

Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ» (ИУ7)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.04 Программная инженерия

ОТЧЕТ

по лабораторной работе № _3_

название:	1 енераторы псевдослучаиных чисел
	•
Дисциплина:	<u>Моделирование</u>

Студент ИУ7-72Б Е.В. Брянская (Группа) (И.О. Фамилия) (Подпись, дата) Преподаватель И.В. Рудаков (И.О. Фамилия) (Подпись, дата)

1 Задание

Программным способом сгенерировать последовательности по 1000 одноразрядных/двухразрядных/трёхразрядных целых чисел. А также найти готовые файлы со случайными числами.

Последовательности необходимо получить алгоритмическим и табличным способом.

Определить и вывести на экран количественный критерий оценки случайности этих последовательностей.

Также дать возможность пользователю ввести последовательность из 10 чисел с указанием количества разрядов. И для неё определить количественный критерий и вывести в соответствующем поле.

2 Теоретическая часть

Существует три основных способа получения последовательностей случайных чисел:

- 1. аппаратный;
- 2. табличный (файловый);
- 3. алгоритмический.

В рамках лабораторной работы рассмотрены алгоритмический и табличный способы.

2.1 Алгоритмический способ

Этот подход основан на базе специальных алгоритмов. К ним относятся:

- метод серединных произведений;
- метод перемешивания;
- линейный конгруэнтный метод.

Было принято решение взять последний для генерации последовательности псевдослучайных чисел.

В этом методе каждое следующее число рассчитывается на основе предыдущего по формуле (1).

$$R_{n+1} = (a \cdot R_n + b) \bmod N, \ n \ge 1 \tag{1}$$

где a, b – коэффициенты, N – модуль.

Для качественного генератора требуется подобрать подходящие коэффициенты. Например, в таблице 2.1 приведены некоторые из них.

a	b	N
106	1283	6075
430	2531	11979
84589	15989	217728
1103515245	12345	2^{31}

Таблица 2.1 — Примеры коэффициентов

2.2 Табличный способ

В качестве источника случайных чисел используют специально заранее составленные таблицы, содержащие проверенные данные.

2.3 Критерий оценки

В рамках лабораторной работы в качестве статистической оценки используется критерий «хи-квадрат» (χ^2 -критерий).

Привлекая этот критерий можно оценить, удовлетворяет ли генератор требованию равномерного распределения или нет.

Используется статистика, представленная формулой (2).

$$V = \frac{1}{n} \sum_{s=min}^{max} \left(\frac{Y_s^2}{p_s}\right) - n \tag{2}$$

где n - длина последовательности, min/max - границы, в пределах которых находятся элементы последовательности, Y_s - число повторений числа s, $p=\frac{1}{max-min}$.

Вычисляется квантиль хи-квадрат от вычисленного V. Если полученное значение будет меньше 0.1 или больше 0.9, то эти числа считаются недостаточно случайными. Если же нет, то последовательность принимается как случайная.

3 Результаты работы программы

Интерфейс предполагает вывод первых 10 элементов из 1000 каждой последовательности (за исключением той, которую введёт пользователь, так как в этом случае длина фиксирована и равна 10).

Результат работы программы приведен на рисунке 3.1. Можно заметить, что значения критерия в любом из столбцов находится в интервале от 0.1 до 0.9, что позволяет считать последовательности случайными.

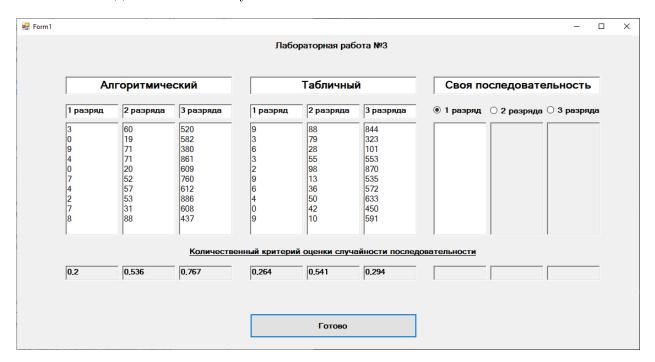


Рисунок 3.1 - Пример 1

На рисунках 3.2-3.4 демонстрируются примеры пользовательских последовательностей.

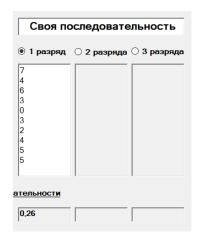


Рисунок 3.2 — Пример 2. Введенная пользователем последовательность из 10 чисел

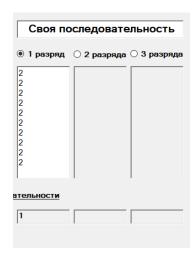


Рисунок 3.3 — Пример 3. Все числа одинаковые



Рисунок 3.4 — Пример 4. Подряд идущие числа

4 Код программы

На Листинге 1 представлены основные методы.

Листинг 1 — Основные методы

```
namespace calculations {
2
     class MathCalc {
       public \ static \ double \ Criteria (int[] \ arr , \ int \ elMin \ , \ int \ elMax) \ \{
3
         double y = 0, p = 1.0 / (elMax - elMin);
4
5
         for (int i = e|Min; i < e|Max; i++)
6
            y += Math.Pow(arr.Count(x => x == i), 2) / p;
7
         y = y / (double) arr. Length - arr. Length;
8
9
          return ChiSquared .CDF(e|Max - e|Min - 1, y);
10
       }
11
     }
12
     class CongruentMethod {
13
       private int a { get; }
14
       private int b { get; }
15
```

```
16
       private int N { get; }
       private int r;
17
18
       public CongruentMethod(int seed) {
19
20
         a = 1103515245;
         b = 12345;
21
         r = seed;
22
23
       }
24
25
       public int next(int start, int end){
26
         r = a * r + b;
         return (int)((uint)r >> 16) % (end - start) + start;
27
28
       }
29
     }
30 }
31
  namespace lab03 {
32
     public partial class Form1 : Form {
33
34
       private void fillAlg(int start, int end, string filename, string field , string fieldK){
35
         int[] arr = new int[1000];
36
         Congruent Method a \mid g = new Congruent Method (rnd. Next());
37
38
         for (int i = 0; i < arr.Length; i++)
39
40
           arr[i] = alg.next(start, end);
41
42
         fillCriteria (fieldK, MathCalc.Criteria (arr, start, end));
43
       }
44
45
       private void useTable(int start, int end, string filename, string field, string fieldK)
         int[] data = new int[1000], arr = new int[1000];
46
         int num = rnd.Next(0, 1000), j = num;
47
48
         _getArrFromFile(filename, ref data);
49
50
         for (int i = 0; i < arr.Length; i++) {
51
52
           arr[i] = data[j];
53
           j = (j + 1) \% data Length;
54
55
         fillCriteria (fieldK, MathCalc. Criteria (arr, start, end));
56
       }
57
       private void processUser(){
58
59
         int[] arr = new int[10];
60
         int flag , start , end;
         string fieldK;
61
62
63
         try {
64
           getArrFromUser(out flag, ref arr);
65
         }
```

```
66
         catch (Exception e){
67
68
         }
69
         switch (flag){
70
71
           case 1:
72
             start = 0;
73
             end = 10;
             fieldK = "criteriaO1";
74
75
             break;
           case 2:
76
77
             start = 10;
             end = 100;
78
79
             fieldK = "criteriaO2";
80
             break;
81
           default:
82
             start = 100;
             end = 1000;
83
84
             fieldK = "criteriaO3";
             break;
85
         }
86
         fillCriteria (fieldK, MathCalc.Criteria (arr, start, end));
87
88
    }
89
90 }
```