



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: Генерация случайных чисел

Дисциплина: Моделирование

Студент

ИУ7-74Б
(Группа)

(Подпись, дата)

Д.В.Жабин
(И.О.Фамилия)

Преподаватель

(Подпись, дата)

И.В.Рудаков
(И.О.Фамилия)

Москва, 2022

Задание

Необходимо сгенерировать последовательности из 1000 целых чисел (одноразрядных, двухразрядных, трёхразрядных) алгоритмическим и табличным способом.

Определить и вывести на экран значения критерия оценки случайности этих последовательностей.

Также дать возможность пользователю ввести последовательность чисел, для неё также определить критерий случайности и вывести на экран.

Способы получения последовательностей случайных чисел

Выделяют 3 основных способа:

1. Аппаратный – случайные числа вырабатываются специальной электронной приставкой, то есть генератором случайных чисел. Как правило это практически любое внешнее устройство компьютера. Реализация этого способа не требует дополнительных вычислительных операций по выработке чисел, а необходима только операция обращения к этому внешнему устройству. Физическая основа - как правило шум на проводниковых и полупроводниковых приборах. Состоит из:
 - Источник шума
 - Ключевая схема
 - Формировщик импульсов
 - Пересчётная схема
2. Табличный – формируется таблица и записывается в память. Числа заранее проверены на случайность и последовательность можно многократно использовать. Недостаток – количество чисел ограничено.
3. Алгоритмический – однократная проверка, можно многократно воспроизводить последовательность, относительно малое место в оперативной памяти, не используются внешние устройства. Недостаток – запас чисел ограничен периодом, требуются затраты машинного времени.

Линейный конгруэнтный метод

В данной работе был выбран линейный конгруэнтный метод в качестве алгоритмического. Для осуществления генерации чисел данным методом, необходимо задать 4 числа:

$$\begin{aligned} m &> 0, \text{ модуль} \\ 0 &\leq a \leq m, \text{ множитель} \\ 0 &\leq c \leq m, \text{ приращение} \\ 0 &\leq X_0 \leq m, \text{ начальное число} \end{aligned}$$

Последовательность случайных чисел генерируется при помощи формулы:

$$X_{n+1} = (aX_n + c) \mod m \quad (1)$$

Критерий случайности последовательности чисел

В данной лабораторной работе используется следующий критерий оценки случайности:

$$C = \sum_{i=1}^N -p_i * \log_N p_i \quad (2)$$

где N – длина последовательности чисел, p_i – отношение количества вхождений очередного числа x_i в последовательность к её длине N .

Таким образом, чем больше количество вхождений числа в последовательность ($freq$), тем ближе значение ΔC на очередном шаге к $\frac{freq}{N}$. В случае, когда $freq = N$ (последовательность состоит из одинаковых чисел), $\log_N p_i$ окажется равным 0, а следовательно, и значение критерия случайности $C = 0$.

Текст программы

В листинге 1 приведен код разработанной программы.

Листинг 1: Текст программы

```
from PyQt5 import QtWidgets, uic
from PyQt5.QtWidgets import QTableWidgetItem
from math import log
from itertools import islice
import sys

STEP = 10

class Window(QtWidgets.QMainWindow):
    def __init__(self):
        QtWidgets.QWidget.__init__(self)
        uic.loadUi("main.ui", self)
        self.fill_alg.clicked.connect(lambda: fillAlgNums(self))
        self.fill_table.clicked.connect(lambda: fillTableNums(self))
        self.manual_input.returnPressed.connect(lambda: calcManual(self))
        self.meas_alg_1.setReadOnly(True)
        self.meas_alg_2.setReadOnly(True)
        self.meas_alg_3.setReadOnly(True)
        self.meas_table_1.setReadOnly(True)
        self.meas_table_2.setReadOnly(True)
        self.meas_table_3.setReadOnly(True)
        self.meas_manual.setReadOnly(True)
        self.alg_table.setEditTriggers(QtWidgets.QAbstractItemView.NoEditTriggers)
        self.table_table.setEditTriggers(QtWidgets.QAbstractItemView.NoEditTriggers)
        self.line_num = 0

        for i in range(10):
            self.alg_table.insertRow(i)
        for i in range(10):
            self.table_table.insertRow(i)
```

```

def randomCriteria(seq):
    N = len(seq)
    if N == 0:
        return 0

    numsFreq = dict()
    for x in seq:
        if x not in numsFreq.keys():
            numsFreq.update({x: 1})
        else:
            numsFreq[x] += 1

    crit = 0
    for x in numsFreq.keys():
        p = numsFreq[x] / N
        crit -= p * log(p, N)
    return crit

class LinearCongruent:
    m = 2**32
    a = 1664525
    c = 1013904223
    _cur = 1

    def next(self):
        self._cur = (self.a * self._cur + self.c) % self.m
        return self._cur

def fillAlgNums(win):
    table = win.alg_table
    one_digit = [generator.next() % 9 + 1 for i in range(1000)]
    two_digits = [generator.next() % 90 + 10 for i in range(1000)]
    three_digits = [generator.next() % 900 + 100 for i in range(1000)]

    for i in range(10):
        item = QTableWidgetItem(str(one_digit[i * STEP]))
        table.setItem(i, 0, item)
    for i in range(10):
        item = QTableWidgetItem(str(two_digits[i * STEP]))
        table.setItem(i, 1, item)
    for i in range(10):
        item = QTableWidgetItem(str(three_digits[i * STEP]))
        table.setItem(i, 2, item)

    crit_one = randomCriteria(one_digit)
    crit_two = randomCriteria(two_digits)
    crit_three = randomCriteria(three_digits)
    win.meas_alg_1.setText('{:.4%}'.format(crit_one))
    win.meas_alg_2.setText('{:.4%}'.format(crit_two))
    win.meas_alg_3.setText('{:.4%}'.format(crit_three))

def fillTableNums(win):
    table = win.table_table

```

```

numbers = set()
with open('nums.txt') as file:
    lines = islice(file, win.line_num, None)
    for l in lines:
        numbers.update(set(l.split(" ")[1:-1]))
        win.line_num += 1
    if len(numbers) > 3000:
        break
    numbers.remove("")
    numbers = list(numbers)[:3000]

one_digit = [int(i) % 9 + 1 for i in numbers[:1000]]
two_digits = [int(i) % 90 + 10 for i in numbers[1000:2000]]
three_digits = [int(i) % 900 + 100 for i in numbers[2000:3000]]

for i in range(10):
    item = QTableWidgetItem(str(one_digit[i * STEP]))
    table.setItem(i, 0, item)
for i in range(10):
    item = QTableWidgetItem(str(two_digits[i * STEP]))
    table.setItem(i, 1, item)
for i in range(10):
    item = QTableWidgetItem(str(three_digits[i * STEP]))
    table.setItem(i, 2, item)

crit_one = randomCriteria(one_digit)
crit_two = randomCriteria(two_digits)
crit_three = randomCriteria(three_digits)
win.meas_table_1.setText(' {:.4%}'.format(crit_one))
win.meas_table_2.setText(' {:.4%}'.format(crit_two))
win.meas_table_3.setText(' {:.4%}'.format(crit_three))

def calcManual(win):
    input = win.manual_input.text().split(" ")
    seq = []
    for x in input:
        try:
            int(x)
        except ValueError:
            continue
        else:
            seq.append(int(x))

    crit = randomCriteria(seq)
    win.meas_manual.setText(' {:.4%}'.format(crit))

if __name__ == "__main__":
    generator = LinearCongruent()
    app = QtWidgets.QApplication(sys.argv)
    w = Window()
    w.show()
    sys.exit(app.exec_())

```

Примеры работы

На рисунках 1-3 приведены примеры работы программы с разным вводом пользователя.

The screenshot shows a window titled "RandomNumbers" with two main sections: "Алгоритмические значения" (Algorithmic values) and "Табличные значения" (Table values). Each section contains a table with 10 rows and 3 columns representing ranges: "0 - 9", "10 - 99", and "100 - 999". Below the tables, there are two groups of "Критерий случайности" (Randomness criteria) with input fields for percentages and "Заполнить" (Fill) buttons. A "Ручной ввод" (Manual input) section includes a text field with the sequence "7 23 14 4 1 5 6 3 2 1" and a corresponding "Критерий случайности" field showing "93.9794%".

	0 - 9	10 - 99	100 - 999
1	1	82	492
2	3	78	602
3	6	68	456
4	2	50	466
5	2	40	112
6	7	28	298
7	9	18	304
8	7	84	846
9	1	62	920
10	7	60	854

	0 - 9	10 - 99	100 - 999
1	2	47	375
2	7	90	383
3	3	33	181
4	8	11	437
5	8	85	293
6	7	78	530
7	7	26	673
8	6	79	109
9	1	51	421
10	8	92	116

Критерий случайности

31.7617%	64.5736%	91.0292%	31.7898%	64.5352%	90.9068%
----------	----------	----------	----------	----------	----------

Заполнить

Ручной ввод

7 23 14 4 1 5 6 3 2 1

Критерий случайности

93.9794%

Рис. 1: Пример 1. Разные числа

RandomNumbers

Алгоритмические значения

	0 - 9	10 - 99	100 - 999
1	6	44	304
2	3	58	434
3	6	18	820
4	7	32	454
5	1	88	420
6	7	18	418
7	1	82	432
8	6	58	682
9	1	98	640
10	3	58	530

Табличные значения

	0 - 9	10 - 99	100 - 999
1	1	21	933
2	2	15	898
3	9	68	173
4	3	52	785
5	1	66	264
6	9	13	619
7	3	93	540
8	7	94	973
9	6	43	852
10	7	11	295

Критерий случайности

31.7404%

64.3752%

91.1747%

Заполнить

Критерий случайности

31.7967%

64.4163%

90.8707%

Заполнить

Ручной ввод

5 5 1 3 5 7 6 5 8 2 5

Критерий случайности

75.9176%

Рис. 2: Пример 2. Несколько повторяющихся чисел

RandomNumbers

Алгоритмические значения

	0 - 9	10 - 99	100 - 999
1	6	60	720
2	9	58	790
3	1	60	636
4	1	80	218
5	7	26	596
6	3	60	638
7	7	24	820
8	9	50	158
9	2	14	376
10	6	94	782

Табличные значения

	0 - 9	10 - 99	100 - 999
1	4	85	916
2	8	67	610
3	5	32	406
4	6	34	160
5	6	37	525
6	3	52	425
7	4	16	620
8	4	56	501
9	3	67	729
10	2	33	953

Критерий случайности

31.6587%

64.3925%

90.9844%

Заполнить

Критерий случайности

31.7935%

64.4947%

91.0984%

Заполнить

Ручной ввод

7 7 7 7 7 7 7 7 7

Критерий случайности

0.0000%

Рис. 3: Пример 3. Одинаковые числа