ENGG1410: Introductory Programming for Engineers

Lab 3: "Introduction to C Programming": More on Decision Making and Debugging a C Program

School of Engineering, University of Guelph
Fall 2024

Start Date: Week #4 2024
Duration: 1 Week

Objectives

The objective of this lab is to enhance your programming skills by writing C programs implementing decision-making using `if-else` statements via programs that involve reversing numbers, solving quadratic equations, simulating simple traffic light control system and in the second part to practice debugging programs

Instructions

Follow the instructions for each question. You are required to compile and test your programs. Submit your source code and screenshots of your program output via the learning management system. Once again, place your solution for each part in a separate directory, so that Visual Studio Code will not attempt to compile multiple files and link them together. Each of your directories should have one .c file corresponding to your solution for the individual part of the exercise. Please ensure you are selecting the folder with the .c you want to compile.

Part 1 Programming

Question 1: Solving a Quadratic Equation

Write a C program that finds the roots of a quadratic equation of the form:

$$ax^2 + bx + c = 0$$

The program should prompt the user to enter the coefficients a, b, and c, and it should determine the nature of the roots (whether they are real and distinct, real and equal, or imaginary) and calculate them using the equation

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Program Requirements

- The program should prompt the user to input the values of a, b, and c. - If user enters zero for the value of a, print on screen Invslid input as the program is intended to solve quadrtic equations. - It should calculate the discriminant $D = b^2 - 4ac$. - Based on the value of the discriminant:

- If $D > 0$, the program should print that the roots are real and distinct, and calculate both roots and prints their values.
- If $D = 0$, the program should print that the roots are real and equal, and calculate the single root, and print their values.
- If $D < 0$, the program should print that the roots are imaginary.

  Note: C programming does not natively support complex or imaginary numbers in its standard library. However, starting from the C99 standard, C includes support for complex numbers through the complex.h library.

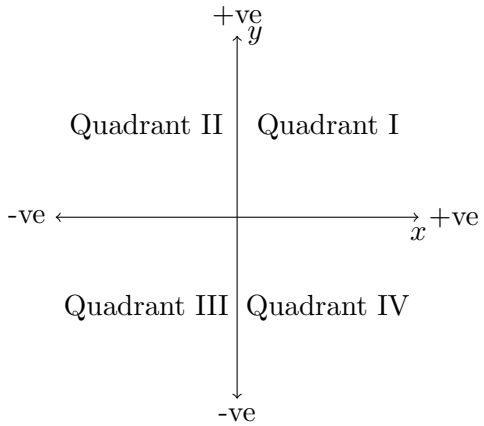
  This library allows you to handle complex numbers, including calculations involving imaginary components. However, this is not to intended to be covered at this time of the course.

Sample Output

```
Enter coefficients a, b, and c: 1 -3 2
The roots are real and distinct.
Root 1 = 2.00
Root 2 = 1.00
```

Question 2: Cartesian Coordinates

Write a C program that takes the $x - y$ coordinates of a point in the Cartesian plane, and prints a message telling either an axis on which the point lies, or the quadrant in which it is found. If the point is at (0.00, 0.00), the program should report that it is at the origin. The floating point coordinates are entered by the user. When coordinates are printed and used to determine the quadrant, they are rounded to the nearest hundredth. For example, 4.866 should be rounded to 4.87, and 18.953 should be rounded to 18.95, then these rounded numbers should be used to determine the quadrant. If the input was (0.0001, 0.001), the rounded numbers would be (0.00, 0.00), therefore the reported quadrant should be at the origin. The following figure shows the Cartesian plane and the quadrants.



Here is a sample output from an execution of the program:

```
Enter the x-coordinate in floating point: 0.0<enter>
Enter the y-coordinate in floating point: -2.5<enter>
(0.00, -2.50) is on the y axis.
```

Here is a second example:

```
Enter the x-coordinate in floating point: 4.556<enter>
Enter the y-coordinate in floating point: 4.864<enter>
(4.56, 4.86) is in quadrant I.
```

Here is a third example:

```
Enter the x-coordinate in floating point: 0.0001<enter>
Enter the y-coordinate in floating point: 0.001<enter>
(0.00, 0.00) is at the origin.
```

Note: You can assume that the user enters valid floating point numbers within the range allowed by the double type.

Part 2: Debugging

Question 3: Simulating a Dice Roll Game (Using Random Numbers)

The following C program simulates a simple dice-rolling game. The player rolls two dice, and the sum of the two dice determines whether they win, lose, or continue playing. However, the program contains several errors. Your task is to debug the program and ensure it works correctly.

Code to Debug

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main() {
    int dice1, dice2, sum;
    char play_again;

    srand(); // Seed the random number generator

    dice1 == rand() % 6;
    dice2 = rand() % 6;

    sum = dice1 + dice2;
```

```
    printf("You rolled a %d and a %d.\n", dice1, dice2);
    printf("The sum is %d.\n", sum);

    if (sum == 7 || sum == 11) {
        printf("You win!\n");
    } else if (sum == 2 || sum == 3 || sum = 12) {
        printf("You lose!\n");
    } else {
        printf("No win or loss, play again.\n");
    }

    return 0;
}
```

Expected Output (After Debugging)

```
You rolled a 3 and a 4.
The sum is 7.
You win!
```

Question 4: Calculating letter grade from student marks

The following C program is designed to calculate the final score of a student based on multiple inputs: quiz marks, assignment marks, and exam marks. The program applies a bonus to the calculated score and determines the student's grade. However, the program contains multiple compile-time, run-time, and logical errors. Your task is to identify and correct the errors.

Code to Debug

```
#include <stdio.h>

int main() {
    int quiz1, quiz2, assignment, exam;
    double weighted_average, final_score, bonus;
    char grade;

    printf("Enter marks for Quiz 1 (out of 20): ");
    scanf("%d", quiz1); // Error: Missing &

    printf("Enter marks for Quiz 2 (out of 20): ");
    scanf("%d", quiz2);

    printf("Enter marks for the Assignment (out of 50): ");
    scanf("%d", &assignment);

    printf("Enter marks for the Exam (out of 100): ");
    scanf("%d", &exam);
```

```
    weighted_average = (quiz1 + quiz2) / 2 * 0.2 + assignment * 0.3 + exam * 0.5;

    bonus = ceil(weighted_average * 0.1); // Bonus calculation

    final_score = weighted_average + bonus;

    printf("Weighted Average: %.2f\n", weighted_average);
    printf("Bonus: %.2f\n", bonus);
    printf("Final Score: %.2f\n", final_score);

    if (final_score >= 90) {
        grade = "A"; // Error: Incorrect use of double quotes for char
    } else if (final_score >= 80 && final_score < 90) {
        grade = 'B';
    } else if (final_score >= 70 && final_score < 80) {
        grade = 'C';
    } else if (final_score >= 60 && final_score < 70) {
        grade = 'D';
    } else {
        grade = 'F';
    }

    printf("Your final grade is: %c\n", grade);

    return 0;
}
```

Expected Output (After Debugging)

```
Enter marks for Quiz 1 (out of 20): 15
Enter marks for Quiz 2 (out of 20): 18
Enter marks for the Assignment (out of 50): 40
Enter marks for the Exam (out of 100): 85
Weighted Average: 73.10
Bonus: 8.00
Final Score: 81.10
Your final grade is: B
```