МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
**«Рязанский государственный радиотехнический университет
Имени В. Ф. Уткина»**

Факультет вычислительной техники
Кафедра вычислительной и прикладной математики

Отчёт по практической работе №4

по дисциплине:
“Моделирование”

по теме:

“Генерирование случайных величин с нормальным законом распределения”

Выполнил: студент гр. 242

Фокин А.М.

Проверил: Анастасьев А. А.

Рязань 2025

Цель работы:

Составить подпрограмму генерирования случайных величин с нормальным законом распределения методом, основанным на центральной предельной теореме, а также методом, определенным в соответствии с вариантом задания (табл. 4). Параметры закона распределения указаны в виде N(μ, σ^2). По полученной с помощью подпрограммы выборке построить и проанализировать гистограмму частот и статистическую функцию распределения, оценить матожидание и дисперсию случайной величины. Соответствие эмпирических данных теоретическому распределению проверить с помощью критерия Пирсона или критерия Колмогорова. Объем выборки случайных величин не менее 1000. Количество интервалов разбиения k = 15 или k = 25.

	Закон распределения	Способ построения
19	Нормальный, N(3, 0.1)	Метод аппроксимации

Программа реализует следующие операции:

– Генерация нормальных случайных величин по ЦПТ (метод аппроксимации):

```
u = np.random.rand(12)
z = np.sum(u) - 6.0
x = mu + sigma * z
```

– Расчёт выборочного среднего и дисперсии:

```
sample_mean = np.mean(data)
sample_var = np.var(data, ddof=1)
```

– Построение гистограммы и наложение теоретической плотности:

```
plt.bar(bin_centers, rel_freq, ...)
plt.plot(xs, ys, ...)
```

– Проверка соответствия распределения (Колмогоров):

```
D = max(D_plus, D_minus)
```

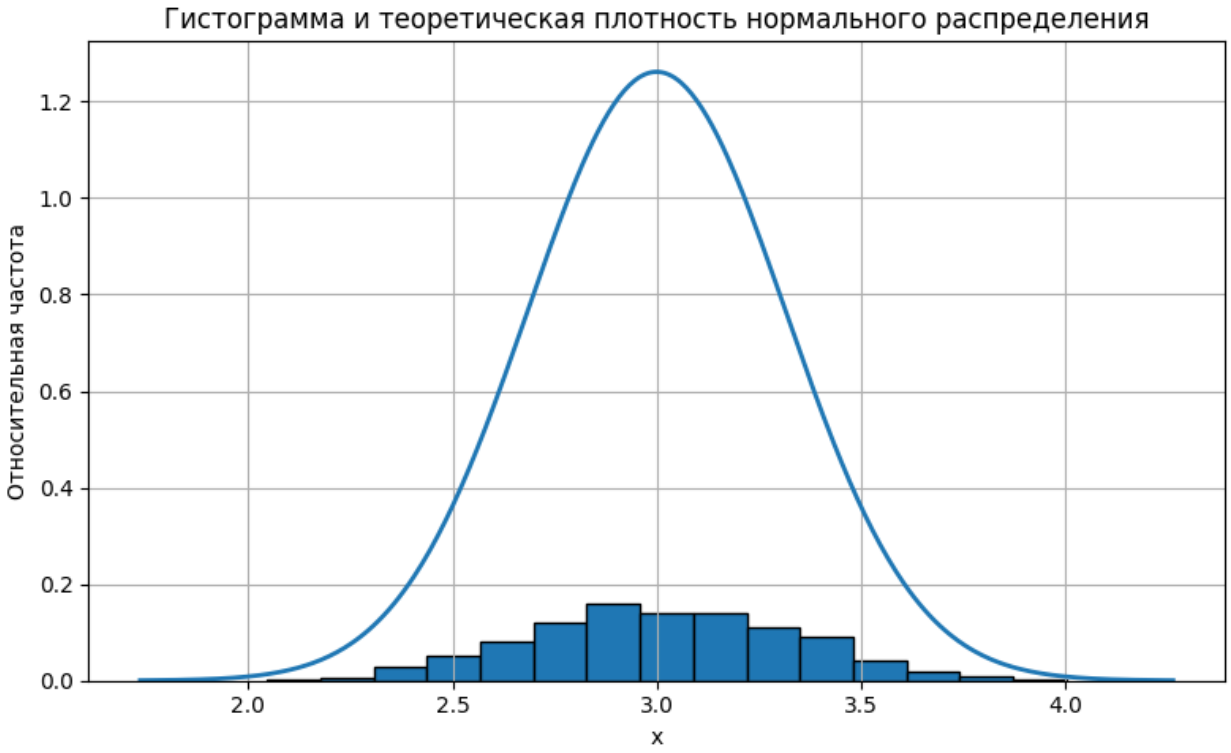
– Проверка критерием χ^2:

```
chi2 += (obs - exp) ** 2 / exp
```

Итог: программа моделирует распределение $N(3,0.1)$методом, основанным на центральной предельной теореме, строит гистограмму и проверяет соответствие теоретическому закону.

полный код программы приведен в приложении 1

Результат работы программы:




```
========== RESTART: D:\мусорка\учеба\моделирования\lab4-Fokin-242.py ==========
Объем выборки: 2000
Выборочное среднее: 3.0106449613272708
Выборочная дисперсия: 0.10449739091818613
Критерий Колмогорова-Смирнова D: 0.02852897067790028
Критическое значение D (α=0.05): 0.03041052449399714
Гипотеза H0 не отвергается при уровне значимости 0.05
Статистика χ²: 23.748740784462218
Количество интервалов: 15
Ожидаемые частоты по интервалам: [  6.652  19.194  46.823  96.563 168.365 248.18
7 309.316 325.932 290.371
 218.716 139.284  74.991  34.134  13.135   4.273]
```

Объем выборки: 2000
Выборочное среднее: 3.0106449613272708
Выборочная дисперсия: 0.10449739091818613
Критерий Колмогорова-Смирнова D: 0.02852897067790028
Критическое значение D (α=0.05): 0.03041052449399714
Гипотеза H0 не отвергается при уровне значимости 0.05
Статистика χ^2: 23.748740784462218
Количество интервалов: 15
Ожидаемые частоты по интервалам: [6.652 19.194 46.823
96.563 168.365 248.187 309.316 325.932 290.371
 218.716 139.284 74.991 34.134 13.135   4.273]

Ответ на вопрос

2. Какие существуют способы формирования последовательности случайных величин, отвечающих нормальному закону распределения?

– Метод центральной предельной теоремы (суммирование нескольких равномерных случайных величин).
– Метод Бокса–Мюллера.
– Метод обратной функции.
– Метод отбора.
– Алгоритм Ziggurat.

Приложение 1 - код программы

```
import numpy as np
import math
import matplotlib.pyplot as plt

mu = 3.0
sigma = math.sqrt(0.1)
n = 2000
k = 15

def normal_central_limit(size):
    samples = []
    for _ in range(size):
        u = np.random.rand(12)
        z = np.sum(u) - 6.0
        x = mu + sigma * z
        samples.append(x)
    return np.array(samples)

def normal_pdf(x):
    return (1 / (sigma * math.sqrt(2 * math.pi))) * math.exp(-
((x - mu) ** 2) / (2 * sigma ** 2))

def normal_cdf(x):
    return 0.5 * (1 + math.erf((x - mu) / (sigma *
math.sqrt(2))))

data = normal_central_limit(n)

sample_mean = float(np.mean(data))
sample_var = float(np.var(data, ddof=1))

counts, bin_edges = np.histogram(data, bins=k)
bin_centers = 0.5 * (bin_edges[:-1] + bin_edges[1:])
counts = counts.astype(float)
rel_freq = counts / n

theo_probs = []
for i in range(len(bin_edges) - 1):
    p = normal_cdf(bin_edges[i + 1]) - normal_cdf(bin_edges[i])
    theo_probs.append(p)
theo_probs = np.array(theo_probs)
expected = theo_probs * n
```

```
plt.figure(figsize=(8, 5))
plt.bar(bin_centers, rel_freq, width=(bin_edges[1] -
bin_edges[0]), align='center', edgecolor='black')
xs = np.linspace(mu - 4 * sigma, mu + 4 * sigma, 400)
ys = [normal_pdf(x) for x in xs]
plt.plot(xs, ys, linewidth=2)
plt.xlabel('x')
plt.ylabel('Относительная частота')
plt.title('Гистограмма и теоретическая плотность нормального
распределения')
plt.grid(True)
plt.tight_layout()
plt.show()

sorted_data = np.sort(data)
emp_x = sorted_data
emp_y = np.arange(1, n + 1) / n
theo_y = np.array([normal_cdf(x) for x in emp_x])
D_plus = np.max(emp_y - theo_y)
D_minus = np.max(theo_y - (np.arange(0, n) / n))
D = max(D_plus, D_minus)
D_crit = 1.36 / math.sqrt(n)

chi2 = 0.0
for obs, exp in zip(counts, expected):
    if exp > 0:
        chi2 += (obs - exp) ** 2 / exp

print('Объем выборки:', n)
print('Выборочное среднее:', sample_mean)
print('Выборочная дисперсия:', sample_var)
print('Критерий Колмогорова-Смирнова D:', D)
print('Критическое значение D (α=0.05):', D_crit)
if D > D_crit:
    print('Гипотеза H0 отвергается при уровне значимости 0.05')
else:
    print('Гипотеза H0 не отвергается при уровне значимости
0.05')
print('Статистика χ²:', chi2)
print('Количество интервалов:', k)
print('Ожидаемые частоты по интервалам:', np.round(expected, 3))
```