МИНИСТЕРСТВО НАУКИ  И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО  ОБРАЗОВАНИЯ
«Национальный исследовательский ядерный университет «МИФИ»
Обнинский институт атомной энергетики
Отделение интеллектуальных кибернетических систем

**Лабораторная работа №2**
Вариант №18.

Выполнила студентка
Группы ИС-Б17
Отделения ИКС
Петренко В. Ю.
Проверила:
профессор, д.т.н. Гулина О. М.

Обнинск, 2020

1. Теорема. Случайная величина ξ, определенная на (a,b) и удовлетворяющая уравнению $\int_a^{\xi} f(u)du=\gamma$, или F(ξ)= γ, имеет плотность распределения f(x).

   Согласно условию нормировки:

   $\int_0^1 a u^9 du=\frac{a}{10}\cdot u^{10}\Big|_0^1=\frac{a}{10}\cdot(1-0)=1 \quad \to \quad a=10$

   $f_{\xi}(x)=10x^9$ ; $\int_0^{\xi} f_{\xi}(x)dx=\gamma$ ; $u^{10}\Big|_0^{\xi}=\gamma$ ; $\gamma=\xi^{10}$ ;

   $\xi=\gamma^{(1/10)}$

2. Алгоритм

```
#python 3.6
from math import *
k=2500 #число элементов
r=0#кол-во интервалов
p=0#теоретическая вероятность попадания в каждый 
интервал
array=[] #массив псевдослучайных чисел
newarray=[]#новый массив
VerU=[]
l_aper=0 #длина апериодичности
l_per=0 #длина периода
p_i =[] #количество попаданий в каждый интервал
X2=0 #хи-вквадрат

def fraction(x):
    # функция для расчета дробной части
    return x - int(x)

def fillArray():
    # функция для заполнения массива
    y0=float(input("Введите гамма-нулевое: "))
    accrs=int(input("Введите количество знаков после запятой: 
"))
    for i in range(k):
        array.append(y0)
        y0=(10 ** -accrs)*int((10 ** accrs)*fraction(float(((1-y0) ** 
3)*(10 ** accrs)))) #метод середины квадратов
    print("Массив заполнен псевдослучайными числами.")
    for i in range(k):
```

```
        #по новой формуле заполняем массив
        newarray.append(array[i]**0.1)

def periodLength():
    global l_aper, l_per
    print("Определение длины периода и апериодичности.")
    flag=True #пока в последовательности будут одинаковые
элементы
    for i in range(k):
        for j in range(i+1, k):
            if(abs(newarray[i]-
newarray[j])<0.00000001):#сравниваем
                print("Совпадение в ", i, "-ом и ", j, "-ом элементах: ",
newarray[i], " и ", newarray[j])
                l_aper = j
                l_per = j-i
                flag=False
            if not flag:
                break
        if flag:
            #если нет одинаковых элементов, длина
апериодичности = длине последовательности
            l_aper=k
            l_per=0
        if not flag:
            break

def calc_pi():
    print("Рассчет количества попаданий в каждый интервал.")
    global newarray, r, p
    r = int((1 + 3.3 * log10(k)))
    p = float((1 / r))
    print("Число интервалов", r)
    for i in range(r):
        #обнуляем p_i
        p_i.append(0)
    print("Распределение по интервалам:")
    for i in range(k):
        for j in range(k):
            if (newarray[j]>(i*p) and newarray[j]<((i+1)*p)):
                p_i[i]+=1
```

```
    for i in range(r):
        print(p_i[i], end = ', ')

def calc_X2():
    print("Рассчет X2.")
    X2 = 0
    tver=[]
    for i in range(r):
        tver.append((((i+1)/r)**10)-(i/r)**10)
    for i in range(r):
        X2+=((p_i[i]-(k*tver[i])) ** 2)/(k*tver[i])
    print("X2 = ", X2)

def show():
    n=int(input("Вывести последовательность до: "))
    if n>k:
        n=k
    for i in range(n):
        print(newarray[i], end = ', ')

fillArray()
periodLength()
calc_pi()
calc_X2()
show()
```

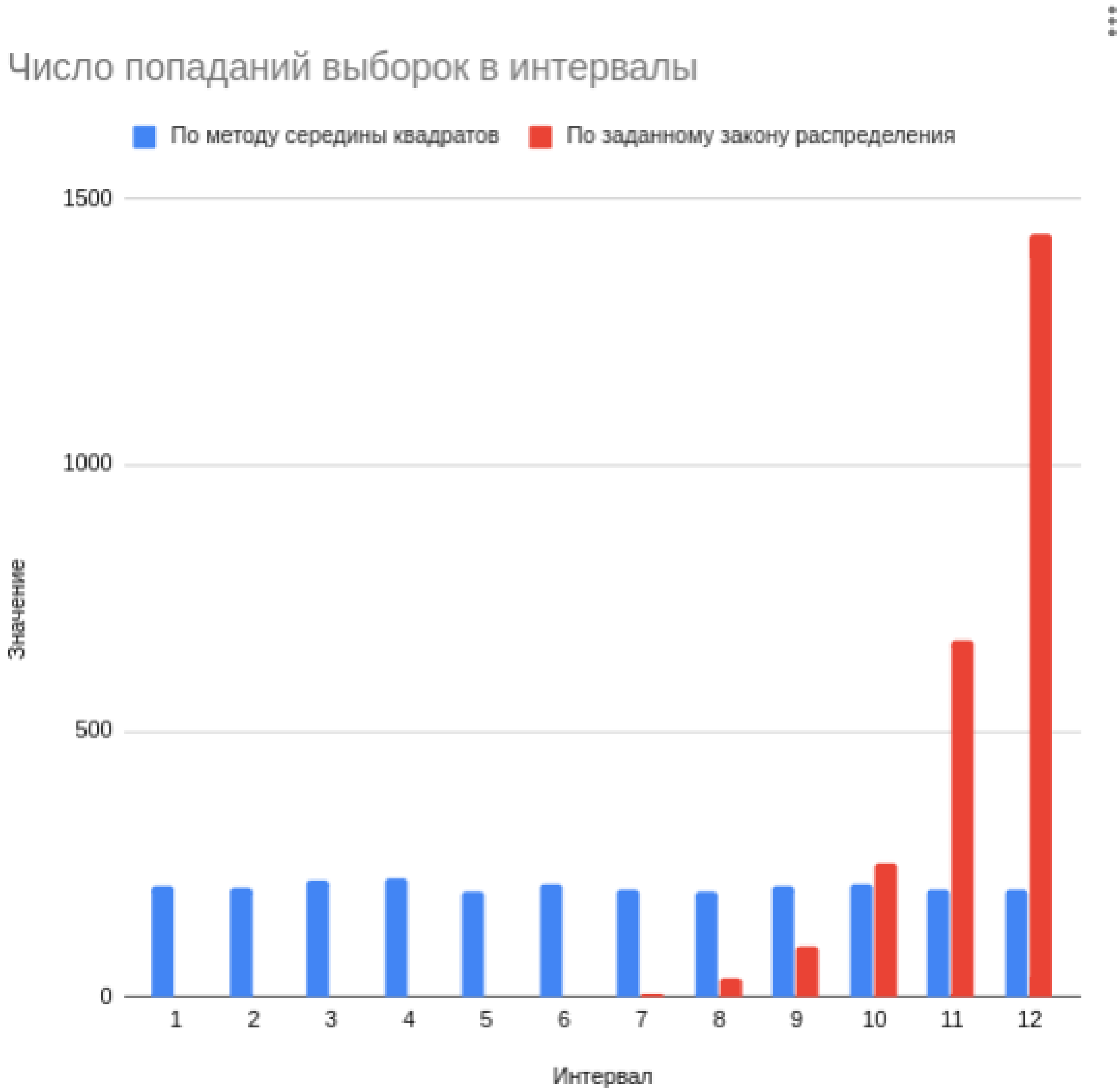
3. Результаты

Число интервалов 12

Распределение по интервалам:

| Интервал | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Число попаданий | 0 | 0 | 0 | 0 | 0 | 12 | 14 | 3 | 28 | 228 | 681 | 1534 |

$X^2 =$ 4.213018302676952




При s=11 и a=0.95 теоретическое значение $\chi^2$=4.58. Экспериментальное значение $\chi^2$ при $\gamma_0$= 0.45864863 получилось 4.213018302676952.

4. Выводы: при выполнении лабораторной работы была сгенерирована последовательность псевдослучайных чисел с заданным законом распределения. Проверка, проведенная с помощью критерия Пирсона, показала соответствие полученной последовательности заданному закону распределения.