**PROJECT RPG Game Simulator - Repetition Structures 100 points**

**Objective** To type a simple Java program, execute ( run ) the program for some particular values, observe the output and then modify the program.

***PROJECT DESCRIPTION***

Type, compile and run the basic Java program that is shown in **Figure 1** , which follows.

Then compile and run your program, observe the output then modify the program.

***Information About This Project***

For this project we will create a simple Role Play Game ( RPG ) simulator.

***Steps to Complete This Project***

**STEP 1**  **Open NetBeans**

Open NetBeans and create a Java project with the following details.

For Project Name include **Lab5**

For the Main Class include **lab5.RPGSimulator**

In your **Code** window, shown below, copy in the program code shown in **Figure 1** below, in the appropriate places, except substitute your own name in place of Sammy Student.

|  |
| --- |
|  |

**Figure 1 Source Code for the RPG Game Simulator Program**

|  |
| --- |
| **import java.util.Random;**  **// Sammy Student, Programmer**  **public class RPGSimulator {**    **public static void main(String args[]) throws InterruptedException**  **{**  **Random rand1 = new Random();**  **Random rand2 = new Random();**    **String mission1 = "Capture the Castle";**  **String mission2 = "Enter the Hidden City";**  **String mission3 = "Rescue the Princess";**  **String mission4 = "Traverse the Forest";**  **String mission5 = "Locate the Tunnel Entrance";**    **String player1 = "", player2 = "";**    **char letter = '\0', role = '\0';**    **// initial health and treasures for the two players**  **int number = 0, mission = 0, health1 = 100, health2 = 100;**  **int treasure1 = 100, treasure2 = 100, encounter = 0;**    **// define your role**  **role = (char)(rand1.nextInt(26) + 'a');**  **if(role >= 'a' && role <= 'm')**  **{**  **player1 = "protagonist";**  **player2 = "opponent";**  **}**  **else**  **{**  **player1 = "opponent";**  **player2 = "protagonist";**  **}**  **System.*out*.println("you are the " + player1);**    **// define your mission**  **number = rand2.nextInt(5) + 1;**    **System.*out*.print("your mission is: ");**  **switch(number)**  **{**  **case 1: System.*out*.println(mission1); mission = 1; break;**  **case 2: System.*out*.println(mission2); mission = 2; break;**  **case 3: System.*out*.println(mission3); mission = 3; break;**  **case 4: System.*out*.println(mission4); mission = 4; break;**  **case 5: System.*out*.println(mission5); mission = 5; break;**  **}**  **System.*out*.println("");**    **// let the game simulation commence (limit the # of encounters)**  **for (encounter = 1; encounter <= 20; encounter++)**  **{**  **letter = (char)(rand1.nextInt(3) + 'a');**  **number = rand2.nextInt(10) + 1;**    **// randomly encounter the enemy**  **if(letter == 'a' || letter == 'b')**  **//if(mission == 1 || mission == 2 || mission == 3)**  **{**  **health1 -= 10;**  **health2 += 15;**  **treasure1 += 10;**  **treasure2 -= 10;**  **//Thread.sleep(5000);**  **System.*out*.print("battle: " + encounter);**  **System.*out*.println("\t" + "health" + "\t" + "treasure");**  **System.*out*.println(player1 + "\t" + health1 + "\t" + treasure1);**  **System.*out*.println(player2 + "\t" + health2 + "\t" + treasure2);**  **System.*out*.println("");**  **//encounter++;**  **}**  **else if(letter == 'c' || letter == 'd')**  **{**  **health1 += 20;**  **health2 -= 30;**  **treasure1 += 40;**  **treasure2 -= 20;**  **//Thread.sleep(500);**  **System.*out*.print("battle: " + encounter);**  **System.*out*.println("\t" + "health" + "\t" + "treasure");**  **System.*out*.println(player1 + "\t" + health1 + "\t" + treasure1);**  **System.*out*.println(player2 + "\t" + health2 + "\t" + treasure2);**  **System.*out*.println("");**  **//break;**  **}**  **else**  **{**  **health1 += 30;**  **health2 -= 20;**  **treasure1 -= 30;**  **treasure2 += 20;**  **//Thread.sleep(500);**  **System.*out*.print("battle: " + encounter);**  **System.*out*.println("\t" + "health" + "\t" + "treasure");**  **System.*out*.println(player1 + "\t" + health1 + "\t" + treasure1);**  **System.*out*.println(player2 + "\t" + health2 + "\t" + treasure2);**  **System.*out*.println("");**  **//break;**  **}**      **health1 -= 5;**  **health2 -= 5;**  **treasure1 -= 5;**  **treasure2 -= 5;**  **if (health1 <= 0 || health2 <= 0 || treasure1 <= 0 || treasure2 <= 0)**  **{**  **//System.out.print("encounter: " + encounter + " action status: ");**  **break;**  **}**  **}**  **System.*out*.println("simulation has ended\n");**  **System.*out*.println("results:\n");**  **System.*out*.println("\t\t" + "health" + "\t" + "treasure");**  **System.*out*.println(player1 + "\t" + health1 + "\t" + treasure1);**  **System.*out*.println(player2 + "\t" + health2 + "\t" + treasure2);**  **}**  **}** |

**STEP 2 Build, Compile and Run the Program**

From the NetBeans Run menu select Run Project (Lab5) to run your app.

**STEP 3 Test the Program**

Once you have successfully compiled your program, review the output **Console** window of NetBeans.

**Initial Test Run**

|  |
| --- |
| **you are the opponent**  **your mission is: Enter the Hidden City**  **battle: 1 health treasure**  **opponent 90 110**  **protagonist 115 90**  **battle: 2 health treasure**  **opponent 105 145**  **protagonist 80 65**  **battle: 3 health treasure**  **opponent 90 150**  **protagonist 90 50**  **battle: 4 health treasure**  **opponent 75 155**  **protagonist 100 35**  **battle: 5 health treasure**  **opponent 90 190**  **protagonist 65 10**  **battle: 6 health treasure**  **opponent 75 195**  **protagonist 75 -5**  **simulation has ended**  **results:**  **health treasure**  **opponent 70 190**  **protagonist 70 -10** |

**STEP 4 Review the Initial Program Code**

Examine the starter source code that was given in this project.

• The Game Missions are hardcoded in the program source code.

• A player role ( protagonist or opponent ) is assigned to the game player at random.

• A game mission is selected also at random and displayed to the player.

• An initial level of health and a default treasure score are assigned to both the player and their adversary.

• The action commences and we are allowed up to twenty encounters with our adversary.

• A letter character is randomly selected to define each encounter with your adversary.

• The initial program code then uses an if / else if block to randomly give or take health and or treasure from both player and opponent.

• After each encounter the current health levels and treasure scores are displayed to the program user.

• The simulation terminates when a player’s health is 0 or less or when their treasure score is 0 or less.

• The final scores are then displayed to the user.

**STEP 4 Modify Your Program**

Modify your existing code as follows for this lab:

( complete at least FOUR of the modifications that are listed below )

• Add two more missions to the original starter code, which already has five listed missions.

• Add some bonus health and treasure points if the simulation exceeds 5 battle encounters.

• Add an additional else if statement block that will allow for more game play interaction.

• Create a new character that appears only for one of the missions. The character can be friend or foe to either of the protagonist player or the opponent player

and can boast or lessen the health and / or treasure values of the two players.

• If the mission is to "Capture the Castle", then allow the players to use a cloaking mode, which will automatically boost their health to a level of 100 .

• If the mission is to "Traverse the Forest", then allow the protagonist to escape an encounter with his / her opponent.

**STEP 5 Submit Your Project**

Once you have determined that your modified program is correctly displaying game play interaction, complete the submission process as follows:

Open MS Word and type a heading for a new document that includes your full name, course number, lab number and date.

Within the document paste in a snapshot of your modified code. Label your snapshot of your modified run with a reasonable description.

After your snapshot, paste in your finished source code as well copied in from your NetBeans editor.

Submit your MS Word document to Blackboard when complete.