МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

ІНСТИТУТ ПІСЛЯДИПЛОМНОЇ ОСВІТИ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

****

**Звіт**

**з лабораторної роботи №1**

# на тему: «Створення генератора псевдовипадкових чисел»

***з дисципліни*** ***“*** ***Захист програм та даних ”***

**ВИКОНАВ:**

Гринчук Т.А.

ПЗС - 21

**ПЕРЕВІРИВ:**

доц. Яковина В.С.

Львів - 2015

**Тема.** Створення генератора псевдовипадкових чисел.

**Мета.** Ознайомитись з джерелами та застосуванням випадкових чисел, алгоритмами генерування псевдовипадкових чисел та навчитись створювати програмні генератори псевдовипадкових чисел для використання всистемах захисту інформації.

**Теоретичні відомості**

Псевдовипадкові послідовності (числа) — послідовності, що отримуються за цілком невипадковим [алгоритмом](http://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC), але мають властивості, дуже подібні до властивостей реалізацій [випадкових чисел](http://uk.wikipedia.org/wiki/%D0%92%D0%B8%D0%BF%D0%B0%D0%B4%D0%BA%D0%BE%D0%B2%D0%B0_%D0%B2%D0%B5%D0%BB%D0%B8%D1%87%D0%B8%D0%BD%D0%B0).

Псевдовипадкове число — елемент отриманої за певним алгоритмом числової послідовності, властивості якої наближаються до випадкової. Негативною особливістю псевдовипадкових чисел (як імітаційної моделі випадкових чисел) є періодичність отриманої послідовності.

Реалізація статистичних моделей за допомогою обчислювальних машин передбачає можливість отримання у достатній мірі випадкових показників (що є нетривіальним завданням, оскільки електронно-обчислювальна машина працює на основі чітко прописаної логіки та детерміністичного підходу), а також імітацію законів розподілу, що часто використовуються на практиці.

Зазвичай генерування псевдовипадкової послідовності передбачає два етапи. На першому генерують [псевдовипадкові числа](http://uk.wikipedia.org/wiki/%D0%9F%D1%81%D0%B5%D0%B2%D0%B4%D0%BE%D0%B2%D0%B8%D0%BF%D0%B0%D0%B4%D0%BA%D0%BE%D0%B2%D1%96_%D1%87%D0%B8%D1%81%D0%BB%D0%B0), що мають рівномірний розподіл на відрізку від нуля до одиниці. На другому цю послідовність перетворюють у послідовність, що має заданий [закон розподілу](http://uk.wikipedia.org/wiki/%D0%97%D0%B0%D0%BA%D0%BE%D0%BD_%D1%80%D0%BE%D0%B7%D0%BF%D0%BE%D0%B4%D1%96%D0%BB%D1%83).

Розглянемо лінійний конгруентний генератор. Саме він був запропонований першим. Математична модель ЛКГ така:

, де і = 1, ..., m – 1.

Вхідними параметрами такого генератора є: коефіцієнти а та с та початковий стан x0 (і, звичайно, значення модуля m). Визначимо межі значень коефіцієнтів. Обчислення здійснюються за модулем, отже максимальне значення всіх параметрів m – 1. Мінімальне значення коефіцієнту с дорівнює 0, а коефіцієнту а – дорівнює 1, оскільки при а = 0 модель вироджується в:

,

тобто весь час на виході генератора буде одне й те саме число. Однак, при а = 1 формула матиме вигляд:

.

Такий генератор формуватиме різні числа, однак його властивості з точки зору псевдовипадковості не є задовільними. Також, при с = 0 модель має вигляд:

.

Генератор формує псевдовипадкові числа, але існує обмеження у вигляді:

.

Таким чином, отримується мультиплікативний генератор.

Таким чином, кожен наступний стан отримується шляхом множення попереднього на коефіцієнт а та додавання коефіцієнту с.

**Виконання роботи**

Згідно до варіанту, наведеного в таблиці, створити програмну реалізацію генератора псевдовипадкових чисел за алгоритмом лінійного порівняння. Програма повинна генерувати послідовність із заданої при вводі кількості псевдовипадкових чисел, результати повинні як виводитись на екран, так і зберігатись у файл. Перевірити період функції генерації, зробити висновок про адекватність вибору параметрів алгоритму. У звіті навести протокол роботи програми, значення періоду функції генерації та зробити висновок про придатність цього генератора для задач криптографії.

**Варіант 3.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № Варіанту | Модуль порівняння,  m | Множник,  a | Приріст,  c | Початкове значення, |
| 3 | 2^12-1 | 4^5 | 2 | 8 |

**Текст програми**

**import** java.io.File;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** java.util.ArrayList;

**import** java.util.List;

**import** java.util.Scanner;

**public** **class** RandGen {

**public** **static** **long** *a* = (**long**)Math.*pow*(4, 5);

**public** **static** **long** *c* = 2;

**public** **static** **long** *m* = (**long**)Math.*pow*(2, 12) - 1;

**public** **static** **long** *x* = 8;

**public** **static** **long** getRand() {

*x* = (**long**)(*a* \* *x* + *c*) % *m*;

**return** *x*;

}

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

Scanner sc = **new** Scanner(System.*in*);

System.*out*.print("Count: ");

**if**(!sc.hasNextInt()) {

// Якщо користувач ввів не ціле число виводимо відповідне повідомлення

System.*out*.println("It's not integer value! Program terminate...");

**return**;

}

**int** count = sc.nextInt();

List<Long> list = **new** ArrayList<Long>();

**for**(**int** i = 0; i < count; i++) {

list.add(*getRand*());

System.*out*.println(list.get(i));

}

*saveToTextFile*(list, "d:\\Numbers.txt");

}

**public** **static** **void** saveToTextFile(List<Long> list, String fileName) {

**try** {

PrintWriter out = **new** PrintWriter(**new** File(fileName));

**try** {

**for**(Long l: list)

out.println(l.toString());

} **finally** {

out.close();

}

} **catch**(IOException ex) {

**throw** **new** RuntimeException("Error saving file !");

}

System.*out*.println("\nSaved to file:\n" + fileName);

}

}

**Результат виконання програми**

Запустимо програму на виконання (рис. 1):

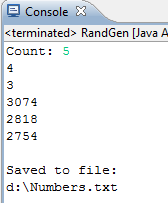
****

Рис. 1. Результат виконання програми

Текстовий файл зображено на рис. 2:

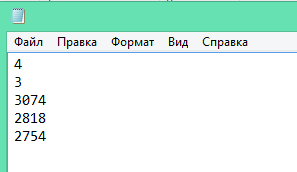


Рис. 2. Текстовий файл

**Висновок**

Виконавши дану лабораторну роботу я ознайомився з джерелами та застосуванням випадкових чисел, алгоритмами генерування псевдовипадкових чисел та навчитись створювати програмні генератори псевдовипадкових чисел для використання в системах захисту інформації.