# Laboratory 1: Introduction to java simulation and Random Number Generation

1. Introduction and Purpose of Experiment

Computer simulation provides students to design and implement computer simulation models, conduct simulation experiments and evaluate system performance. This laboratory exercise will help the students to get familiar with using object‐oriented simulation in Java.

Java (Structured Parallel Discrete Event Simulation in Java) system is designed to incorporate the parallel programming technology into discrete event simulations. The java system adopts the approach of augmenting a general-purpose language with essential constructs to support simulation modeling based on the process-oriented modeling technology.

Random numbers are widely used ingredient in the simulation of almost all discrete systems. Simulation languages generate random numbers that are used to generate event times and other random variables. Random number generators have applications in gambling, statistical sampling, computer simulation, cryptography, completely randomized design and other areas where producing an unpredictable result is desirable. The generation of pseudo random numbers is an important and common task in computer programming.

1. Aim and Objectives

Aim

* To use Netbeans and understand using object‐oriented simulation in Java
* To develop programs generating random numbers and Understand its significance in various applications

Objectives

At the end of this lab, the student will be able to

* Explain the features and use of Netbeans IDE to develop java programs for simulation
* Edit, compile and execute java programs successfully using Netbeans IDE

Use different random generation methods for generating random numbers

Create java programs for generating random numbers

1. Experimental Procedure

Students are given a set of programs for generating random numbers using built-in methods. Programs should be edited, compiled and executed using Netbeans IDE.

Random number generation using inbuilt methods/manually

Ex: coin toss, die, and cards

1. Calculations/Computations/Algorithms

Generate a random numbers for coin flip, die and cards

1. Presentation of Results

/\*

 \* To change this license header, choose License Headers in Project Properties.

 \* To change this template file, choose Tools | Templates

 \* and open the template in the editor.

 \*/

package rnumgen;

import java.util.Random;

import java.util.Scanner;

/\*\*

 \*

 \* @author Student

 \*/

public class RNUMGEN {

    /\*\*

     \* @param args the command line arguments

     \*/

    public static void main(String[] args) {

        // TODO code application logic here

        Random rgen = new Random();

        Scanner input = new Scanner(System.in);

        System.out.println("RANDOMGEN");

        System.out.println("Enter your choice : ");

        System.out.println("1.Coin\t2.Dice\t3.Card");

        int choice = input.nextInt();

        switch(choice) {

            case 1: {

                System.out.println("How many times do you want to flip a coin ? : ");

                int N = input.nextInt();

                System.out.print("[ ");

                int count = 0;

                for (int i = 0 ; i < N ; i++) {

                    int val = rgen.nextInt(2);

                    if (val == 0)

                        count++;

                    System.out.print( val + ", ");

                }

                System.out.print("\b]");

                System.out.println("\nCOUNT 0 : " + count + " , 1 : " + (N-count) );

                break;

            }

            case 2: {

                int []count = new int[7];

                System.out.println("How many times do you want to roll a dice ? : ");

                int N = input.nextInt();

                System.out.print("[ ");

                for (int i = 0 ; i < N ; i++) {

                    int val = (rgen.nextInt(6)+1);

                    count[val]++;

                    System.out.print( val + ", ");

                }

                System.out.print("\b]\nCOUNTS: \n");

                System.out.print("[ ");

                int j = 0;

                for (int e : count) {

                    System.out.print(j + " : " + e + ", ");

                    j++;

                }

                System.out.print("\b]\nCOUNTS: \n");

                break;

            }

            case 3: {

                int []count = new int[53];

                System.out.println("How many card do you want ? : ");

                int N = input.nextInt();

                System.out.print("[ ");

                for (int i = 0 ; i < N ; i++) {

                    int val = (rgen.nextInt(52)+1);

                    count[val]++;

                    System.out.print( val + ", ");

                }

                System.out.print("\b]\nCOUNTS: \n");

                System.out.print("[ ");

                int j = 0;

                for (int e : count) {

                    System.out.print(j + " : " + e + ", ");

                    j++;

                }

                System.out.print("\b]\nCOUNTS: \n");

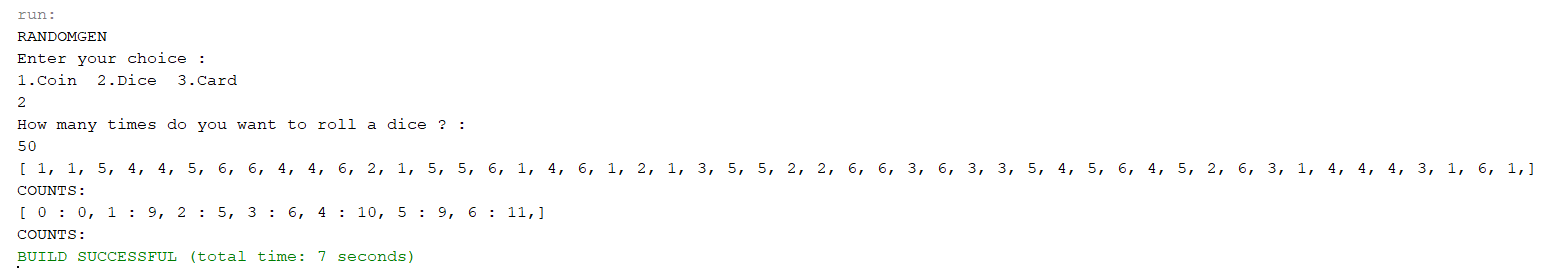
                break;

            }

        }

    }

}

1. Analysis and Discussions
2. Conclusions

Random class is used to generate pseudo-random numbers in java. An instance of this class is thread-safe. The instance of this class is however cryptographically insecure. This class provides various method calls to generate different random data types such as float, double, int.

Constructors:

Random(): Creates a new random number generator

Random(long seed): Creates a new random number generator using a single long seed

The random() method returns a double value with a positive sign, greater than or equal to 0.0 and less than 1.0. When you call Math.random(), under the hood, a java.util.Random pseudorandom-number generator object is created and used.

The java.util.Random class implements what is generally called a linear congruential generator (LCG). It is designed to be fast but does not meet requirements for real-time use, such as use in unique session ID generation on a web server, scientific experiments, cryptography, or lotteries and sweepstakes where a monetary stake is involved.