**SUPROTIK DEY**

**IT Hx-31**

**4TH SEM, 510815050**

1. **Write a program in C/C++/Java to test the randomness of LCG, BBS and Stock random function.**

**SOURCE CODE: (Frequency Test)**

#include <stdio.h>

#include <time.h>

#define maxlen 1000

//Linear congruential generator

#define X0 2

#define mul 2

#define c 1

//blum blum shub generator..

#define P 47

#define Q 67

#define R 207

//freqRand test

#define D\_AlphaFor0\_05and10 0.410

void lcgInit(int \*x)

{

\*x = X0;

return;

}

int lcgRand(int a, int b, int \*xn)

{

int m, new;

m = b - a + 1;

if(X0 > m || mul > m || c > m)

{

printf("\nCannot compute lcg!!");

return 0;

}

new = ( mul \* (\*xn) + c) % m;

\*xn = new;

return (new + a);

}

void generateLcgStream(int a, int b, int \*xn, int \*arr, int num)

{

int i;

for(i = 0; i < num; i++)

{

arr[i] = lcgRand(a, b, xn);

}

return;

}

int gcd(int a, int b)

{

// Everything divides 0

if (a == 0 || b == 0)

return 0;

// base case

if (a == b)

return a;

// a is greater

if (a > b)

return gcd(a-b, b);

return gcd(a, b-a);

}

void bbsInit(int \*N, int \*M, int \*X, int A, int B)

{

\*N = P\*Q;

if((P%4!=3)||(Q%4!=3)||(gcd(\*N, R) != 1))

{

printf("\nBBS cannot be generated!!");

return;

}

\*X = (R\*R)%(\*N); //seeding

\*M = B- A +1;

}

void generateBbsStream(int N, int M, int A, int \*X, int \*stream, int n)

{

int i;

for(i = 0; i < n; i++)

{

\*X = ((\*X)\*(\*X)) % N;

stream[i] = ((\*X) % M) + A;

}

}

void initStockRandomFn()

{

srand(time(NULL));

}

void generateStockRandomFnStream(int a, int b, int n, int \*arr)

{

int i, m;

m = b - a + 1;

for(i = 0; i < n; i++)

{

arr[i] = rand() % m + a;

}

}

void printStream(int \*arr, int n)

{

int i;

for(i = 0; i < n; i++)

{

printf("%d ", arr[i]);

}

}

void printStreamFL(float \*arr, int n)

{

int i;

for(i = 0; i < n; i++)

{

printf("%f ", arr[i]);

}

}

void scaleToProperFraction(int \*arr, float \*arr2, int n, float div)

{

int i;

for(i = 0; i < n; i++)

{

arr2[i] = (float) arr[i] / div;

}

}

void swap(float \*xp, float \*yp)

{

float temp = \*xp;

\*xp = \*yp;

\*yp = temp;

}

void sortascFL(float \*arr, int n)

{

int i, j;

for(i = 0; i < n - 1; i++)

{

// Last i elements are already in place

for (j = 0; j < n-i-1; j++)

if (arr[j] > arr[j+1])

swap(&arr[j], &arr[j+1]);

}

}

void freqTestForRand(float \*arrFL, int n)

{

float DPlus, DMinus, temp, temp2, D;

int i;

//sorting...

sortascFL(arrFL, n);

printStreamFL(arrFL, n);

DPlus = (1.0/n) - arrFL[0];//init

DMinus = arrFL[0];//init

for(i = 0; i < n; i++)

{

temp2 = (float) ((i + 1.0) / n);

temp = (float) (temp2 - arrFL[i]);

if(temp > DPlus)

DPlus = temp;

temp2 = (float) ((i + 0.0)/n);

temp = (float) (arrFL[i] - temp2 );

if(temp > DMinus)

DMinus = temp;

}

if(DPlus > DMinus)

D = DPlus;

else

D = DMinus;

printf("\nDPlus is %f.",DPlus);

printf("\nDMinus is %f.",DMinus);

printf("\nD is %f.",D);

if(D > D\_AlphaFor0\_05and10)

printf("\nNull hypothesis (Data are a sample from uniform distribution) is rejected.");

else

printf("\nNull hypothesis is accepted.");

}

int main()

{

int A, B, iter, i;

int XnForLcg, lcgStream[maxlen]; //vars for lcg

int XnForBbs, BBSstream[maxlen], M, N; //vars for bbs

int streamStock[maxlen]; //vars for stock rand fn

float lcgStreamFL[maxlen], BBSstreamFL[maxlen], streamStockFL[maxlen];

A = 0;

B = 1000;

iter = 10;

lcgInit(&XnForLcg);

generateLcgStream(A, B, &XnForLcg, lcgStream, iter);

printf("\n\nLCG:\n");

printStream(lcgStream, iter);

bbsInit(&N, &M, &XnForBbs, A, B);

generateBbsStream(N, M, A, &XnForBbs, BBSstream, iter);

printf("\n\nBBS:\n");

printStream(BBSstream, iter);

initStockRandomFn();

generateStockRandomFnStream(A, B, iter, streamStock);

printf("\n\nStock rand fn:\n");

printStream(streamStock, iter);

//generation done.. implementing frequency test!!

//scaling

scaleToProperFraction(lcgStream, lcgStreamFL, iter, B);

scaleToProperFraction(BBSstream, BBSstreamFL, iter, B);

scaleToProperFraction(streamStock, streamStockFL, iter, B);

printf("\n\nLCG scaled:\n");

printStreamFL(lcgStreamFL, iter);

printf("\n\nBBS scaled:\n");

printStreamFL(BBSstreamFL, iter);

printf("\n\nStock rand fn scaled:\n");

printStreamFL(streamStockFL, iter);

printf("\n\nLCG Test:\n");

freqTestForRand(lcgStreamFL, iter);

printf("\n\nBBS Test:\n");

freqTestForRand(BBSstreamFL, iter);

printf("\n\nStock rand fn Test:\n");

freqTestForRand(streamStockFL, iter);

return 0;

}

**OUTPUT:**

LCG:

5 11 23 47 95 191 383 767 534 68

BBS:

902 194 995 60 527 374 224 700 617 639

Stock rand fn:

917 230 146 29 307 186 981 763 580 237

LCG scaled:

0.005000 0.011000 0.023000 0.047000 0.095000 0.191000 0.383000 0.767000 0.534000

0.068000

BBS scaled:

0.902000 0.194000 0.995000 0.060000 0.527000 0.374000 0.224000 0.700000 0.617000

0.639000

Stock rand fn scaled:

0.917000 0.230000 0.146000 0.029000 0.307000 0.186000 0.981000 0.763000 0.580000

0.237000

LCG Test:

0.005000 0.011000 0.023000 0.047000 0.068000 0.095000 0.191000 0.383000 0.534000

0.767000

DPlus is 0.509000.

DMinus is 0.005000.

D is 0.509000.

Null hypothesis (Data are a sample from uniform distribution) is rejected.

BBS Test:

0.060000 0.194000 0.224000 0.374000 0.527000 0.617000 0.639000 0.700000 0.902000

0.995000

DPlus is 0.100000.

DMinus is 0.127000.

D is 0.127000.

Null hypothesis is accepted.

Stock rand fn Test:

0.029000 0.146000 0.186000 0.230000 0.237000 0.307000 0.580000 0.763000 0.917000

0.981000

DPlus is 0.293000.

DMinus is 0.117000.

D is 0.293000.

Null hypothesis is accepted.