Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**Выполнили с-ты гр. 17ВВ2:**

Живаев В.В.

Безжонов Р.Д.

**Приняли:**

Дубравин А.В.  
Карамышева Н.С.

Лабораторная работа № 3

**«Генераторы псевдослучайных последовательностей»**

## Отчёт по дисциплине

**«Защита информации в сети Интернет»**

**Цель работы.**

Изучить методы генерации случайных чисел. Разработать генератор случайных чисел, используя один из методов.

**Задание.**

Разработать генератор случайных чисел, используя метод в соответствии с номером варианта. Исследовать качество генератора, путем оценки распределения генерируемых чисел.

Для линейного конгруэнтного метода значение m всегда должно быть 231 -1.

Таблица: Источники энтропии.

|  |  |
| --- | --- |
| № Варианта | Метод генерации случайных чисел |
| 6 | Метод серединных произведений. Разрядность при вычислении 32 бита, разрядность результата 16 бит. |

**Листинг.**

#include "stdafx.h"

#include<iostream>

#include <Windows.h>

#include <vector>

using namespace std;

void middleMul(); // прототипфункциигенерациипсевдослучайныхчисел

void separateToIntervals();

unsigned int r2 = 0;

#define NUMBERS\_COUNT 2000

#define INTERVAL\_WIDTH 1000

vector<int> valuesVector;

unsigned int r0 = 0, r1 = 0;

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

const int number\_numbers = NUMBERS\_COUNT; // количество псевдослучайных чисел

printf("Введите р1 и р0: ");

scanf\_s("%d %d", &r0, &r1);

cout << "\n";

for (int i = 0; i <= number\_numbers; i++) {

middleMul();

cout << r2 << "\n"; // генерация i-го числа

valuesVector.push\_back(r2);

}

separateToIntervals();

system("pause");

return 0;

}

void middleMul() // функция генерации псевдослучайных чисел

{

r2 = r1 \* r0;

r2 = (r2 >> 8 & 65535);

r0 = r1;

r1 = r2;

}

void separateToIntervals()

{

double percent = 0.0;

for (int interval = 0; interval < 66000; interval += INTERVAL\_WIDTH)

{

int count = 0;

for each (int val in valuesVector)

{

if (val >= interval && val < (interval + INTERVAL\_WIDTH)) {

count++;

}

}

double curPercent = (double)count / NUMBERS\_COUNT;

percent += curPercent;

printf("\nfrom %5d to %5d: %3d {%.3f\%%} Total%%: %.2f",

interval,

interval + INTERVAL\_WIDTH,

count,

curPercent,

percent);

count = 0;

}

}

**Результат работы.**

Ввод чисел и начало генерации показано на рисунке 1.

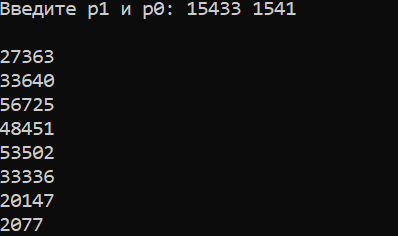


Рисунок 1. Результат работы программы**.**

Исследовали качество генератора, путем оценки распределения генерируемых чисел (рис. 2-5). Оценка велась при разбиении по интервалам шириной в 1000. Как видно из результатов ниже генератор выдает числа с примерно одинаковой частотой, в среднем 0.012%-0.017% чисел на интервал ширины 1000.

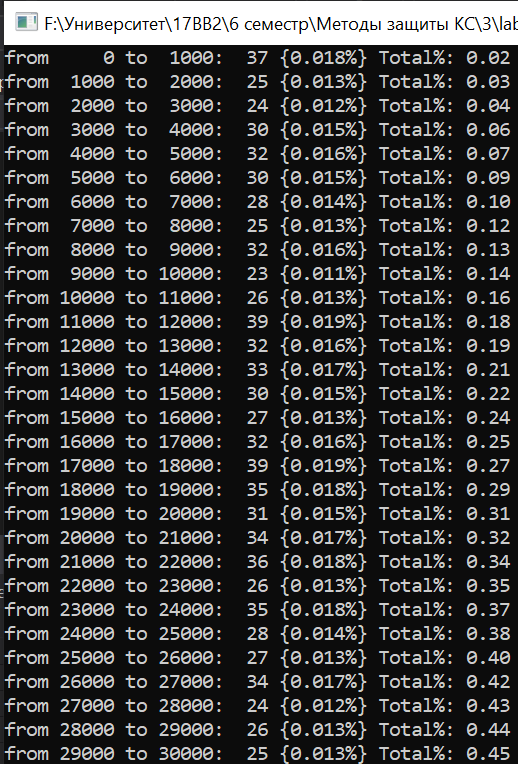


Рисунок 2. Распределение.

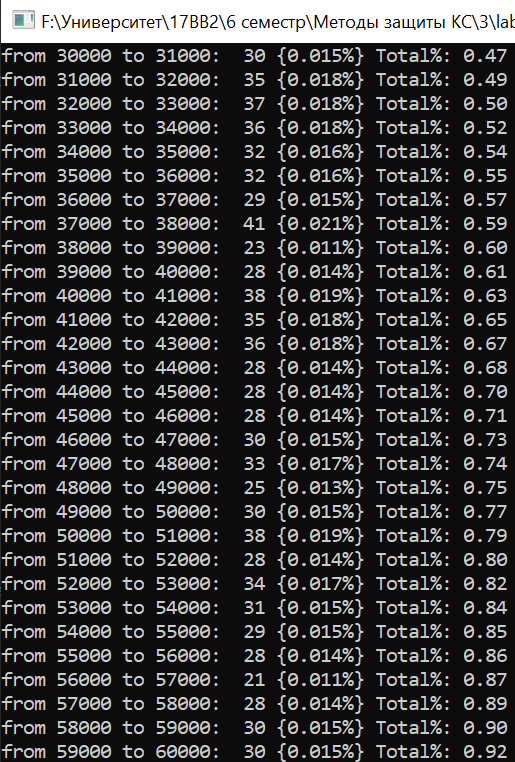


Рисунок 3. Распределение.

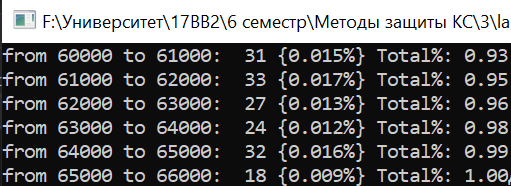


Рисунок 4. Распределение.

**Вывод.**

Изучили методы генерации случайных чисел. Разработали генератор случайных чисел, используя метод серединных преобразований.