NEPAL ENGINEERING COLLEGE

(Affiliated to Pokhara University)

Changunarayan, Bhaktapur

A report on : #4

4. Congruential & Mid Square Method to generate Random number.

Submitted By:

Name: Raj Kumar Khadka

Roll No.: 020-348

Date: 27th June 2024

Submitted To:

Department: Computer Engineeri

Teacher's Signature:

2081/03/13

[illegible]

Labsheet - 4

OBJECTIVE

To generate random numbers using Linear Congruential method and mid square method.

THEORY

Linear Congruential Method is a class of Pseudo Random Number Generator (PRNG) algorithms used for generating sequences of random-like numbers in a specific range.

$$X_{n+1} = (aX_n + c) \bmod m, \quad n \geq 0. \; X_n \text{ is choosen } [0, m-1], n \geq 0$$

Approach:

- Choose X_0 mod m, multiplier a & increment term c.
- Initialize reqd amount of random numbers to generate (say, an integer variable noOfRandomNums).
- Define size noofRandomNums.
- Initialize 0th index of vector with seed value.
- Now use, randomNums[i] = ((randomNums[i-1] * a) + c) % m

Finally, return the random number.

PROGRAM

```
import java.util *;
class GFG {
    static void linearCongruentialMethod (int X0, int m, int a, int c,
            int[] randomNums, int noOfRandomNums)
    {
        randomNums[0] = X0;
        for (int i=1; i<noOfRandomNums ; i++)
        {
            randomNums[i] = ((randomNums[i-1] *a) +c) % m;
        }
    }
    public static void main (Sting[]args)
    {
        int X0 = 5;
        int m = 7;
        int a = 3;, c = 3;
        int noOfRandom Nums = 10;
int[] ~~Ra~~ randomNums = new int [noOfRandomNums];
 linearCongruentialMethod (x0, m, a, c, randomNums, noOfRandomNum
 for (int i=0; i<noOfRandomNums; i++)
  {
        System.out.print (randomNums[i] + " ");
   }
}
}
```

Output:

```
5 4 1 6 0 3 5 4 1 6
```

Generate Random Numbers Using Mid Square

Proposed by Van Neumann. In this method, we have seed & seed is squared & its midterm is fetched as random numbers. Suppose a N number is squared & becomes 2N, if it doesnot become 2N then we add zeros to make 2N.
A good algorithm is one which doesnot depend on seed & period should be maximally long that it should almost touch every number in its range before it starts repeating itself as a rule of thumb remember that longer the period more random is the number.

Example:

```
Number --> Square --> Mid-term
  14   -->  0196  -->  19
  19   -->  0361  -->  36
  36   -->  1296  -->  29
  29   -->  0841  -->  84
  84   -->  7056  -->  05
  05   -->  0025  -->  02
  02   -->  0004  -->  00
  00   -->  0000  -->  00
```

Implementation :

```
import java.util.Random;
public class Main {
      static int rangeArray[]
         = {1, 10, 100, 1000, 10000, 100000, 1000000, 10000000,
                                               100000000 };
      static long middleSquareNumber(long num, int digit)
      {   long sqn = num * num, nextNum = 0;
          int trim = (digit/2);
          sqn = sqn/ rangeArray [trim];
          for (int i=0; i < digit; i++)
          {   nextNum += (sqn % (rangeArray [trim])) * (rangeArray[i]);
              sqn = sqn /10;
          }
          return nextNum;
      }
      public static void main (String args[])
      {   int numberOfDigit = 3;
          int start = range Array [numberOfDigit - 1],
               end = rangeArray [numberOfDigit];
              1000                              1000
         Random rand = new Random ();      900 3225 +100
         long nextNumber = rand.nextInt (end - start) + start;
         System.out.print ("The random numbers for the Geeks
         are : \n" + nextNumber + ", ");
         for (int i=0; i<9; i++) {
              nextNumber = middle SquareNumber (nextNumber, number
                                                   OfDigit);
              System.out.print (nextNumber + ", ");
         }
   }  }
}
```

Output :

```
The random numbers for the Geeks are :

325, 562, 584, 105, 102, 40, 160, 560, 360, 960
```

CONCLUSION

Hence, we successfully generated random numbers using linear congruential method & mid square method.