МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра информационной безопасности

ОТЧЕТ

по лабораторной работе №2

на тему: «**Псевдослучайные последовательности чисел**»

по дисциплине «Информационная безопасность»

Выполнили: Кожухова О.А. Шифр: 170582

Шорин В.Д. Шифр: 171406

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 71-ПГ

Проверил: Еременко В.Т.

Отметка о зачете:_____

Дата «____» _____ 2021г.

Орел, 2021 г.

Задание

1. В 1946 году Джон фон Нейман предложил генератор псевдослучайных чисел, основанный на том, что каждое последующее сило образуется возведением предыдущего в квадрат и отбрасыванием цифр с обоих концов. Постройте диаграмму распределения сгенерированной последовательности.

2. Написать программу, которая получает последовательности псевдослучайных чисел используя дробную часть многочлена

$$y_n = f(a \cdot n + b)$$

Получите последовательности, используя рациональные и иррациональные значения коэффициента *a.* Постройте диаграммы распределения полученной последовательности.

3. Создайте программу, реализующую конгруэнтный способ Лемера. Используйте целые, рациональные и иррациональные значения коэффициентов К, С и М. Постройте диаграммы распределения полученной последовательности.

4. Пользуясь конгруэнтным способом Лемера при С=0, $M=2^N$, К=3+8*I напишите программу генерирующую последовательность случайных чисел. Постройте диаграмму распределения полученных чисел.

5. Реализуйте алгоритм Марсалиа-Зеймана. Получите последовательность из 1000 первых чисел Фибоначчи, отбрасывая все разряды кроме единиц. Складывая все числа, удаленные на n элементов (используйте «бит переноса»), получите ряд случайных чисел Фибоначчи. Постройте диаграмму распределения полученных чисел.

Ход работы

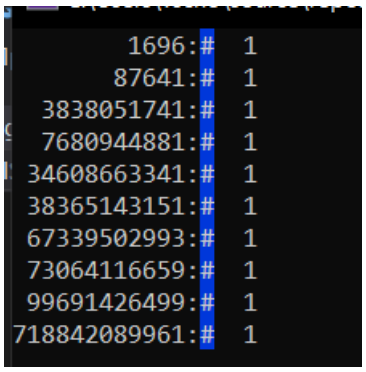


Рисунок 1 – Алгоритм фон Неймана

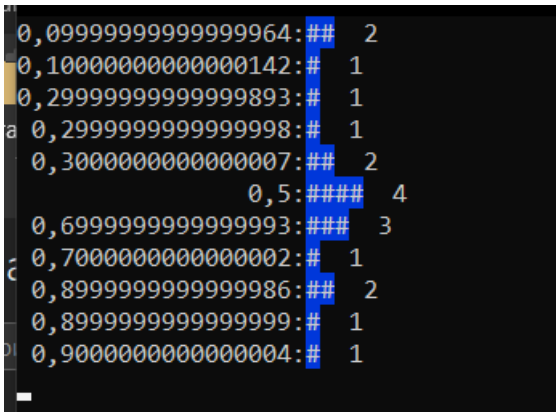


Рисунок 2 – Дробная часть многочлена

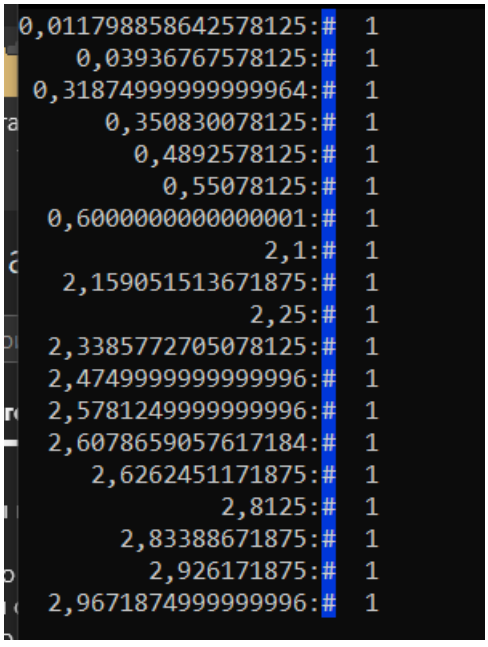
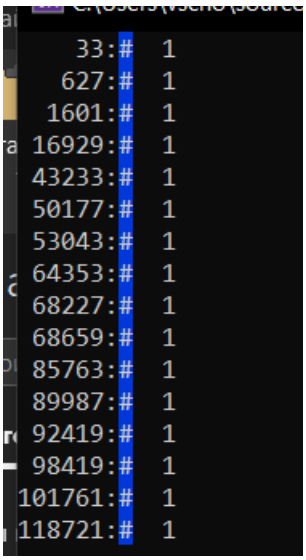


Рисунок 3 – Конгурентный способ Лемера



.Рисунок 4 – Конгурентный способ Лемера 2

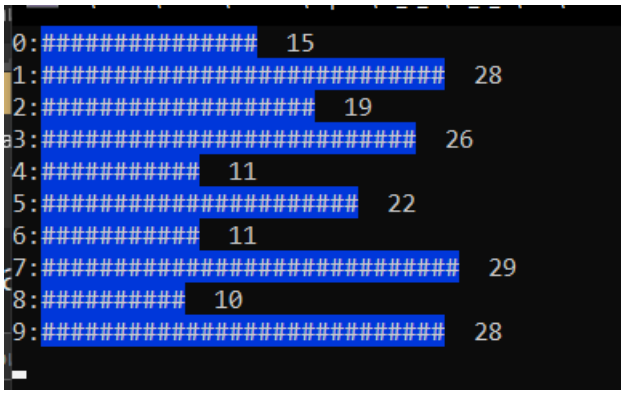


Рисунок 5 – Алгоритм Марсалиа-Зеймана

Код

«Program.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace IS_L_2
{
    class Program
    {
        static void Main(string[] args)
        {
            while (true)
            {
                Console.Clear();
                Console.WriteLine("1 - Алгоритм фон Неймана");
                Console.WriteLine("2 - Дробная часть многочлена");
                Console.WriteLine("3 - Конгурентный способ Лемера");
                Console.WriteLine("4 - Конгурентный способ Лемера 2");
                Console.WriteLine("5 - Алгоритм Марсалиа-Зеймана");
                Console.WriteLine("0 - Выход");

                int res = Convert.ToInt32(Console.ReadLine());
```

```
switch (res)
{
    case 1:
        {
            Console.Clear();

            //Console.Write("1st number: ");
            //int num = Convert.ToInt32(Console.ReadLine());

            task1 t1 = new task1();
            var result = t1.VonNeumann(342);
            PrintGistogram(result);

            Console.ReadLine();
            break;
        }
    case 2:
        {
            Console.Clear();

            task2 t2 = new task2();
            var result = t2.PolynomialFractionalPart(1.2, 1.5);
            PrintGistogram(result);

            Console.ReadLine();
            break;
        }
    case 3:
        {
            Console.Clear();

            task3 t3 = new task3();
            var result = t3.Lemair(1.5, 2.1, 3);
            PrintGistogram(result);

            Console.ReadLine();
            break;
        }
    case 4:
        {
            Console.Clear();

            task4 t4 = new task4();
            var result = t4.Lemair();
            PrintGistogram(result);

            Console.ReadLine();
            break;
        }
    case 5:
        {
            Console.Clear();

            task5 t5 = new task5();
            var result = t5.MZ(5);
            PrintGistogram(result);

            Console.ReadLine();
            break;
        }
    case 0:
        return;
    default:
        Console.WriteLine("Нет такой команды");
```

```
                    break;
                }
            }
        }

        public static void PrintGistogram<T>(List<T> nums)
        {
            nums.Sort();
            Dictionary<T, int> graf = new Dictionary<T, int>();
            for (int i = 1; i < nums.Count; i++)
            {
                if (!graf.Keys.Contains(nums[i]))
                {
                    graf.Add(nums[i], 1);
                }
                else graf[nums[i]]++;
            }

            var max = graf.Max(x => x.Key.ToString().Length);
            foreach (var item in graf)
            {
                Console.Write(item.Key.ToString().PadLeft(max));
                Console.Write(":");
                Console.BackgroundColor = ConsoleColor.DarkBlue;

                for (int i = 0; i < item.Value; i++)
                {
                    Console.Write("#");
                }

                Console.BackgroundColor = ConsoleColor.Black;
                Console.Write("  ");
                Console.WriteLine(item.Value);
            }
        }
    }
}
```

«task1.cs»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace IS_L_2
{
    class task1
    {
        private List<decimal> nums;
        public List<decimal> VonNeumann(int firstNumber)
        {
            nums = new List<decimal>();
            nums.Add(Convert.ToUInt64(firstNumber));
            for (int i = 1; i < 11; i++)
            {
                var tmp = Math.Pow(Convert.ToDouble(nums[i - 1]), 2);
                string str = tmp.ToString();
                //Console.WriteLine($"{i}: {str}");
                if (str.Length > 2 && str.Length < 10)
                {
                    str = str.Remove(str.Length - 1, 1);
                    str = str.Remove(0, 1);
                }
```

```
                else if (str.Length > 10)
                {
                    str = str.Remove(str.Length - 8, 7);
                    str = str.Remove(0, 3);
                }
                nums.Add(Convert.ToDecimal(str));
            }

            return nums;
        }

    }
}
```

«task2.cs»

```
using System;
using System.Collections.Generic;
using System.Text;

namespace IS_L_2
{
    class task2
    {
        private List<double> nums;
        public List<double> PolynomialFractionalPart(double a, double b)
        {
            nums = new List<double>();
            for (int i = 0; i < 20; i++)
                nums.Add(FractionalPart(a * i + b));

            return nums;
        }

        private double FractionalPart(double x)
        {
            double tmp = x - Math.Truncate(x);
            return tmp;
        }
    }
}
```

«task3.cs»

```
using System;
using System.Collections.Generic;
using System.Text;

namespace IS_L_2
{
    class task3
    {
        private List<double> nums;
        public List<double> Lemair(double k, double c, double m)
        {
            nums = new List<double>();
            nums.Add((k + c) % m);

            for (int i = 1; i < 20; i++)
            {
                var tmp = (nums[i - 1] * k + c);
                nums.Add(tmp % m);
            }
```

```
            return nums;
        }
    }
}
```

«task4.cs»

```
using System;
using System.Collections.Generic;
using System.Text;

namespace IS_L_2
{
    class task4
    {
        private List<double> nums;
        public List<double> Lemair()
        {
            nums = new List<double>();
            int count = 17;
            double k = 3;
            double m = Math.Pow(2, count);
            nums.Add(k % m);

            for (int i = 1; i < count; i++)
            {
                k = 3 + 8 * i;
                var tmp = (nums[i - 1] * k);
                nums.Add(tmp % m);
            }

            return nums;
        }
    }
}
```

«task5.cs»

```
using System;
using System.Collections.Generic;
using System.Text;

namespace IS_L_2
{
    class task5
    {
        private List<double> nums;
        public List<double> MZ(int n)
        {
            nums = new List<double>();
            List<int> fib = new List<int>();
            fib.Add(1);
            fib.Add(1);

            for (int i = 2; i < 1000; i++)
            {
                fib.Add(fib[i - 2] + fib[i - 1]);
            }

            for (int i = 0; i < fib.Count; i++)
            {
                string str = fib[i].ToString();
```

```
                str = str[str.Length - 1].ToString();
                fib[i] = Convert.ToInt32(str);
            }

            for (int i = 0; i < fib.Count; i += n)
            {
                nums.Add(fib[i]);
            }

            return nums;
        }
    }
}
```