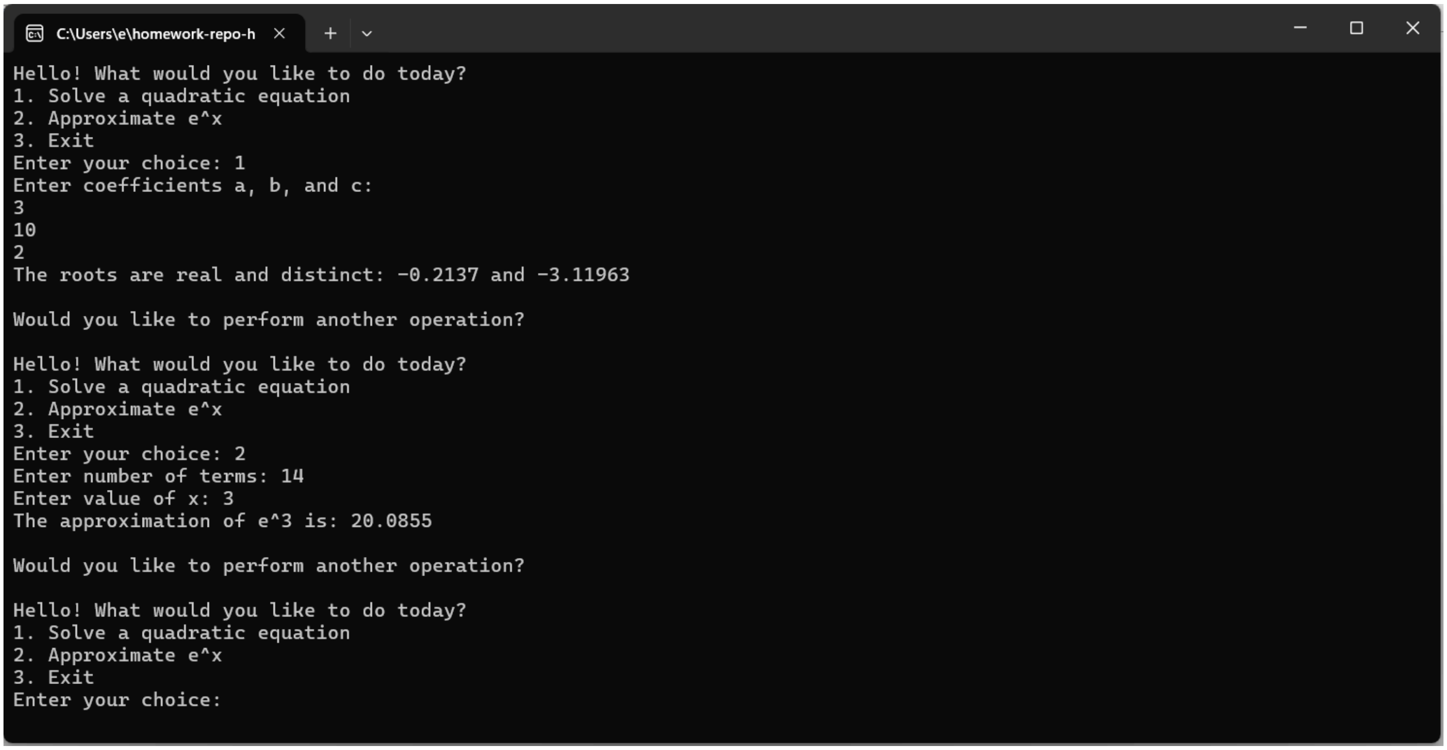
$$e^{x} = 1.0 + \frac{x}{1!} + \frac{x^{2}}{2!} + \frac{x^{3}}{3!} + \cdots + \frac{x^{n}}{n!}$$

The main program should input the number of terms and value of x, pass them to the function, and evaluate e^x. The program should use a sentinel loop to call the function multiple times.

```
// Function to calculate factorial for exponential calculation
double factorial(int n) {
    double fact = 1;
    for (int i = 1; i <= n; ++i) {
        fact *= i;
    }
    return fact;
}

// Function to approximate e^x
double calculateExponential(int numTerms, double x) {
    double result = 1.0;                                            // Start with 1.0 (the first term in the series)

    for (int i = 1; i < numTerms; ++i) {
        result += (pow(x, i) / factorial(i));
    }

    return result;
}
```

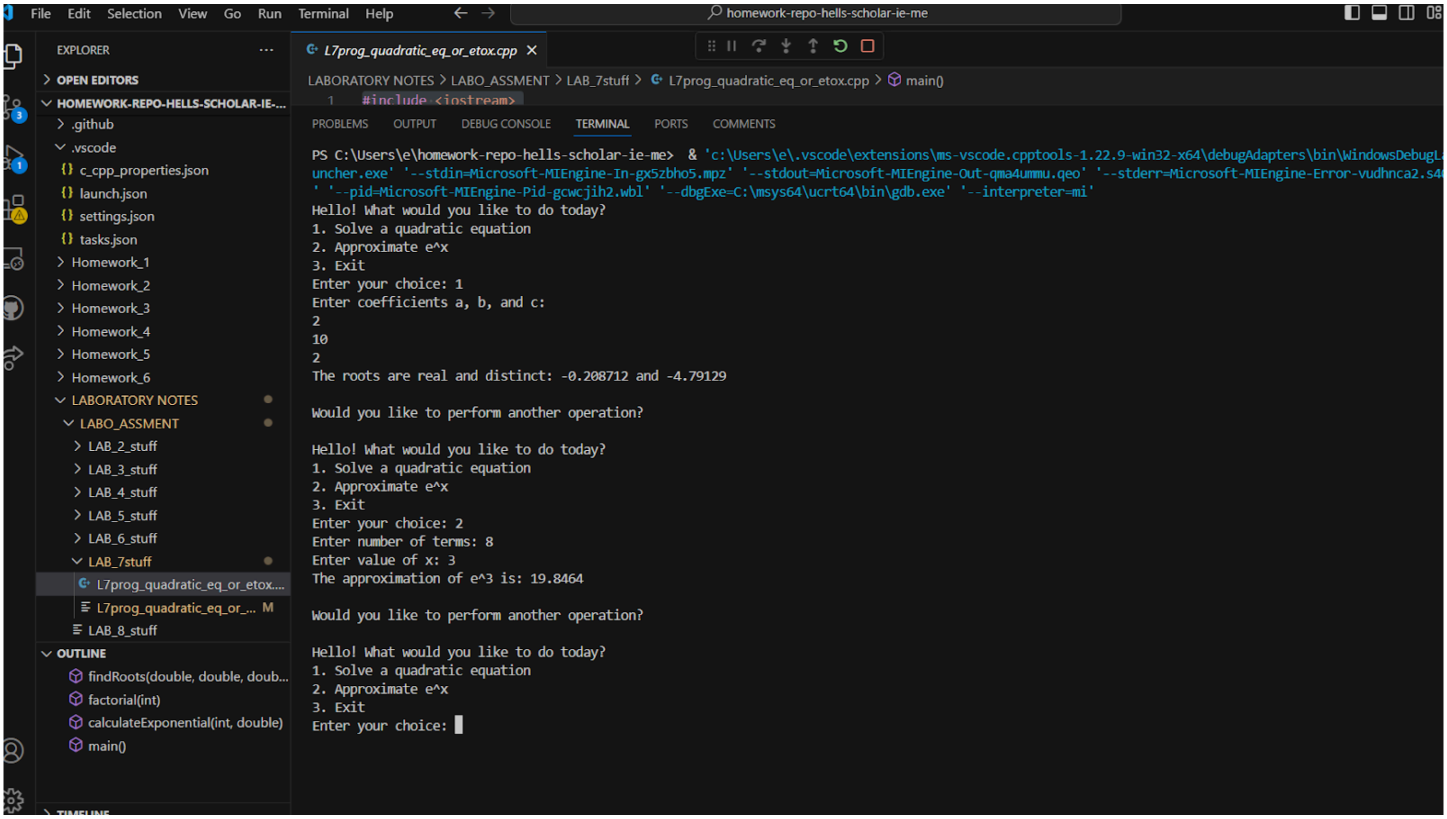
Main Program :

```
int main() {
    int choice;                                                  // Variable to store the user's menu choice
    double a = 0, b = 0, c = 0;                                  // Initialize coefficients a, b, and c in main

    while (true) {                                               // Sentinel loop keeps program running until user exits
        cout << "Hello! What would you like to do today?"        // Prompt the user with a greeting message
             << endl;
        cout << "1. Solve a quadratic equation" << endl;         // Menu option 1: Solve a quadratic equation
        cout << "2. Approximate e^x" << endl;                    // Menu option 2: Approximate e^x
        cout << "3. Exit" << endl;                               // Menu option 3: Exit the program
        cout << "Enter your choice: ";                           // Prompt the user for their menu choice
        cin >> choice;                                           // Read the user's choice from input

        if (choice == 1) {                                       // If user chooses option 1
            // Quadratic equation solver
            cout << "Enter coefficients a, b, and c: " << endl;  // Prompt user to enter the coefficients
            cin >> a >> b >> c;                                  // Read the coefficients from input
            findRoots(a, b, c);                                  // Call the findRoots function to solve the equation

            cout << endl;                                        // Add blank line after result to create spacing
        }
        else if (choice == 2) {                                  // If user chooses option 2
            // Exponential approximation
            int numTerms;                                        // Variable stores number of terms for approximation
            double x;                                            // Variable to store the value of x
            cout << "Enter number of terms: ";                   // Prompt user to enter the number of terms
            cin >> numTerms;                                     // Read the number of terms from input
            cout << "Enter value of x: ";                        // Prompt user to enter the value of x
            cin >> x;                                            // Read the value of x from input
            double result = calculateExponential(numTerms, x);   // Call the calculateExponential function
            cout << "The approximation of e^" << x               // Output the result of the approximation
                 << " is: " << result << endl;                   // Output the result of e^x

            cout << endl;                                        // Add a blank line after the result to create spacing
        }
        else if (choice == 3) {                                  // If user chooses option 3
            // Exit the program
            cout << "Goodbye!" << endl;                          // Output a goodbye message
            break;                                               // Exit the loop, ending the program
        }
        else {                                                   // If the user enters an invalid choice
            cout << "Invalid choice, please try again."          // Output an error message
                 << endl;                                        // Move to a new line
        }

        cout << "Want to do another operation?" << endl;         // Prompt for another operation
        cout << endl;                                            // Add a blank line for additional spacing between
iterations
    }

    return 0;                                                    // Return 0 to indicate successful program termination
}
```



```
Hello! What would you like to do today?
1. Solve a quadratic equation
2. Approximate e^x
3. Exit
Enter your choice: 1
Enter coefficients a, b, and c:
3
10
2
The roots are real and distinct: -0.2137 and -3.11963

Would you like to perform another operation?

Hello! What would you like to do today?
1. Solve a quadratic equation
2. Approximate e^x
3. Exit
Enter your choice: 2
Enter number of terms: 14
Enter value of x: 3
The approximation of e^3 is: 20.0855

Would you like to perform another operation?

Hello! What would you like to do today?
1. Solve a quadratic equation
2. Approximate e^x
3. Exit
Enter your choice:
```

Screen shot of external windows terminal



```
#include <iostream>
```

```
PS C:\Users\e\homework-repo-hells-scholar-ie-me>  & 'c:\Users\e\.vscode\extensions\ms-vscode.cpptools-1.22.9-win32-x64\debugAdapters\bin\WindowsDebugL
uncher.exe' '--stdin=Microsoft-MIEngine-In-gx5zbho5.mpz' '--stdout=Microsoft-MIEngine-Out-qma4ummu.qeo' '--stderr=Microsoft-MIEngine-Error-vudhnca2.s4
' '--pid=Microsoft-MIEngine-Pid-gcwcjih2.wbl' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
Hello! What would you like to do today?
1. Solve a quadratic equation
2. Approximate e^x
3. Exit
Enter your choice: 1
Enter coefficients a, b, and c:
2
10
2
The roots are real and distinct: -0.208712 and -4.79129

Would you like to perform another operation?

Hello! What would you like to do today?
1. Solve a quadratic equation
2. Approximate e^x
3. Exit
Enter your choice: 2
Enter number of terms: 8
Enter value of x: 3
The approximation of e^3 is: 19.8464

Would you like to perform another operation?

Hello! What would you like to do today?
1. Solve a quadratic equation
2. Approximate e^x
3. Exit
Enter your choice: 
```

A screenshot from my local visual studio test run

Explanation:

This program uses a loop to evaluate the exponential function e^x by adding terms of the series expansion up to the specified number of terms. The user inputs both x and the number of terms, and the program calculates the series incrementally using a loop. The loop continues until the user inputs `-1` to exit.

Summary:

This lab helped me understand fundamental C++ programming concepts, particularly function use, parameter passing, and data return. The primary focus was on mastering function creation, which is essential for modular programming and allows for better code organization and reuse. Additionally, I explored how to trace through a program's flow, pass parameters to functions, and return results, which are critical skills for efficient problem-solving in more complex programs.

In **Program 1**, I worked with quadratic equations, exploring how to calculate and print their roots. This program reinforced my knowledge of conditional statements and the use of the discriminant to determine whether the roots are real, repeated, or complex. The challenge in this problem was handling multiple cases (distinct real roots, repeated real roots, or complex roots) using if-else logic. By leveraging mathematical functions like `sqrt()` from the C++ library, I could provide accurate results for each case, solidifying my understanding of quadratic equations and how to implement mathematical formulas in code.

In **Program 2**, I tackled the evaluation of exe^xex using a series expansion. This problem involved working with loops and floating-point arithmetic to ensure accurate results across several iterations of the expansion. One of the key aspects of this task was managing the precision of real numbers and ensuring that each term in the expansion was calculated correctly. Additionally, I implemented a sentinel loop to allow the user to input different values of xxx and the number of terms, which reinforced my ability to create flexible and reusable code. Handling floating-point arithmetic was critical here, as small inaccuracies in earlier terms could affect the overall result.

Overall, this lab allowed me to refine my technical coding and problem-solving skills. I encountered minor issues, such as debugging syntax errors and ensuring the accuracy of floating-point arithmetic, which provided valuable learning experiences. These small challenges underscored the importance of precision and attention to detail in coding. Completing this lab has strengthened my confidence in using C++ for practical applications, particularly in mathematical operations, user input, and function-driven design scenarios. These skills will be essential as I progress to more advanced courses and projects.