Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

**ТЕОРИЯ ПСЕВДОСЛУЧАЙНЫХ ГЕНЕРАТОРОВ**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ**

студентки 4 курса 431 группы

факультета компьютерных наук и информационных технологий

*Змеевой Вероники Александровны*

фамилия, имя, отчество

Научный руководитель

Ст. преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_ И.И. Слеповичев

подпись, дата

Саратов 2024

**СОДЕРЖАНИЕ**

[**1 Постановка задачи** 3](#_Toc166788867)

[**2 Тестирование критериев для каждой ПСЧ из практической работы** 5](#_Toc166788868)

[**2.1** **Линейный конгруэнтный метод** 5](#_Toc166788869)

[**2.2 Аддитивный метод** 6](#_Toc166788870)

[**2.3 Пятипараметрический метод** 8](#_Toc166788871)

[**2.4 Регистр сдвига с обратной связью (РСЛОС)** 9](#_Toc166788872)

[**2.5 Нелинейная комбинация РСЛОС** 10](#_Toc166788873)

[**2.6 Вихрь Мерсенна** 12](#_Toc166788874)

[**2.7 RC4** 13](#_Toc166788875)

[**2.8 ГПСЧ на основе RSA** 15](#_Toc166788876)

[**2.9 Алгоритм Блюма-Блюма-Шуба** 17](#_Toc166788877)

[**3 Таблица с результатами проверки ПСП различными критериями** 19](#_Toc166788878)

[**ПРИЛОЖЕНИЕ** 20](#_Toc166788879)

# **1 Постановка задачи**

***Цель***

1. Сгенерировать псевдослучайную последовательность заданным методом.

2. Исследовать полученную псевдослучайную последовательность на случайность.

***Исходные данные***

Исходными данными для лабораторных занятий являются метод генерации псевдослучайных чисел, диапазон генерации случайных чисел, функция распределения, которой должны подчиняться случайные числа, количество генерируемых чисел.

**Задачи**

1. Сгенерировать последовательность из 10000 случайных чисел из диапазона [0,1]. Исходной программой для генерации ПСЧ может быть программа, созданная в рамках практической работы по данному курсу.
2. Протестировать статистические свойства последовательности псевдослучайных чисел:
   1. Вычислить математическое ожидание последовательности;
   2. Вычислить среднеквадратичное отклонение последовательности;
   3. Сравните полученные оценки с заданными в пп. 1 параметрами. Постройте графики зависимостей оценок от объема выборки. Оцените относительные погрешности для какой-либо одной выборки.
   4. Вычислить значение и дать ответ на вопрос удовлетворяет ли ППСЧ
      1. Критерию хи-квадрат;
      2. Критерию серий;
      3. Критерию интервалов;
      4. Критерию разбиений;
      5. Критерию перестановок;
      6. Критерию монотонности;
      7. Критерию конфликтов.

# **2 Тестирование критериев для каждой ПСЧ из практической работы**

**2.1** **Линейный конгруэнтный метод**

**Использованные параметры:**

prng.exe -g lc -i 1021,376,7,13 -n 10000 -f lc.dat

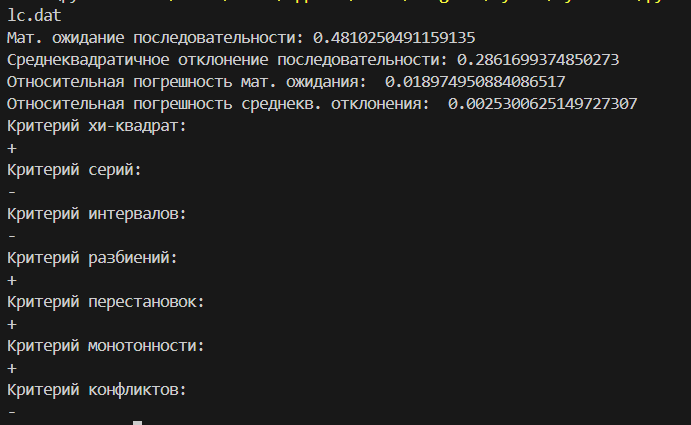
**Пример работы программы:**

Мат. ожидание последовательности: 0.4810250491159135

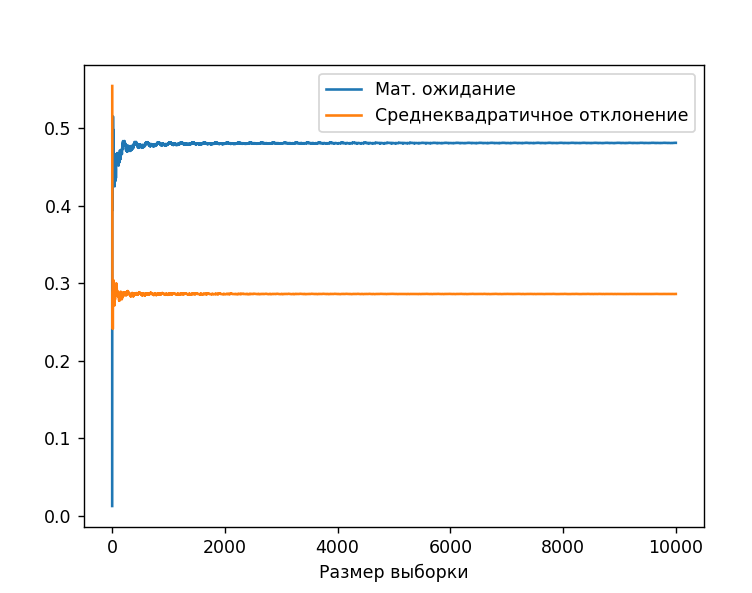
Среднеквадратичное отклонение последовательности: 0.2861699374850273

Относительная погрешность мат. ожидания: 0.018974950884086517

Относительная погрешность среднекв. отклонения: 0.0025300625149727307



**Графики зависимостей оценок от объема выборки:**



**2.2 Аддитивный метод**

**Использованные параметры:**

prng.exe -g add -i 100,24,55,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59 -n 10000 -f add.dat

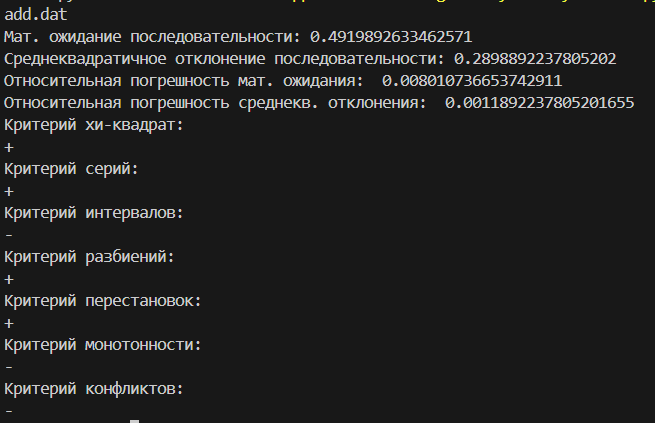
**Пример работы программы:**

Мат. ожидание последовательности: 0.4919892633462571

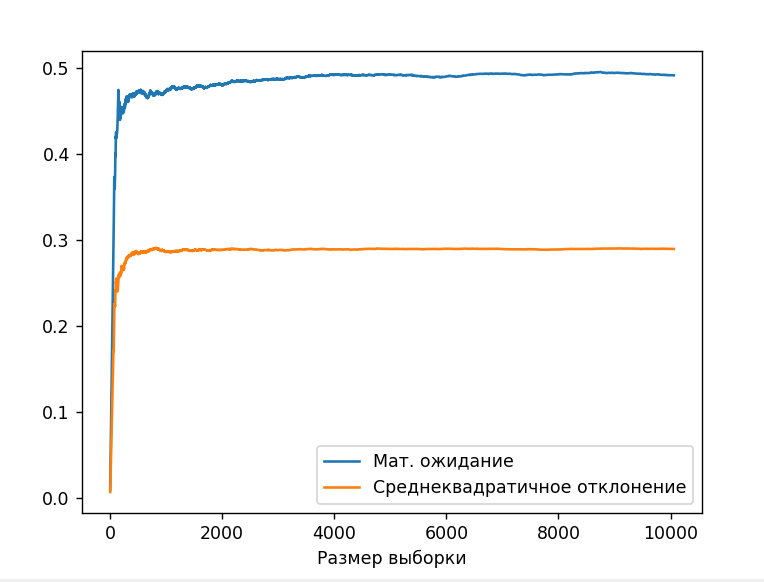
Среднеквадратичное отклонение последовательности: 0.2898892237805202

Относительная погрешность мат. ожидания: 0.008010736653742911

Относительная погрешность среднекв. отклонения: 0.0011892237805201655



**Графики зависимостей оценок от объема выборки:**



**2.3 Пятипараметрический метод**

**Использованные параметры**:

prng.exe -g 5p -i 89,7,13,24,10,764 -n 10000 -f 5p.dat

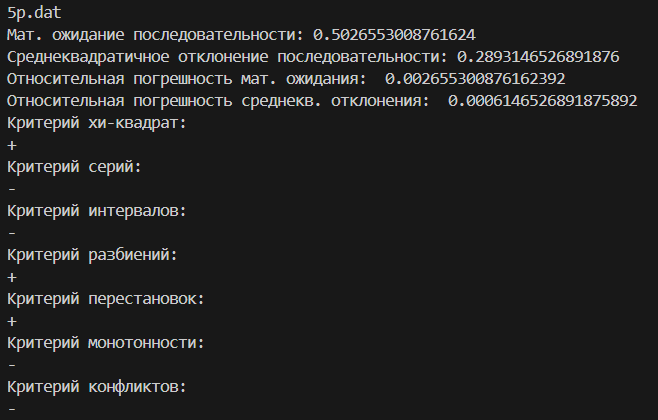
**Пример работы программы:**

Мат. ожидание последовательности: 0.5026553008761624

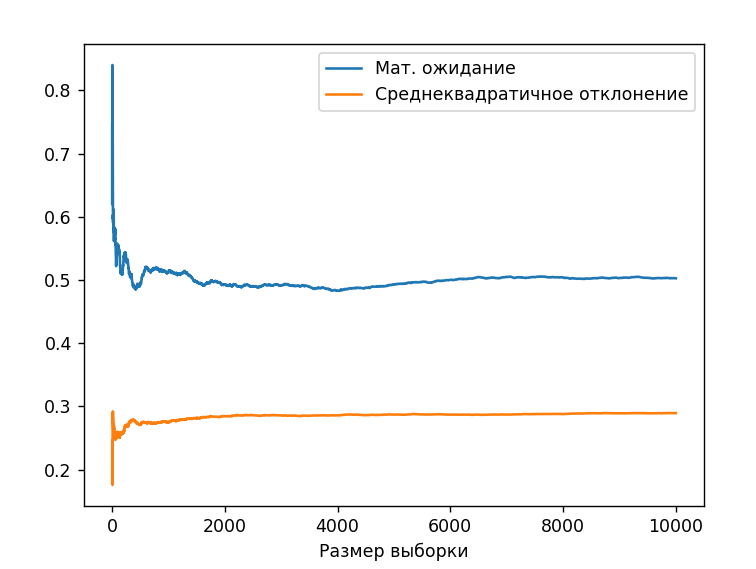
Среднеквадратичное отклонение последовательности: 0.2893146526891876

Относительная погрешность мат. ожидания: 0.002655300876162392

Относительная погрешность среднекв. отклонения: 0.0006146526891875892



**Графики зависимостей оценок от объема выборки:**



**2.4 Регистр сдвига с обратной связью (РСЛОС)**

**Использованные параметры:**

prng.exe -g lfsr -i 110110010,0000010011 -n 10000 -f lfsr.dat

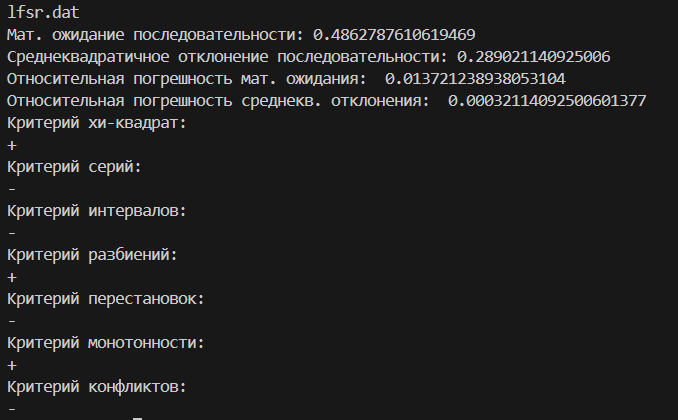
**Пример работы программы:**

Мат. ожидание последовательности: 0.4862787610619469

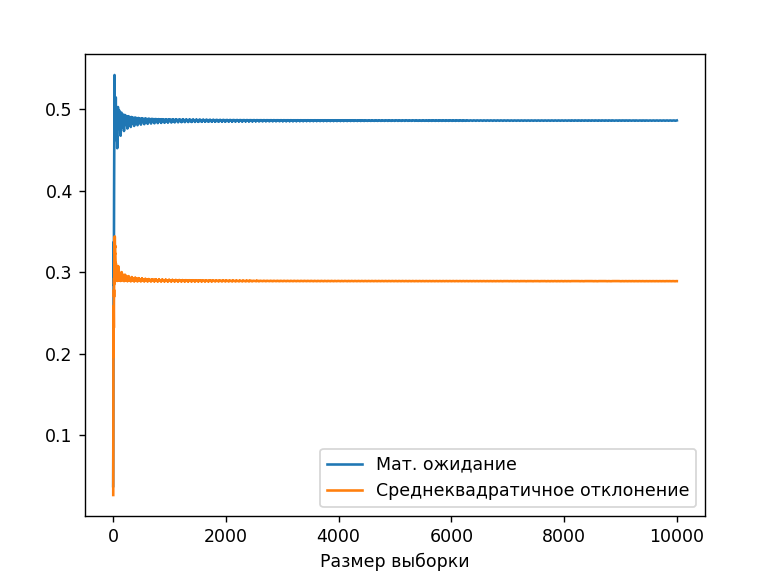
Среднеквадратичное отклонение последовательности: 0.289021140925006

Относительная погрешность мат. ожидания: 0.013721238938053104

Относительная погрешность среднекв. отклонения: 0.00032114092500601377



**Графики зависимостей оценок от объема выборки:**



**2.5 Нелинейная комбинация РСЛОС**

**Использованные параметры**:

prng.exe -g nfsr -i 100000001001001,0011000000,101011001001001,9,25,60,45 -n 10000 -f nfsr.dat

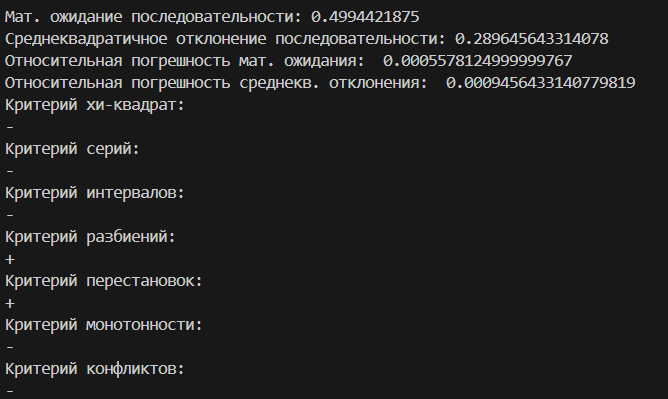
**Пример работы программы:**

Мат. ожидание последовательности: 0.4994421875

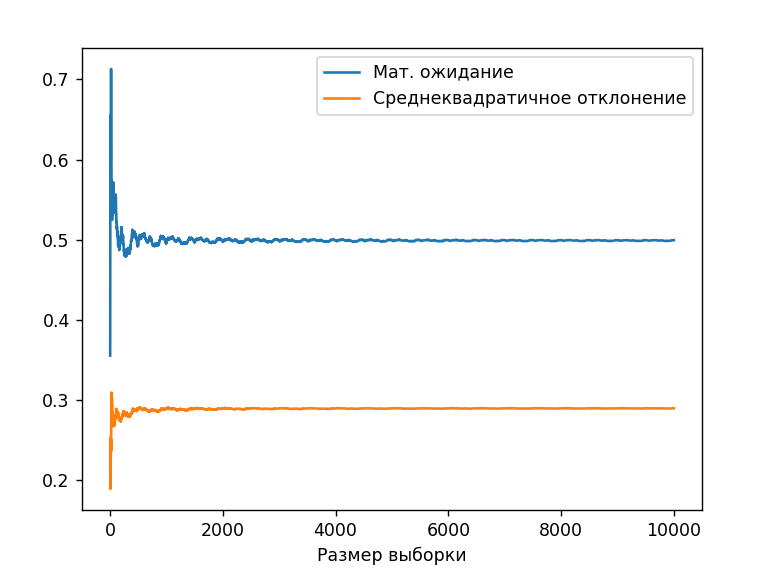
Среднеквадратичное отклонение последовательности: 0.289645643314078

Относительная погрешность мат. ожидания: 0.0005578124999999767

Относительная погрешность среднекв. отклонения: 0.0009456433140779819



**Графики зависимостей оценок от объема выборки:**



**2.6 Вихрь Мерсенна**

**Использованные параметры**:

prng.exe -g mt -i 100,1313 -n 10000 -f mt.dat

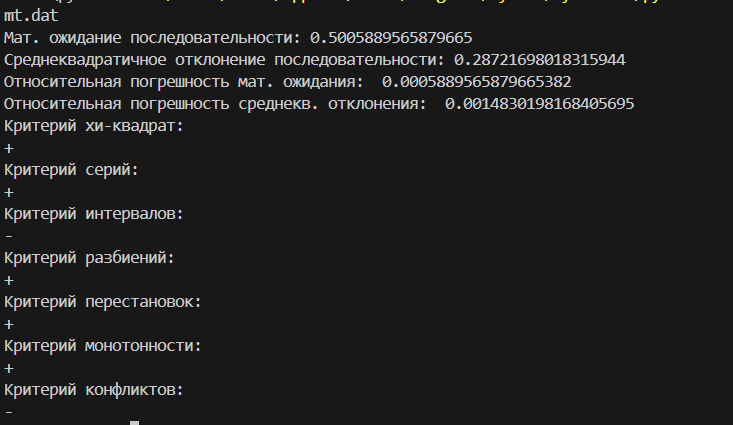
**Пример работы программы:**

Мат. ожидание последовательности: 0.5005889565879665

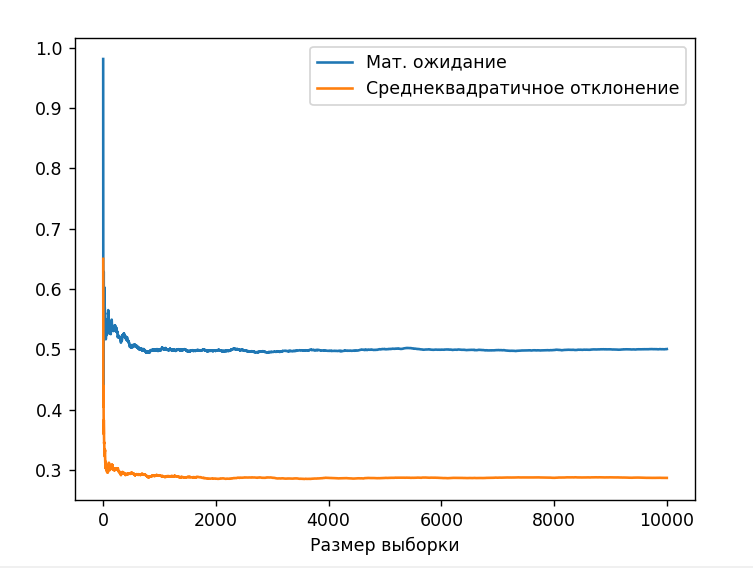
Среднеквадратичное отклонение последовательности: 0.28721698018315944

Относительная погрешность мат. ожидания: 0.0005889565879665382

Относительная погрешность среднекв. отклонения: 0.0014830198168405695



**Графики зависимостей оценок от объема выборки:**



**2.7 RC4**

**Использованные параметры**:

prng.exe -g rc4 -i 40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,92,93,94,95,96,97,98,99,100,101,102,103,104,105,106,107,108,109,110,111,112,113,114,115,116,117,118,119,120,121,122,123,124,125,126,127,128,129,130,131,132,133,134,135,136,137,138,139,140,141,142,143,144,145,146,147,148,149,150,151,152,153,154,155,156,157,158,159,160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175,176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191,192,193,194,195,196,197,198,199,200,201,202,203,204,205,206,207,208,209,210,211,212,213,214,215,216,217,218,219,220,221,222,223,224,225,226,227,228,229,230,231,232,233,234,235,236,237,238,239,240,241,242,243,244,245,246,247,248,249,250,251,252,253,254,255,256,257,258,259,260,261,262,263,264,265,266,267,268,269,270,271,272,273,274,275,276,277,278,279,280,281,282,283,284,285,286,287,288,289,290,291,292,293,294,295 -n 10000 -f rc4.dat

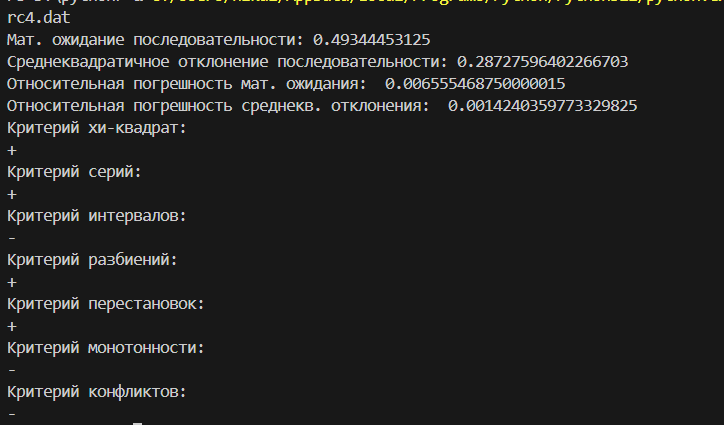
**Пример работы программы:**

Мат. ожидание последовательности: 0.49344453125

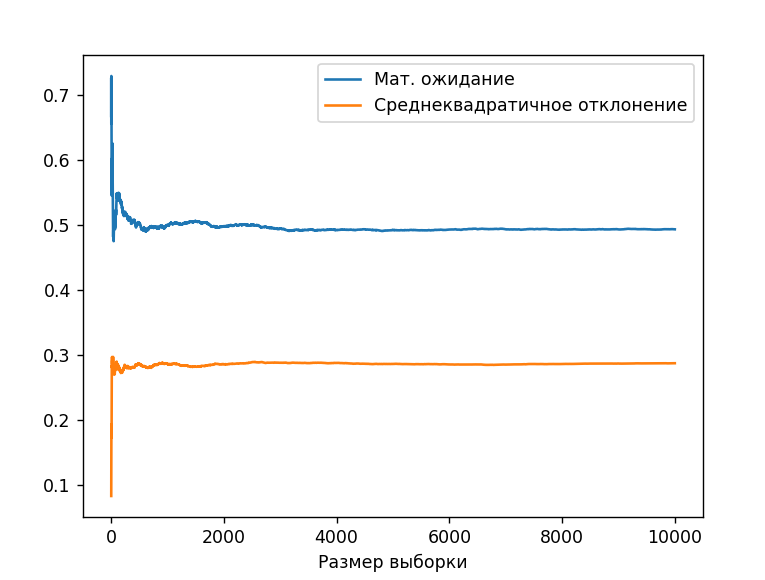
Среднеквадратичное отклонение последовательности: 0.28727596402266703

Относительная погрешность мат. ожидания: 0.006555468750000015

Относительная погрешность среднекв. отклонения: 0.0014240359773329825



**Графики зависимостей оценок от объема выборки:**



**2.8 ГПСЧ на основе RSA**

**Использованные параметры**:

prng.exe -g rsa -i 12709189,53,10,245 -n 10000 -f rsa.dat

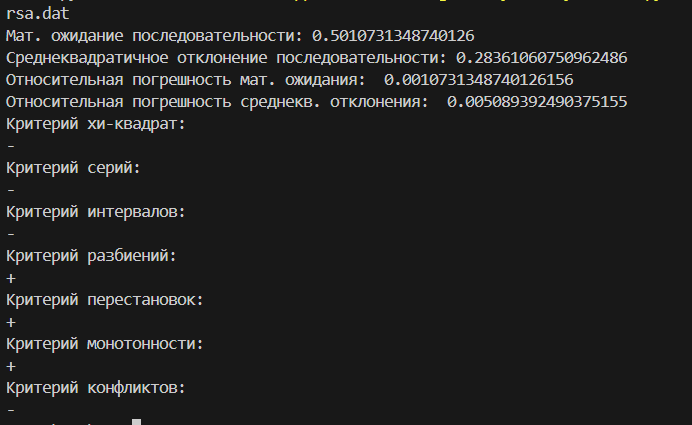
**Пример работы программы:**

Мат. ожидание последовательности: 0.5010731348740126

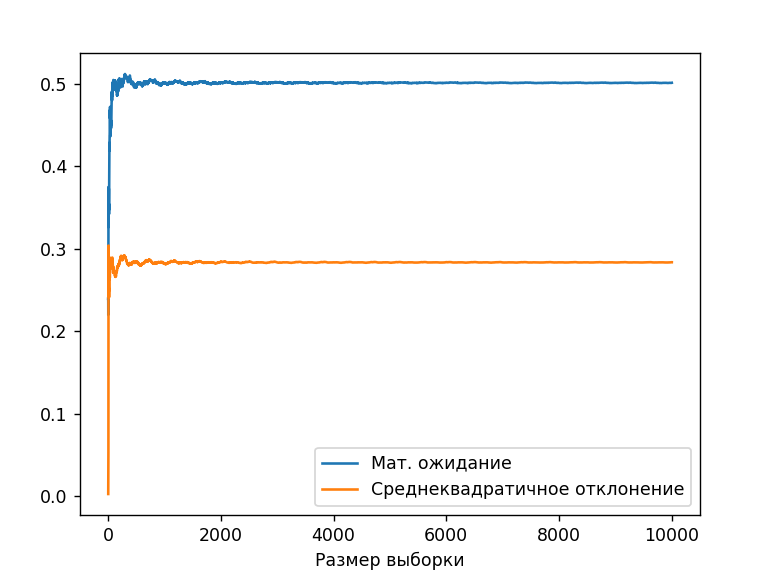
Среднеквадратичное отклонение последовательности: 0.28361060750962486

Относительная погрешность мат. ожидания: 0.0010731348740126156

Относительная погрешность среднекв. отклонения: 0.005089392490375155



**Графики зависимостей оценок от объема выборки:**



**2.9 Алгоритм Блюма-Блюма-Шуба**

**Использованные параметры**:

prng.exe -g bbs -i 1562341,10 -n 10000 -f bbs.dat

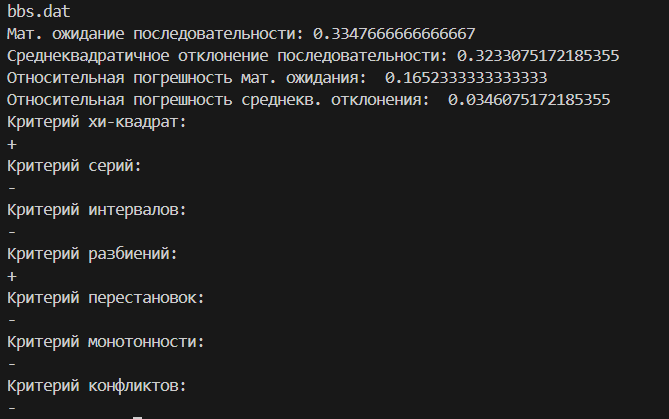
**Пример работы программы:**

Мат. ожидание последовательности: 0.3347666666666667

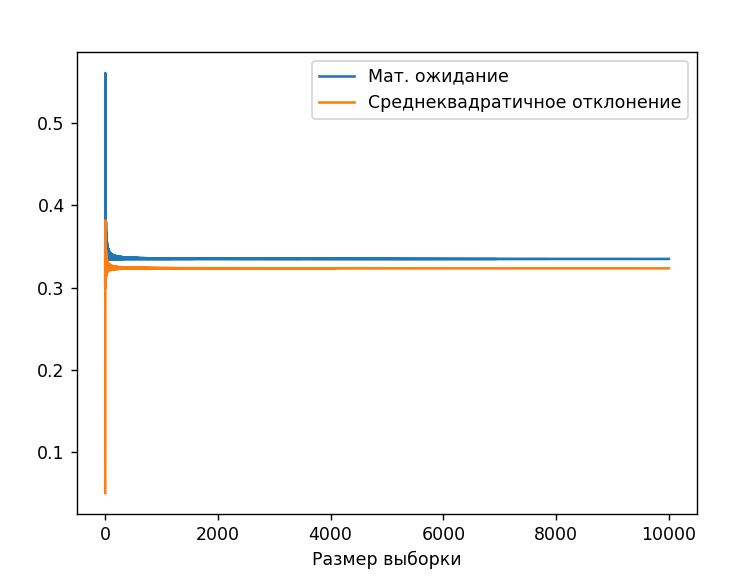
Среднеквадратичное отклонение последовательности: 0.3233075172185355

Относительная погрешность мат. ожидания: 0.1652333333333333

Относительная погрешность среднекв. отклонения: 0.0346075172185355



**Графики зависимостей оценок от объема выборки:**



# **3 Таблица с результатами проверки ПСП различными критериями**

Таблица 1. Результаты проверки ПСП различными критериями

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **lc** | **add** | **5p** | **lfsr** | **nfsr** | **mt** | **rc4** | **rsa** | **bbs** |
| **xи-квадрат** | + | + | + | + | − | + | + | − | + |
| **серий** | − | + | − | − | − | + | + | − | − |
| **интервалов** | − | − | − | − | − | − | − | − | − |
| **разбиений** | + | + | + | + | + | + | + | + | + |
| **перестановок** | + | + | + | − | + | + | + | + | − |
| **монотонности** | + | − | − | + | − | + | − | + | − |
| **конфликтов** | − | − | − | − | − | − | − | − | − |

# 

# **ПРИЛОЖЕНИЕ**

**Программа для лабораторной работы**

import numpy as np

from matplotlib import pyplot as plt

import scipy.stats as sps

import math

def read\_from\_file(filename):

    with open(filename) as f:

        lines = f.readlines()

    input = "".join(lines)

    a = input.split(",")

    sequence = [int(elem) for elem in a if len(elem) != 0]

    return sequence

def plot\_distribution(nums):

    size = []

    mean\_array = []

    std\_array = []

    for i in range(len(nums)):

        size.append(i+1)

        mean\_array.append(np.mean(nums[:i+1]))

        std\_array.append(sps.tstd(nums[:i+1]))

    plt.plot(size, mean\_array, label="Мат. ожидание")

    plt.plot(size, std\_array, label="Среднеквадратичное отклонение")

    plt.xlabel("Размер выборки")

    plt.legend()

    plt.show()

def pogr(seq):

    GOAL\_MEAN = 0.5

    GOAL\_STD = 0.2887

    cur\_value\_mean = np.mean(seq)

    cur\_value\_std = sps.tstd(seq)

    print("Относительная погрешность мат. ожидания: ", abs(GOAL\_MEAN - cur\_value\_mean))

    print("Относительная погрешность среднекв. отклонения: ", abs(GOAL\_STD - cur\_value\_std))

def transform\_lst\_nums(num\_lst):

    max\_elem = max(num\_lst) + 1

    return [num / max\_elem for num in num\_lst]

def chi\_2(seq, alpha=0.05, lst\_=None, exp=None, param=None):

    if param is None:

        param = len(np.unique(seq))

    if lst\_ is None:

        \_, lst\_ = np.unique(seq, return\_counts=True)

    if exp is None:

        exp = np.array([len(seq) / param] \* param)

    chi, stat = np.sum((lst\_ - exp) \*\* 2 / exp), sps.chi2.ppf(1 - alpha, param - 1)

    if chi > stat:

        return "-"

    else:

        return "+"

def series(seq):

    d = 16

    alpha = 0.05

    param = d \*\* 2

    res = np.zeros(param, dtype=int)

    for j in range(len(seq) // 2):

        res[int(seq[2 \* j] \* d) \* d + int(seq[2 \* j + 1] \* d)] += 1

    return chi\_2(seq, alpha, res, np.full(param, len(seq) / (2 \* param)), param)

def intervals(seq):

    d = 16

    j, s, emp = -1, 0, 8 \* [0]

    t = 7

    n = len(seq)

    interval\_amount = n / 10

    half = 0.5

    theor = [interval\_amount \* half \* (1.0 - half) \*\* r for r in range(t)] + [interval\_amount \* (1.0 - half) \*\* t]

    while s != interval\_amount and j != n:

        j += 1

        r = 0

        while j != n and seq[j] < d / 2:

            j += 1

            r += 1

        emp[min(r, t)] += 1

        s += 1

    if j == n:

        return "-"

    return chi\_2(seq, 0.05 ,theor, emp, t + 1)

def partitions(seq):

    alpha = 0.05

    n = 100

    param = int(10000 / n)

    r = np.array([0] \* (param + 1))

    for i in range(n):

        r[len(np.unique(seq[param \* i : param \* (i + 1)]))] += 1

    p = []

    s = 1

    for i in range(param + 1):

        d = 100

        p\_i = d

        for j in range(1, i):

            p\_i \*= d - j

        p.append(p\_i / pow(d, param) \* s)

    dk\_lst = np.array([math.comb(param + i - 1, i) / pow(d, param) for i in range(param + 1)])

    return chi\_2(seq, alpha, dk\_lst[1:], p[1:], param)

def permutations(seq):

    alpha = 0.05

    t = 10

    n = len(seq)

    dict = {}

    param = math.factorial(t)

    for i in range(0, n, t):

        group = tuple(sorted(seq[i:i + t]))

        dict[group] = dict.get(group, 0) + 1

    lst\_obs = sorted(list(dict.values()), reverse=True)

    exp = np.array([n / param] \* len(lst\_obs))

    return chi\_2(seq, alpha, lst\_obs, exp, param)

def monotony(seq):

    alpha = 0.05

    A = [

        [4529.4, 9044.9, 13568, 22615,  22615,  27892 ],

        [9044.9, 18097,  27139, 36187,  452344, 55789 ],

        [13568,  27139,  40721, 54281,  67582,  83685 ],

        [18091,  36187,  54281, 72414,  90470,  111580],

        [22615,  45234,  67852, 90470,  113262, 139476],

        [27892,  55789,  83685, 111580, 139476, 172860]

    ]

    b = [1 / 6, 5 / 24, 11 / 120, 19 / 720, 29 / 5040, 1 / 840]

    n = len(seq)

    lst = []

    i = 0

    while i < n:

        s = 1

        while i + s < n and seq[i + s - 1] <= seq[i + s]:

            s += 1

        lst.append(s)

        i += s

    counts = {}

    for l in lst:

        counts[l] = counts.get(l, 0) + 1

    res = []

    temp = 0

    for c in lst:

        m = 1 / 6

        min\_val = min(c, 6)

        for i in range(min\_val):

            for j in range(min\_val):

                m += (seq[i + temp] - n \* b[i]) \* (seq[j + temp] - n \* b[j]) \* A[i][j]

        temp += c

        res.append(m)

    return chi\_2(res, alpha)

def conflicts(srq):

    m = 1024

    l = len(srq)

    sr\_ = l / m

    p0 = 1 - l / m + math.factorial(l) / (2 \* math.factorial(l - 2) \* m\*2)

    conf = l / m - 1 + p0

    return "-" if abs(conf - sr\_) > 10 else "+"

if \_\_name\_\_ == "\_\_main\_\_":

    path = input()

    p = read\_from\_file(path)

    trans\_p = transform\_lst\_nums(p)

    mean = np.mean(trans\_p)

    print(f"Мат. ожидание последовательности: {mean}")

    std = sps.tstd(trans\_p)

    print(f"Среднеквадратичное отклонение последовательности: {std}")

    pogr(trans\_p)

    print("Критерий хи-квадрат:")

    print(chi\_2(trans\_p))

    print("Критерий серий:")

    print(series(trans\_p))

    print("Критерий интервалов:")

    print(intervals(trans\_p))

    print("Критерий разбиений:")

    print(partitions(trans\_p))

    print("Критерий перестановок:")

    print(permutations(trans\_p))

    print("Критерий монотонности:")

    print(monotony(trans\_p))

    print("Критерий конфликтов:")

    print(conflicts(trans\_p))

    plot\_distribution(trans\_p)