Министерство образования Российской Федерации  
Пензенский государственный университет  
Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №3  
по курсу «Методы защиты КС»

Выполнили студенты группы 17ВВ2:

Левина Е.В.

Кудряшов Я.В.

Приняли:

Карамышева Н.С.

Дубравин А.В.

Пенза, 2020

**Цель** **работы:** генераторы псевдослучайных последовательностей

**Задание:** разработать генератор случайных чисел, используя метод в соответствии с номером варианта. Исследовать качество генератора, путем оценки распределения генерируемых чисел. Для этого необходимо разделить весь диапазон генерируемых чисел на 10 равных интервалов и подсчитать количество чисел, попадающих в каждый интервал. В качестве начального значения следует выбирать текущее время в формате Unix time (можно получить, используя функцию time\_t time(time\_t\* timer)). Для линейного конгруэнтного метода значение m всегда должно быть 231 -1



**Код программы:**

#include <iostream>

#include <iomanip>

#include <ctime>

using namespace std;

char getRandomNumber(short numberRandom);

int main()

{

setlocale(LC\_ALL, "rus");

cout << "Введите количество чисел для генерации: ";

short numbersAmount;

cin >> numbersAmount;

//short numbersAmount = 10;

char numberRandom = 0;

time\_t timer = time(&timer);

float max = 0;

char\* arr = new char[numbersAmount];

cout << "\nСгенерированные числа: ";

for (int i = 0; i < numbersAmount; i++)

{

numberRandom = getRandomNumber((short)timer + (short)numberRandom);

\*(arr + i) = numberRandom;

if (max < (short)numberRandom)

{

max = (short)numberRandom;

}

cout << "[" << (short)numberRandom << "] ";

}

float min = max;

for (int i = 0; i < numbersAmount; i++)

{

if (min > (short)\*(arr + i))

{

min = (short)\*(arr + i);

}

}

cout << "\n\nОценка распределения генерируемых чисел\n\n";

float step = (max - min) / 10.0f;

for (int i = 0; i < numbersAmount; i++)

{

cout << setw(2) << i + 1 << ")"

<< setprecision(1) << fixed

<< setw(7) << min << setw(3) << "~"

<< setw(7) << min + step << setw(3) << ": ";

short count = 0;

for (int j = 0; j < numbersAmount; j++)

{

if (min <= (short)\*(arr + j) && (short)\*(arr + j) <= (min + step))

{

count++;

}

}

cout << count << endl;

min += step;

}

delete[] arr;

return 0;

}

/\* Получить случайное значение \*/

char getRandomNumber(short numberRandom)

{

const int N = sizeof(short) \* 8 / 4; //сдвиг на 1/4 ячейки

short numberLeftShift = 0;

for (int i = 0; i < N - 1; i++)

{

numberLeftShift |= (bool)((1 << ((sizeof(short) \* 8 - 1) - i)) & numberRandom);

numberLeftShift <<= 1;

}

numberLeftShift |= (bool)((1 << ((sizeof(short) \* 8 - 1) - (N - 1))) & numberRandom);

numberLeftShift |= numberRandom << N;

short numberRightShift = 0;

for (int i = 0; i < N; i++)

{

numberRightShift |= ((bool)((1 << ((N - 1) - i)) & numberRandom)) << ((sizeof(short) \* 8 - 1) - i);

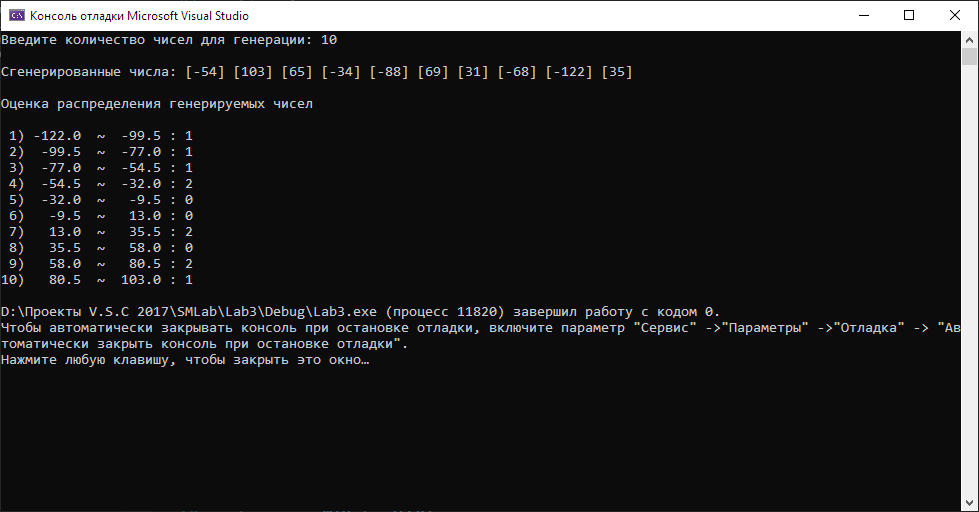
}

numberRightShift |= numberRandom >> N;

return (char)(numberLeftShift + numberRightShift);

}

**Результат работы программы:**



**Вывод**

При выполнении данной лабораторной работы мы научились разрабатывать генераторы псевдослучайных последовательностей