Java Project

PROJECT REPORT

Number Guessing Game


```
PS C:\Users\Ngs11\Dropbox\Java_Programming\src> java Game.java

 THE GAME OF GUESSING !

 The computer has guessed a number !

 Enter a number: 50
 Take it up a notch.

 Enter a number: 75
 Take it up a notch.

 Enter a number: 85
 Take it up a notch.

 Enter a number: 93
 Little lower.....

 Enter a number: 89
 Take it up a notch.

 SHOOT !

 The number was 91

 Thanks For Playing !
```

Friday 21st May, 2021.

1 Introduction

This is a simple game programmed using basic Java concepts. The game consists of the user requiring to guess a number randomly generated by the computer.

2 Working of the Game

Following is how the game works:

1. The computer generates a number between 1 to 100

2. The computer asks the user to guess

3. If the guess is correct user wins and program exits

4. Else computer gives a hint to the user for the correct answer

5. While the user exhausts his 5 attempts the cycle of guessing-checking continues.

```java
/* A simple number guessing game.

   Uses java.util.Random and java.util.Scanner packages
   User gets 5 attempts to guess a random number from 1 to 100.

   The number of attempts is intentionally kept less than 7.
   If the number of guesses is 7 or more, a smart user can
   use the concept of binary search to arrive at the correct
   guess since 2^7 = 128.
 */
/* ************************************/

import java.util.Random; // package to generate random numbers
import java.util.Scanner; // package to get user input

/* ************************************/

/**
 * Game class starts the game.
 */
public class Game {

    public static void main(String[] args) // main function
    {
        System.out.println("\n THE GAME OF GUESSING !");

        int num; // var for user input
```

```java
29        int success = 7; // checks if user has exhausted all
   attempts
30
31        Working guess = new Working(success); // creating an object
    to play the game
32
33        try (Scanner sc = new Scanner(System.in)) {
34            while (guess.valid()) // loop till the guess is valid
35            {
36                System.out.print("\n Enter a number: "); // user
   input
37                num = sc.nextInt();
38
39                guess.check(num); // checking user's guess
40                success--; // incrementing the attempts used
41            }
42        }
43
44        if (success == 0) // if all attempts are exhausted
45            System.out.println("\n SHOOT !");
46
47        System.out.println("\n The number was " + guess.getNumber()
   ); // prompting the correct answer
48        System.out.println("\n Thanks For Playing !\n\n\n"); //
   outro
49     }
50 }
51
52 /* ***************************************/
53
54 /**
55  * Working class implements the actual working of the game
56  */
57 class Working {
58     private final int number; // number should not be public or
   mutable
59     private int attempts; // number of attempts to validate
   guessing attempts
60     private final int noOfGuesses; // setting the maximum guesses
   the user can make
61
62     /**
63      * Constructor
64      * initialises the random number, number of guesses and
   attempts
65      */
66     Working(int _noOfGuesses) {
67         Random rand = new Random(); // generating random number
68         number = rand.nextInt(100) + 1;
69
70         System.out.println("\n The computer has guessed a number !"
   );
71
72         noOfGuesses = _noOfGuesses;
73         attempts = 0;
```

```
 74       }
 75
 76       /**
 77        * Checks if the attempt is valid
 78        *
 79        * @return boolean value true if the attempt is valid
 80        */
 81       boolean valid() {
 82           return attempts < noOfGuesses;
 83       }
 84
 85       /**
 86        * checks the guess of the user
 87        * informs the user about the guess made
 88        *
 89        * @param guess - guess of the user
 90        *
 91        */
 92       void check(int guess) {
 93           attempts = attempts + 1; // incrementing the number of
     attempts
 94
 95           if (guess == number) { // case user guessed it right
 96               System.out.println("\n Congratulations You Guessed It !
     ");
 97               attempts = noOfGuesses;
 98           } else if (guess < number) { // case user guessed less than
     the answer
 99               System.out.println(" Take it up a notch.");
100           } else { // case user guessed higher than answer
101               System.out.println(" Little lower.....");
102           }
103       }
104
105       /**
106        * to get the random number generated
107        *
108        * @return generated random number
109        */
110       int getNumber() {
111           return number; // number being private requires a method to
     get it
112       }
113 }
114 // End of Code
115 /* **********************************/
```

Listing 1: PageRank Simulation

3 Working of the Program

The program uses basic Java Programming concepts like classes, for-loops, conditions, constructors and data protection. The Game class is the class that begins the execution of the program. The Working class is where the actual working of the game is implemented. An object is created of the working class which initialises the random number and the number of guesses the user will have. The check method checks for each guess if it is correct and provides hints for the next guess. The validate method makes sure that the user does not make more than 5 guesses. The get number method is used to retrieve the random number generated by the computer. The number generated is kept private and static so as to prevent its modification and make it inaccessible from outside the class. The noOfAttempts variable is also kept private and final so that one cannot modify or access it to their advantage.

4 Why only 5 attempts?

Quite intuitively, lesser the number of attempts the tougher the game gets. However there is also a mathematical argument to limit the number of guesses to 6.

This is because after every guess the computer is providing us with some extra information about the number. Since the numbers from 1 to 100 could be considered to be an array sorted in ascending order,we can use the concept of binary search to arrive at the answer.

However the binary search decreases the search space by half after every iteration. So, after 4 iterations, we can narrow down the search space to 6 numbers. Hence the game reduces to guessing a number from 6 numbers with the probability of getting the correct answer on the 5th guess to be 0.1666667.

Similarly in guess of 6 allowed attempts the game reduces to guessing a number out of 3 numbers on the 6th guess.This makes the probability of winning to be 0.333333.

And if the number of attempts are 7, then one could definitely win

every time if s/he plays smartly.

5 Example when allowed attempts are 7

Figure 1: With 7 guesses one is sure to win.