

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Инженерно-физический факультет
Кафедра автоматизированных систем обработки информации и
управления

ОТЧЕТ ПО ПРАКТИКЕ

Программная реализация генератора случайных
чисел. Генератор случайных чисел
Парка-Миллера с перетасовкой и без.

2 курс, группа 2УТС

Выполнил:

_____ А. В. Сахаутдинов
«___» _____ 2020 г.

Руководитель:

_____ С. В. Теплоухов
«___» _____ 2020 г.

Майкоп, 2020 г.

1. Введение

1.1. Цель работы

Целью данной работы является реализация генератора случайных чисел Парка-Миллера с перетасовкой и без.

1.2. Теория

Самая простая последовательность, которую можно предложить для реализации генератора равномерного распределения:

$$(j+1) = a * I(j) \pmod{m}$$

при соответствующем выборе констант. Константы были предложены Park и Miller:

$$a=75=16807, m=2^{31}-1=2147483647.$$

Модуль разлагается в выражение:

$$m=a*q+r$$

Если $r < q$ и $0 < z < m-1$, то при этом величины $a*(z \bmod q)$ и $r*[z/q]$ всегда лежат в интервале $0, \dots, m-1$. Для умножения $(a*z) \pmod{m}$ при этом используется алгоритм:

- $t = a(z \bmod q) - r[z/q]$
- если $t < 0$, то $t += m$.
- $(a*z) \pmod{m} = t$.

В случае констант Парка-Миллера можно использовать $q=12773$ и $r=2836$.

2. Ход работы

2.1. Код программы

```
#include <iostream>
#include <vector>
using namespace std;
class RandomNumberGenerator
{
protected:
    unsigned int    init_seed; // Начальное случайное значение
    unsigned int    cur_seed;  // Текущее случайное значение
    unsigned int    num_draws; //Размерность
public:
    RandomNumberGenerator(unsigned int _num_draws, unsigned int _init_seed) :
        num_draws(_num_draws), init_seed(_init_seed), cur_seed(_init_seed)
    {
    };
    virtual ~RandomNumberGenerator()
    {
    }
```

```

};
    virtual unsigned int get_random_seed() const
    {
        return cur_seed;
    }
    virtual void set_random_seed(unsigned int _seed)
    {
        cur_seed = _seed;
    }

    virtual void reset_random_seed()
    {
        cur_seed = init_seed;
    }

    virtual void set_num_draws(unsigned int _num_draws)
    {
        num_draws = _num_draws;
    }

    // получить случайное целое число
    virtual unsigned int get_random_integer() = 0;

    // Заполняет вектор однородными случайными величинами
    //на открытом интервале (0,1)
    virtual void get_uniform_draws(std::vector<double>& draws) = 0;
};

class LinearCongruentialGenerator : public RandomNumberGenerator
{
private:
    double max_multiplier;

public:
    LinearCongruentialGenerator(unsigned int _num_draws, unsigned int _init_seed = 1);
    virtual ~LinearCongruentialGenerator()
    {
    };

    virtual unsigned int get_random_integer();
    virtual void get_uniform_draws(std::vector<double>& draws);
};

```

```

const unsigned int  a = 16807;          //  $7^5$ 
const unsigned int  m = 2147483647;    //  $2^{32}$ 

// Константы алгоритма Шраге

const unsigned int  q = 127773;
const unsigned int  r = 2836;

// Конструктор параметров
LinearCongruentialGenerator::
LinearCongruentialGenerator(unsigned int _num_draws, unsigned int _init_seed) :
RandomNumberGenerator(_num_draws, _init_seed)
{
    if (!_init_seed)
    {
        init_seed = 1;
        cur_seed  = 1;
    }

    max_multiplier = 1.0 / (1.0 + (m - 1));
}

// Получает случайное целое число без знака
unsigned int LinearCongruentialGenerator::get_random_integer()
{
    unsigned int  kk  = 0;

    kk = cur_seed / q;

    cur_seed = a * (cur_seed - kk * q) - r * kk;

    if (cur_seed < 0)
    {
        cur_seed += m;
    }

    return cur_seed;
}

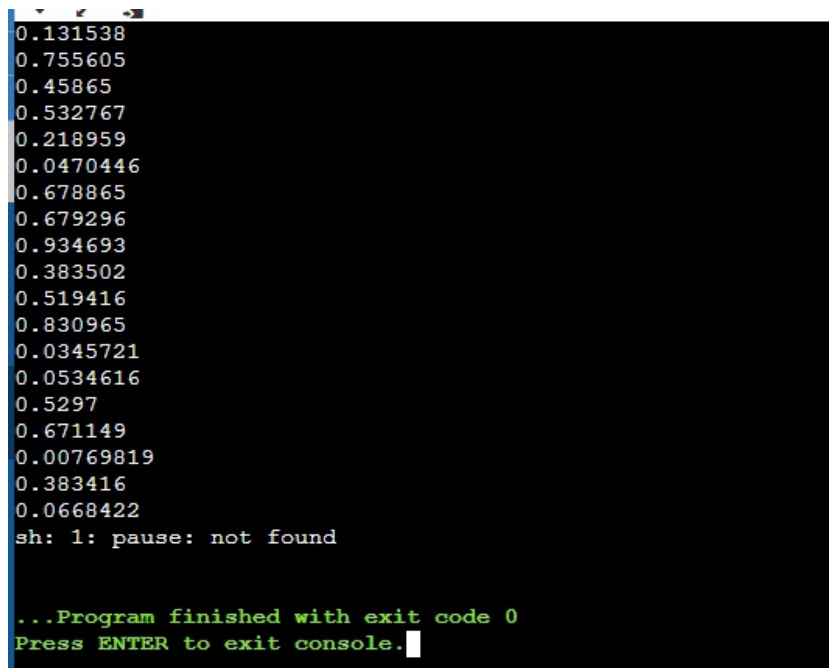
// Создайте вектор равномерных участков между (0,1)
void LinearCongruentialGenerator::get_uniform_draws(std::vector<double>& draws)
{
    for (unsigned int ii = 0; ii < num_draws; ++ii)
    {

```

```

        draws[ii] = get_random_integer() * max_multiplier;
    }
}
int main()
{
    unsigned int    init_seed = 1;
    unsigned int    num_draws = 20;
    vector<double> random_draws(num_draws,0.0);
    // Создать случайные формы
    // открыть интервал (0,1)
    LinearCongruentialGenerator    lcg(num_draws,init_seed);
    lcg.get_uniform_draws(random_draws);
    // вывод случайных чисел
    for (unsigned int ii = 1; ii < num_draws; ++ii)
    {
        cout << random_draws[ii] << endl;
    }
    system("pause");
    return 0;
}

```



```

0.131538
0.755605
0.45865
0.532767
0.218959
0.0470446
0.678865
0.679296
0.934693
0.383502
0.519416
0.830965
0.0345721
0.0534616
0.5297
0.671149
0.00769819
0.383416
0.0668422
sh: 1: pause: not found

...Program finished with exit code 0
Press ENTER to exit console.

```

Рис. 1. Окно программы с сгенерированными числами

Список литературы

- [1] Кнут Д.Э. Всё про $\text{T}_\text{E}\text{X}$. — Москва: Изд. Вильямс, 2003 г. 550 с.
- [2] Львовский С.М. Набор и верстка в системе $\text{L}_\text{A}\text{T}_\text{E}\text{X}$. — 3-е издание, исправленное и дополненное, 2003 г.
- [3] Воронцов К.В. $\text{L}_\text{A}\text{T}_\text{E}\text{X}$ в примерах, 2005 г.
- [4] Страуструп Б. Язык программирования $\text{C}++$, 2013 г.
- [5] Кёниг Э., Му Б. Эффективное программирование на $\text{C}++$, 2016 г.
- [6] Мейерс С. Эффективный и современный $\text{C}++$, 2018 г.
- [7] Довгаль В.А., Коробков В.Н. Программирование на языке $\text{C}++$ в среде Microsoft Visual Studio — Часть 1, Учебно-методическое пособие, 2015 г.