КИЇВСЬКИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

**Лабораторна робота №1**

З теорії ймовірності та математичної статистики

**Тема**: «Генератори псевдовипадкових послідовностей»

Виконав:

студент ІІ курсу ФІОТ

група ІО-32

Попенко Руслан

Перевірив:

Марковський О. П.

Київ-2014

1. **Завдання.**

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Розрядність | Об’єм вибірки | Складність |
| **Т (р6)** | **8+** | **10000** | **Н** |

1. **Лістинг програми**

**Клас лаб. Робота**

**package** laba1;

/\*\*

\* **@author** Ruslan Popenko

\*

\*/

**public** **class** **Laba1** {

/\*\*

\* **@param** args

\*/

**public** **static** **void** **main**(**String**[] args) {

**TestSequence** **test** = **new** TestSequence();

**System**.***out***.println("Частотний тест");

**System**.***out***.println(test.Test1());

**System**.***out***.println("Диференціальний тест");

**System**.***out***.println(test.Test2());

test.Test3(3);

**System**.***out***.println("Нелінійна складність");

**System**.***out***.println(test.Test4());

}

}

**Клас LFSR**

**package** laba1;

**import** java.util.Random;

/\*\*

\* **@author** Ruslan Popenko

\*

\*/

**public** **class** LFSR {

**byte**[] work;

**byte**[] buf;

**byte**[] pol;

LFSR(**byte** bit, **byte**[] polinom) {

pol = polinom;

work = **new** **byte**[bit];

buf = **new** **byte**[bit];

Random rand = **new** Random();

**for** (**int** i = 0; i < work.length; i++) {

work[i] = (**byte**) rand.nextInt(2);

}

**for** (**int** i = 0; i < buf.length; i++) {

buf[i] = 95;

}

}

**public** **byte** Gen() {

**int** where = 0;

**byte** gen = work[0];

buf[buf.length - 1] = gen;

**for** (**int** i = 0; i < pol.length; i++) {

where = work.length - pol[i] - 1;

buf[where] = (**byte**) (gen ^ work[work.length - pol[i]]);

}

**for** (**int** i = 0; i < buf.length; i++) {

**if** (buf[i] == 95) {

buf[i] = work[i + 1];

}

}

**for** (**int** i = 0; i < buf.length; i++) {

work[i] = buf[i];

}

**for** (**int** i = 0; i < buf.length; i++) {

buf[i] = 95;

}

**return** gen;

}

}

**Клас Генератор**

**package** laba1;

**import** java.util.Random;

/\*\*

\* **@author** Ruslan Popenko

\*

\*/

**public** **class** TGenerator {

**byte**[] sequence;

TGenerator(**int** size) {

sequence = **new** **byte**[size];

**for (int i = 0; i < tabl.length; i++) {**

**if (i % 2 == 1) {**

**tabl[i] = 1;**

**} else {**

**tabl[i] = 0;**

**}**

**}**

**byte**[] p8 = { 4, 3, 2 };

LFSR R8 = **new** LFSR((**byte**) 8, p8);

**byte**[] p9 = { 4 };

LFSR R9 = **new** LFSR((**byte**) 9, p9);

**byte**[] p10 = { 3 };

LFSR R10 = **new** LFSR((**byte**) 10, p10);

**byte**[] p11 = { 2 };

LFSR R11 = **new** LFSR((**byte**) 11, p11);

**byte**[] p12 = { 6, 4, 1 };

LFSR R12 = **new** LFSR((**byte**) 12, p12);

**byte**[] p13 = { 4, 3, 1 };

LFSR R13 = **new** LFSR((**byte**) 13, p13);

**byte**[] buf = **new** **byte**[6];

**int** s = 0;

**for** (**int** i = 0; i < sequence.length; i++) {

buf[0] = R8.Gen();

buf[1] = R9.Gen();

buf[2] = R10.Gen();

buf[3] = R11.Gen();

buf[4] = R12.Gen();

buf[5] = R13.Gen();

**for** (**int** j = 0; j < buf.length; j++) {

s += (**int**) buf[j] \* Math.*pow*(2, j);

}

sequence[i] = tabl[s];

s = 0;

}

}

}

**Клас Тестування**

package laba1;

import java.util.ArrayList;

import java.util.Arrays;

/\*\*

\* @author Ruslan Popenko

\*

\*/

public class TestSequence {

TGenerator T = new TGenerator(10000);

public TestSequence() {

}

public double Test1() {

double s = 0;

for (int i = 0; i < T.sequence.length; i++) {

if (T.sequence[i] == 1)

s++;

}

double g = (double) T.sequence.length;

return (s / g);

}

public double Test2() {

double s = 0;

byte[] buf = new byte[T.sequence.length - 1];

for (int i = 0; i < T.sequence.length - 1; i++) {

buf[i] = (byte) (T.sequence[i] ^ T.sequence[i + 1]);

if (buf[i] == 1)

s++;

}

return (s / buf.length);

}

public void Test3(int sizeW) {

byte[] window = new byte[sizeW];

int length = (int) (Math.pow(2, sizeW));

int[] rang = new int[length];

for (int i = 0; i < window.length; i++) {

window[i] = 0;

}

boolean t = false;

for (int i = 0; i < rang.length; i++) {

rang[i] = 0;

for (int j = 0; j < T.sequence.length - sizeW + 1; j++) {

t = true;

for (int n = j; n < j + window.length; n++) {

if (T.sequence[n] != window[n - j]) {

t = false;

}

}

if (t) {

rang[i]++;

}

}

for (int j = window.length - 1; j >= 0; j--) {

if (window[j] == 0) {

window[j] = 1;

if (j != window.length - 1) {

for (int j2 = j + 1; j2 < window.length; j2++) {

window[j2] = 0;

}

}

break;

}

}

}

System.out.println("Вікно розміром " + sizeW);

double buf = 0;

for (int i = 0; i < rang.length; i++) {

buf = (double) rang[i];

buf = buf / T.sequence.length;

System.out.println(i + ") " + buf);

}

}

public int Test4() {

byte[] arr = T.sequence;

int windowSize = 0;

boolean isSizeFinded = false;

int maxNum = 0;

byte[] valuesArr = null;

for (int j = 0; j < arr.length - 1 && isSizeFinded == false; j++) {

windowSize++;

maxNum = (int) Math.pow(2, windowSize);

valuesArr = new byte[maxNum];

isSizeFinded = true;

for (int i = 0; i < valuesArr.length; i++) {

valuesArr[i] = -1;

}

for (int i = 0; i < arr.length - windowSize; i++) {

int numInWindow = getIntFromByteArr(arr, i, i + windowSize - 1);

int valueForNum = valuesArr[numInWindow];

if (valueForNum == -1) {

valuesArr[numInWindow] = arr[i + windowSize];

} else if (valueForNum != arr[i + windowSize]) {

isSizeFinded = false;

break;

}

}

}

return windowSize;

}

private int getIntFromByteArr(byte[] arr, int start, int end) {

int num = 0;

int twoInPower = 1;

for (int i = end; i >= start; i--) {

if (arr[i] == 1) {

num += twoInPower;

}

twoInPower \*= 2;

}

return num;

}

}

**Результат виконання програми**

