Allowed Time: 60 Minutes | Total Marks: 30 + 10 Marks

Instructions:

1. Gossips are not allowed.
2. Teacher assistants are for your help, so be nice with them. Respect them as they are teaching you. Raise your hands if you have some problem and need help from TA. Avoid calling them by raising your voice and disturbing the environment of Lab.
3. TA may deduct your marks for any kind of ill-discipline or misconduct from your side.
4. Evaluation will be considered final and you cannot debate for the marks. So, focus on performing the tasks when the time is given to you.

Task 01: (10 Marks, 20 min)

Write a C program to simulate a **Wheel of Fortune** game. The program should:

1. Allow the player to spin a virtual wheel up to 3 times.
2. Generate a random prize between $100 and $1000 for each spin.
3. Keep track of the highest prize won.
4. Ask the player after each spin if they want to spin again (up to 3 spins).
5. Display the final prize at the end of the game.

The program should use a **loop** to handle the spinning logic and modularize the code using functions:

- `int` **`spinWheel():`** Generates and returns a random prize.
- `void` **`playWheelOfFortune():`** Implements the game logic using a loop.

Requirements:

1. Implement the **`spinWheel()`** function to generate a random prize between $100 and $1000.
2. Implement the **`playWheelOfFortune()`** function to manage the gameplay, including:
   - Keeping track of the number of spins.
   - Asking the player if they want to continue spinning.
   - Keeping track of the highest prize won.
3. Ensure the program handles user input and displays results clearly.

Example Output:

```
=== Wheel of Fortune ===
You can spin the wheel up to 3 times. Your highest prize will be your final reward!

Spin 1: You won $250!
Do you want to spin again? (y/n): y

Spin 2: You won $600!
Do you want to spin again? (y/n): n

Congratulations! Your final prize is $600.
```

Task 02: (20 Marks, 30 min)

Problem Statement:

Write a C program to solve a quadratic equation of the form **ax^2+ bx + c =0.** The program should take the coefficients a, b, and c as input and determine the roots of the equation. The program should handle the following cases:

1. **Two real and distinct roots** (when the discriminant is positive).

2. **One real root (repeated)** (when the discriminant is zero).
3. **Complex conjugate roots** (when the discriminant is negative).

The program should use **functions** to modularize the code. Specifically, you should implement the following functions:

1. **`double calculateDiscriminant(double a, double b, double c):`** Calculates and returns the discriminant.
2. **`void printRealDistinctRoots(double a, double b, double discriminant):`** Calculates and prints two real and distinct roots.
3. **`void printRealRepeatedRoot(double a, double b):`** Calculates and prints one real root (repeated).
4. **`void printComplexConjugateRoots(double a, double b, double discriminant):`** Calculates and prints complex conjugate roots.
5. **`void solveQuadraticEquation(double a, double b, double c):`** Determines the nature of the roots based on the discriminant and calls the appropriate function to calculate and print the roots.

Quadratic Formula:

The roots of a quadratic equation $ax^2 + bx + c = 0$ are given by the quadratic formula:

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

Where:

- a, b, and c are coefficients of the quadratic equation.
- $b^2 - 4ac$ is called the **discriminant** (denoted as D).

Cases:

1. **Two real and distinct roots** ($D > 0$):

$$\text{Root 1} = \frac{-b + \sqrt{D}}{2a}, \quad \text{Root 2} = \frac{-b - \sqrt{D}}{2a}$$

2. **One real root (repeated)** ($D = 0$):

$$\text{Root} = \frac{-b}{2a}$$

3. **Complex conjugate roots** ($D < 0$):

$$\text{Root 1} = \frac{-b}{2a} + \frac{\sqrt{-D}}{2a}i, \quad \text{Root 2} = \frac{-b}{2a} - \frac{\sqrt{-D}}{2a}i$$

- In the main function:
  - Prompt the user to input the coefficients aa, bb, and cc.
  - Call the **solveQuadraticEquation()** function to solve the equation.
- Handle invalid input gracefully (e.g., if a=0 the equation is not quadratic).

Example Outputs:

Case 1: Two Real and Distinct Roots

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

Case 2: One Real Root (Repeated)

```
Enter coefficients a, b, and c: 1 2 1
One real root (repeated):
Root = -1.00
```

Case 3: Complex Conjugate Roots

```
Enter coefficients a, b, and c: 1 2 5
Complex conjugate roots:
Root 1 = -1.00 + 2.00i
Root 2 = -1.00 - 2.00i
```

Sample Code Structure:

```c
#include <stdio.h>
#include <math.h>

// Function prototypes
double calculateDiscriminant(double a, double b, double c);
void printRealDistinctRoots(double a, double b, double discriminant);
void printRealRepeatedRoot(double a, double b);
void printComplexConjugateRoots(double a, double b, double discriminant);
void solveQuadraticEquation(double a, double b, double c);

int main() {
    // Your code here
    return 0;
}

// Function definitions
double calculateDiscriminant(double a, double b, double c) {
    // Your code here
}

void printRealDistinctRoots(double a, double b, double discriminant) {
    // Your code here
}

void printRealRepeatedRoot(double a, double b) {
    // Your code here
}

void printComplexConjugateRoots(double a, double b, double discriminant) {
    // Your code here
}

void solveQuadraticEquation(double a, double b, double c) {
    // Your code here
}
```

Task 03: (BONUS)
(10 Marks, 10 min)

Question Statement:

Write a **menu-based C program** that allows the user to choose between two tasks:

1. **Play Wheel of Fortune** (Task 01)
2. **Solve a Quadratic Equation** (Task 02)

Requirements:

1. Display a menu to the user with the following options:

```
Welcome to the Task Manager System.
Choose a task:
1. Wheel of Fortune
2. Quadratic Equation Solver
Enter your choice (1 or 2): 1

=== Wheel of Fortune ===
You can spin the wheel up to 3 times. Your highest prize will be your final reward!

Spin 1: You won $100!
Do you want to spin again? (y/n): y

Spin 2: You won $500!
Do you want to spin again? (y/n): n

Congratulations! Your final prize is $500.

Do you want to return to the main menu? (y/n): y

Welcome to the Game Center!
Choose a task:
1. Wheel of Fortune
2. Quadratic Equation Solver
Enter your choice (1 or 2): 2

=== Quadratic Equation Solver ===
Enter coefficients a, b, and c: 1 -3 2
Two real and distinct roots:
Root 1 = 2.00
Root 2 = 1.00

Do you want to return to the main menu? (y/n): n
Goodbye! See you next time!
```

2. Based on the user's input:
    - If the user selects **1**, execute the Wheel of Fortune game.
    - If the user selects **2**, solve a quadratic equation.
3. Handle invalid inputs gracefully (e.g., if the user enters an invalid option).
4. Continue displaying the menu until the user chooses to exit.
5. Use appropriate function calls for both tasks:
    - **`playWheelOfFortune()`** for Task 01
    - **`solveQuadraticEquation()`** for Task 02