Assignment 2: Computer Engineering Case Study – Rock Paper Scissors Game

Step 1: Problem Identification and Statement

The point of this assignment is to create a program that allows a player to play the Rock Paper Scissors (RPS) game versus a computer. The program should let the player choose between Rock, Paper, or Scissors, with the computer randomly selecting one of the three possibilities. The game will be played exactly five times, with the program displaying the results of each round, including any ties. After 5 plays, the program should show a summary of the results, including whether the player won, lost, or tied with the computer's score.

Step 2: Gathering Information and Input/Output Description

Relevant Information:

Rock Paper Scissors (RPS) is an intransitive hand game, usually played between two people, in which each player simultaneously forms one of three shapes with an outstretched hand. These shapes are "Rock", "Paper", and "Scissors". It has three possible outcomes: a draw, a win, or a loss. The rules are as follows:

- *Scissors beats Paper*
- *Paper beats Rock*
- *Rock beats Scissors*
- *If both players select the same choice, the game is tied.*

In this assignment, the player plays against a computer. This program will:

- Allow the player to choose between Rock, Paper, or Scissors for each round.
- Get the computer's choice: For each round, the computer will randomly select one of three possibilities (Rock, Paper, or Scissors).
- Play the game five times: The game will consist of exactly five rounds. If a round ends in a tie, it will be played again until a winner is determined. Tied rounds do not count towards the five rounds.
- Show the results for each round: Following each round, the program displays:
  The player's decision.
  The computer's choice.
  The results of the round (win, loss, or tie)

- develop a game summary: After 5 rounds, the program generates a summary table with all of the round results. It will also show whether the player won, lost, or tied against the computer based on the amount of wins and losses.

Input/output Description (I/O Diagram) :

Input:

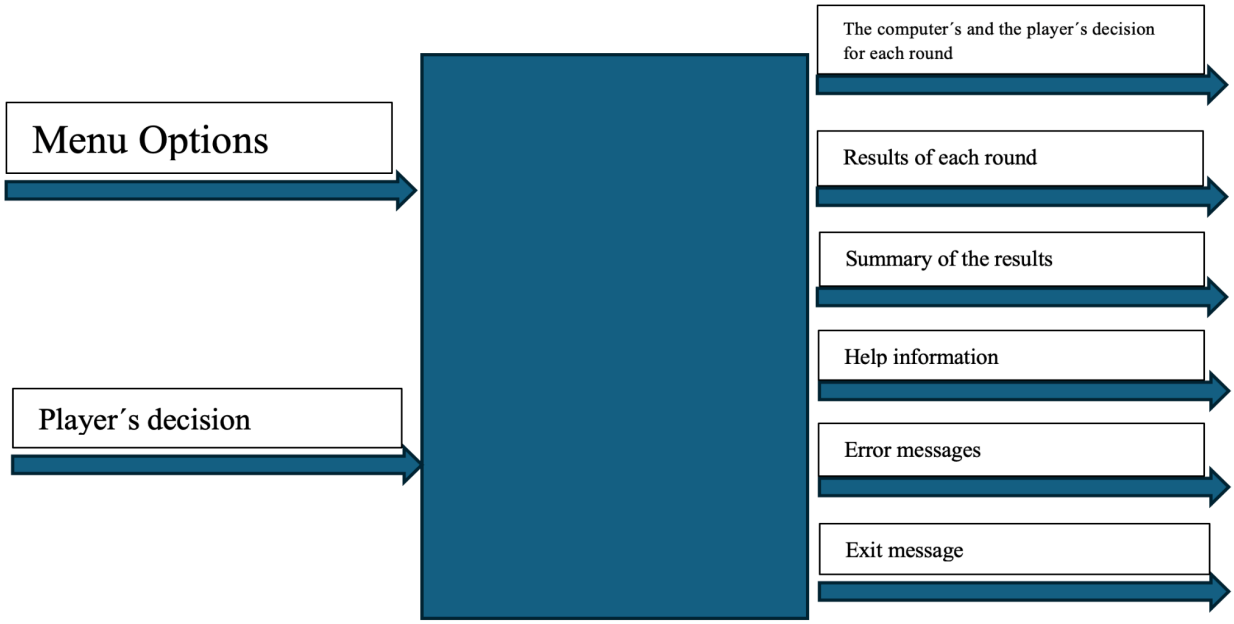
1.User input

a) The player's decision for each round (1 for rock, 2 for paper, and 3 for scissors).
b) Menu options (1 for starting the game, 2 for Help, 3 for Exiting the program)

Output:

1. The player´s and the computer´s decisions for each round
2. The result of each round: win, loss, tie
3. The summary of results after 5 games displaying whether the player won, lost, or tied against the computer.
4. Help information : the player can see the description of the game if the player selects this option
5. Error messages : displaying an error message for invalid input
6. Exit Message: displayed when terminating the program

I/O Diagram:



Step 3: Test Cases and algorithm

In this section, we show how to check the game's logic and results using manual computations rather than code execution, by having the user inputs and expecting the output.

Test Case	User Inputs	Expected Output
User Wins the Game	Menu option : choose 1 (Play). the user got winning moves in the majority of rounds.	program shows the user´s and the computer´s decisions. At the end, the summary will show more wins for the user and displays: "You win!"
The Computer Wins the Game	User got moves resulting in more computer wins.	program shows the user´s and the computer´s decisions and displays: "You lose!"
Tie Handling	User got moves that cause a tie. Round will be replayed until resolution.	program will output "Tie! Please select again." and replay until the resolution
Handling Invalid Input	User types invalid inputs (., 0, 4, letters...).	the program displays: "Invalid input, please try again."
Help Option	User chooses Help from the menu.	Game instructions are displayed before returning to the main menu.
Exit Option	User selects Exit from menu.	Displays: "Terminating... Program ended successfully."

Algorithm Design:

<u>Pseudo code</u>

FUNCTION print_Menu() RETURNS INTEGER

PRINT "1) Play", newline

PRINT "2) Help", newline

PRINT "3) Exit", newline

PRINT "Please make a selection (1-3): "

INPUT selection

RETURN selection

END FUNCTION

```
FUNCTION play_RPS(player[], computer[], size)
    DECLARE Plays[3] AS ARRAY OF STRINGS = ["Rock", "Paper", "Scissors"]
    FOR i FROM 0 TO size - 1 DO
        WHILE TRUE DO
            PRINT "Round " + (i + 1) + ". Please select among 1) Rock, 2) Paper, or 3) Scissors: "
            INPUT player[i]
            IF player[i] < 1 OR player[i] > 3 THEN
                PRINT " This is an Invalid input! Please enter one of the following: 1, 2, or 3.", newline
                CONTINUE
            END IF
            computer[i] = RANDOM(1, 3)  // generate computer's choice in a random way
            PRINT "You: " + Plays[player[i] - 1] + " Computer: " + Plays[computer[i] - 1] , newline

            IF player[i] == computer[i] THEN
                PRINT "Tie! Please select again.", newline
            ELSE
                BREAK
            END IF
        END WHILE
    END FOR
END FUNCTION
```

```
FUNCTION print_Summary(player[], computer[], size)
    DECLARE Plays[3] AS ARRAY OF STRINGS = ["Rock", "Paper", "Scissors"]
    DECLARE victory_count = 0, loss_count = 0
    PRINT "| Round | You | Computer | Result |", newline
    FOR i FROM 0 TO size - 1 DO
        DECLARE result
        IF player[i] == computer[i] THEN
            result = "Tie"
        ELSE IF (player[i] == 1 AND computer[i] == 3) OR
                (player[i] == 2 AND computer[i] == 1) OR
                (player[i] == 3 AND computer[i] == 2) THEN
            result = "Win"
            INCREMENT victory_count
        ELSE
            result = "Lose"
            INCREMENT loss_count
        END IF
        PRINT "| " + (i + 1) + " | " + Plays[player[i] - 1] + " | " + Plays[computer[i] - 1] + " | " + result +
" |", newline
    END FOR
    IF victory_count > loss_count THEN
        PRINT "You win! (win: " + victory_count + ", lose: " + loss_count + ")", newline
```

```
    ELSE IF victory_count < loss_count THEN
        PRINT "You lose! (win: " + victory_count + ", lose: " + loss_count + ")", newline
    ELSE
        PRINT "It's a tie! (win: " + victory_count + ", lose: " + loss_count + ")", newline
    END IF
END FUNCTION


FUNCTION main()
    DECLARE player[5], computer[5] AS ARRAY OF INTEGERS
    SEED_RANDOM(time(0))  // Seed the random number generator with the current time
    WHILE TRUE DO
        DECLARE selection = print_Menu()
        IF selection == 1 THEN
            CALL play_RPS(player, computer, 5)
            CALL print_Summary(player, computer, 5)
        ELSE IF selection == 2 THEN
            PRINT "This is a Rock Paper Scissors game against a computer. You will play 5 rounds then see
the results.", newline
        ELSE IF selection == 3 THEN
            PRINT "Exiting the game ...", newline
            BREAK
        ELSE
            PRINT "Invalid input. Please retry.", newline
```

END IF

END WHILE

END FUNCTION

Step 4: Code or implementation

```
#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

int print_Menu() {
    int selection;
    cout << "1) Play" << endl;
    cout << "2) Help" << endl;
    cout << "3) Exit" << endl;
    cout << "Please make a selection (1-3): ";
    cin >> selection;
    return selection;
}

void play_RPS(int player[], int computer[], int size) {
    // Array to turn options (1, 2, 3) to "Rock", "Paper",
"Scissors"
    string Plays[3] = {"Rock", "Paper", "Scissors"};
    for (int i = 0; i < size; i++) {
        while (true) {
            cout << "Round " << i + 1 << ". Please select among
1) Rock, 2) Paper, or 3) Scissors: ";
            cin >> player[i];
            // Input validation
            if (player[i] < 1 || player[i] > 3) {
                cout << "This is an Invalid input! Please enter
one of the following: 1, 2, or 3." << endl;
                continue; // Start over after skipping the rest
of the loop.
            }
```

```
            computer[i] = rand() % 3 + 1; //to generate the
computer's choice (1, 2, or 3) in a random way
            // Use the Plays array to show the moves taken by
the computer and the user.
            cout << "You: " << Plays[player[i] - 1] << "
Computer: " << Plays[computer[i] - 1] << endl; // array indices
start at 0, but the user's options start at 1. so we decrement
            if (player[i] == computer[i]) {
                cout << "Tie! Please select again." << endl;
            } else {
                break;
            }
        }
    }
}

void print_Summary(int player[], int computer[], int size) {
    // Array to turn options (1, 2, 3) to "Rock", "Paper",
"Scissors"
    string Plays[3] = {"Rock", "Paper", "Scissors"};
    int victory_count = 0;
    int loss_count = 0;
    for (int i = 0; i < size; i++) {
        // Find out the round's result
        if (player[i] == computer[i]) {
            // Ties do not count as wins or losses.
        } else if ((player[i] == 1 && computer[i] == 3) ||  //
Rock beats Scissors
                   (player[i] == 2 && computer[i] == 1) ||  //
Paper beats Rock
                   (player[i] == 3 && computer[i] == 2)) {  //
Scissors beats Paper
            victory_count++; // user wins
        } else {
            loss_count++; // user loses
        }
    }

    // Printing the summary tabular at the end
    cout << "| Round | You | Computer | Result |" << endl;
    for (int i = 0; i < size; i++) {
```

```
        string result;
        if (player[i] == computer[i]) {
            result = "Tie";
        } else if ((player[i] == 1 && computer[i] == 3) ||  //
Rock beats Scissors
                   (player[i] == 2 && computer[i] == 1) ||  //
Paper beats Rock
                   (player[i] == 3 && computer[i] == 2)) {  //
Scissors beats Paper
            result = "Win";
        } else {
            result = "Lose";
        }
        cout << "| " << i + 1 << " | " << Plays[player[i] - 1]
<< " | " << Plays[computer[i] - 1] << " | " << result << " |" <<
endl;
    }

    // displaying the last result
    if (victory_count > loss_count) {
        cout << "You win! (win: " << victory_count << ", lose: "
<< loss_count << ")" << endl;
    } else if (victory_count < loss_count) {
        cout << "You lose! (win: " << victory_count << ", lose:
" << loss_count << ")" << endl;
    } else {
        cout << "It's a tie! (win: " << victory_count << ",
lose: " << loss_count << ")" << endl;
    }
}

int main() {
    srand(time(0)); // Initialise the random number generator.
    int player[5];
    int computer[5];
    while (true) {
        int selection = print_Menu();
        if (selection == 1) {
            play_RPS(player, computer, 5);
            print_Summary(player, computer, 5);
        } else if (selection == 2) {
```

```
            cout << " This is a Rock Paper Scissors game against
a computer. You will play 5 rounds then see the results." <<
endl;
        } else if (selection == 3) {
            cout << "Exiting the game ..." << endl;
            break;
        } else {
            cout << "Invalid input. Please retry." << endl;
        }
    }
    return 0;
}
```

Step 5: Test and Verification

To test if the program is successful and working, we will run the program and try each test case

Test Case 1: User wins the game

```
Round 5. Please select among 1) Rock, 2) Paper, or 3) Scissors: 2
You: Paper Computer: Rock
| Round | You | Computer | Result |
| 1 | Scissors | Rock | Lose |
| 2 | Scissors | Paper | Win |
| 3 | Scissors | Rock | Lose |
| 4 | Scissors | Paper | Win |
| 5 | Paper | Rock | Win |
You win! (win: 3, lose: 2)
1) Play
2) Help
3) Exit
Please make a selection (1-3):
```

No issues

Test Case 2: The computer wins the game

```
Round 5. Please select among 1) Rock, 2) Paper, or 3) Scissors: 1
You: Rock Computer: Paper
| Round | You | Computer | Result |
| 1 | Rock | Paper | Lose |
| 2 | Rock | Scissors | Win |
| 3 | Rock | Paper | Lose |
| 4 | Rock | Scissors | Win |
| 5 | Rock | Paper | Lose |
You lose! (win: 2, lose: 3)
1) Play
2) Help
3) Exit
Please make a selection (1-3):
```

No issues

Test Case 3: Tie handling

```
Round 4. Please select among 1) Rock, 2) Paper, or 3) Scissors: 2
You: Paper Computer: Paper
Tie! Please select again.
Round 4. Please select among 1) Rock, 2) Paper, or 3) Scissors: 2
You: Paper Computer: Paper
Tie! Please select again.
Round 4. Please select among 1) Rock, 2) Paper, or 3) Scissors:
```

No issues

Test Case 4: handling invalid input

For main menu :

```
Please make a selection (1-3): 6
Invalid input. Please retry.
1) Play
2) Help
3) Exit
Please make a selection (1-3):
```

No issues

For choices menu :

```
Round 4. Please select among 1) Rock, 2) Paper, or 3) Scissors: 7
This is an Invalid input! Please enter one of the following: 1, 2, or 3.
Round 4. Please select among 1) Rock, 2) Paper, or 3) Scissors:
```

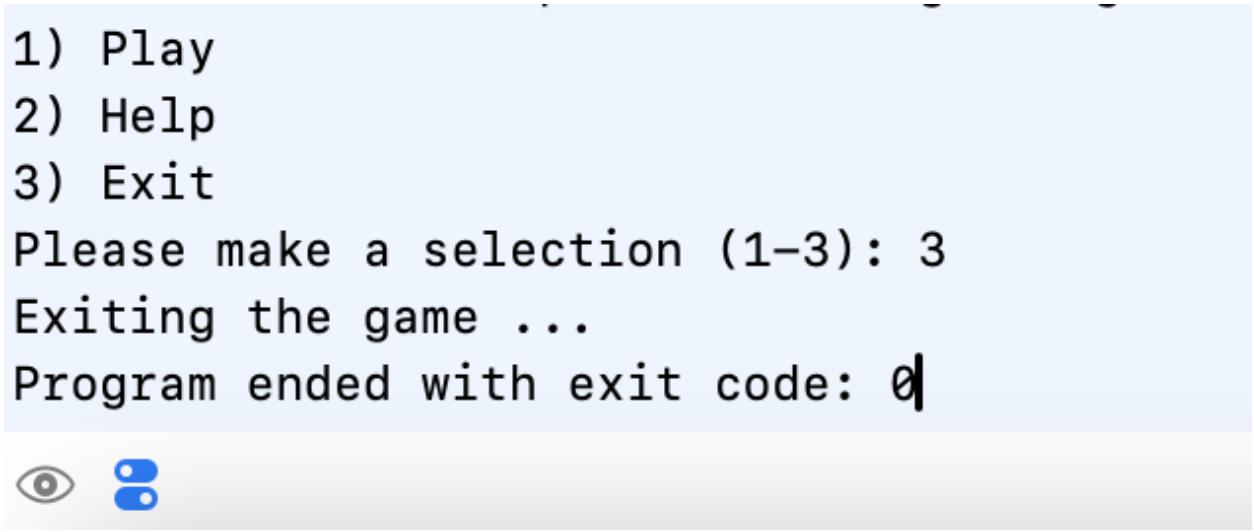
No issues

Test Case 5: Help Option

```
1) Play
2) Help
3) Exit
Please make a selection (1-3): 2
 This is a Rock Paper Scissors game against a computer. You will play 5 rounds then see the results.
1) Play
2) Help
3) Exit
Please make a selection (1-3):
```

No issues

Test Case 6: Exit Option



No issues

User guide:

1. Beginning:

To begin playing, launch the application. A menu containing the following choices will appear:

1. *Play*: begin a Rock, Paper, Scissors game.
2. *Help*: View guidelines and helpful game information
3. Exit: Exiting the program

2. The actual game:

• The game will start automatically after you select to play;
• You will be playing against the computer in five rounds;
• You will be asked to select between Rock, Paper, or Scissors for each round by entering the corresponding number (1 for Rock, 2 for Paper, and 3 for Scissors);
• The computer will then choose its option at random.
• The traditional Rock, Paper, Scissors rules will be used to determine the winner of each round:
1. Scissors lose against Rock
2. Paper loses to scissors
3. Rock loses to paper.
• The round will conclude in a *tie* if you and the computer select the identical option.

3. Displaying the Game Summary:

• The game will present a summary of the results after five rounds, including the results of each round and whether you won, lost, or got tied.
• The overall number of wins, ties, and defeats for you and the computer will be shown.

4. Exit the Game: To stop the game at any point, just choose the "Exit" selection on the Main menu.

5. Extra Details:

• You compete against the computer in the game's single-player mode.

• You may review the game's instructions at any moment by selecting the "Help" option from the menu.