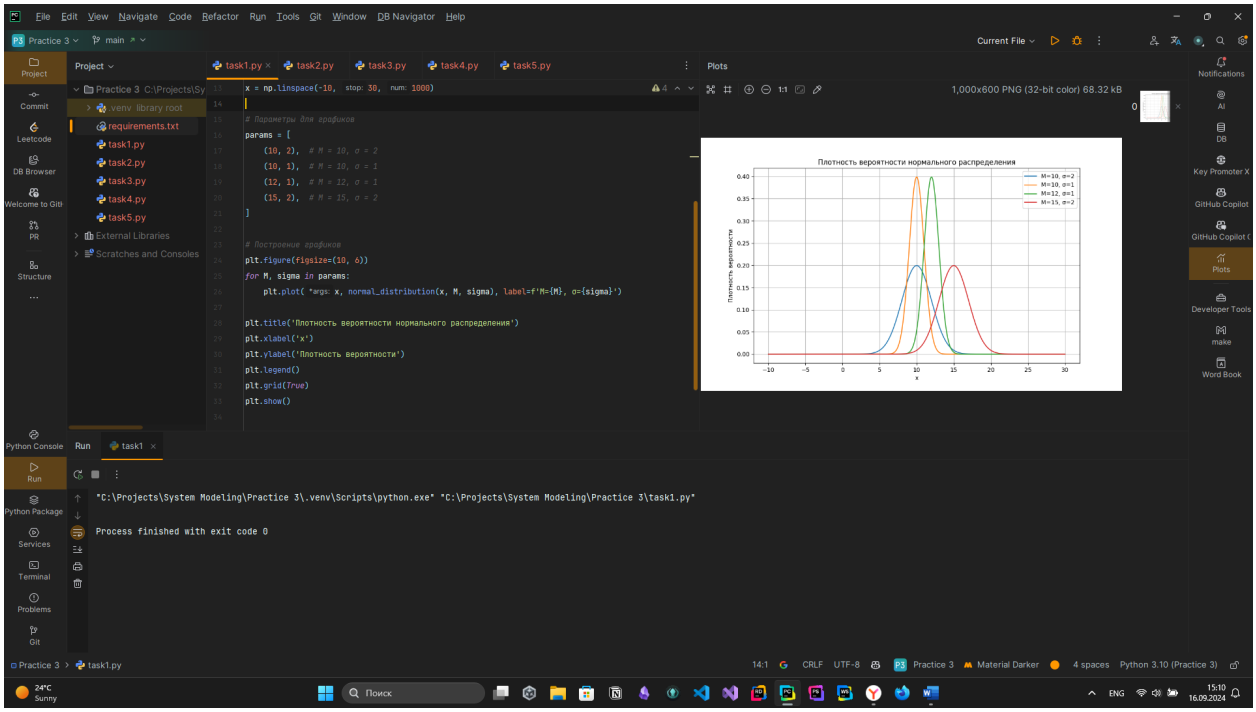
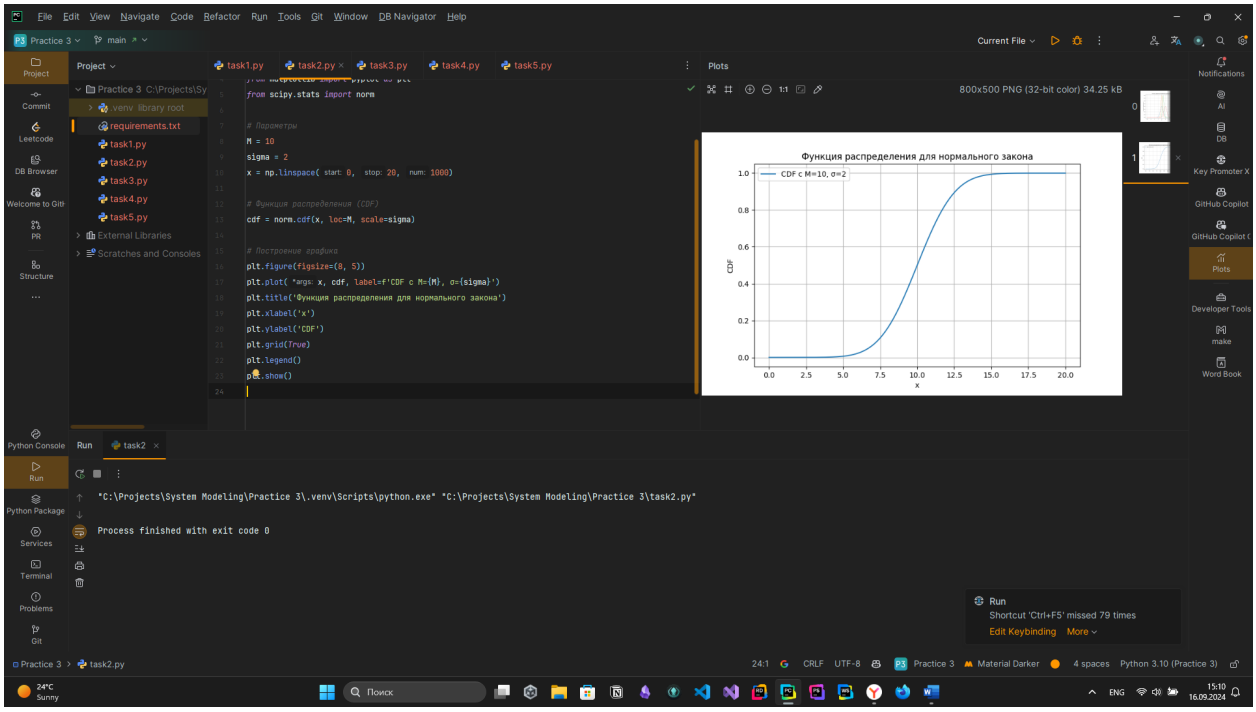
Практика 3

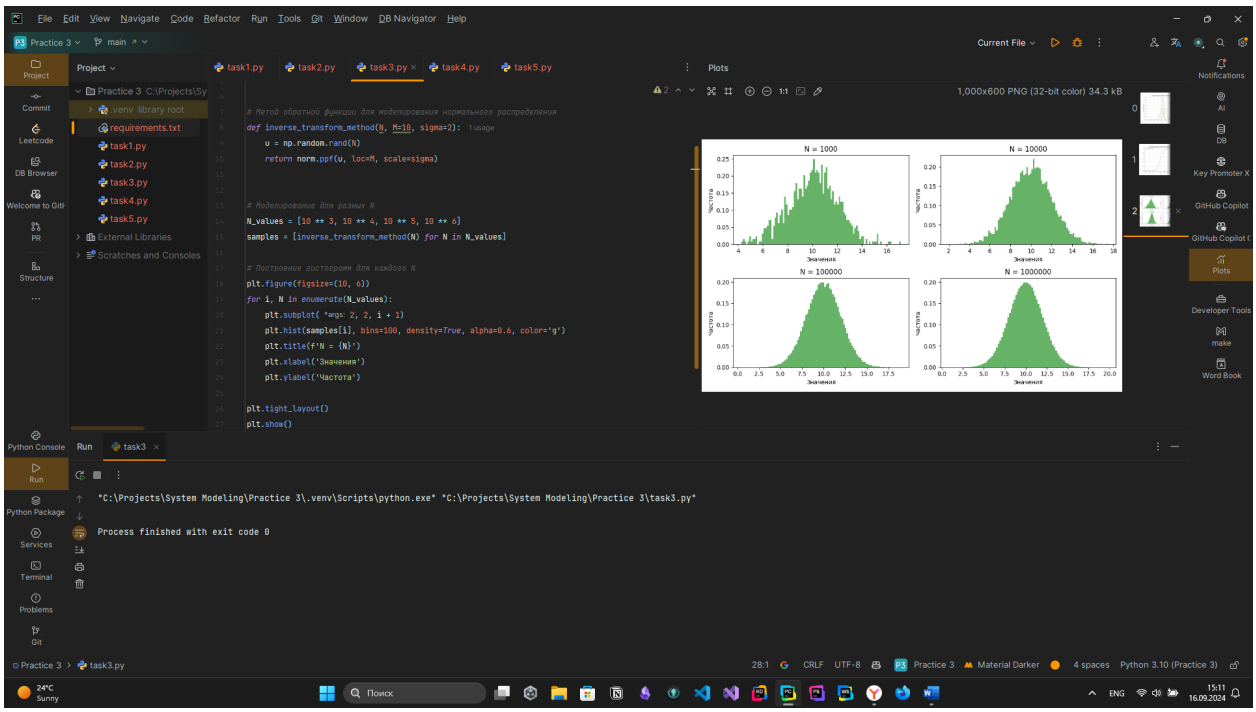
БСБО-09-23

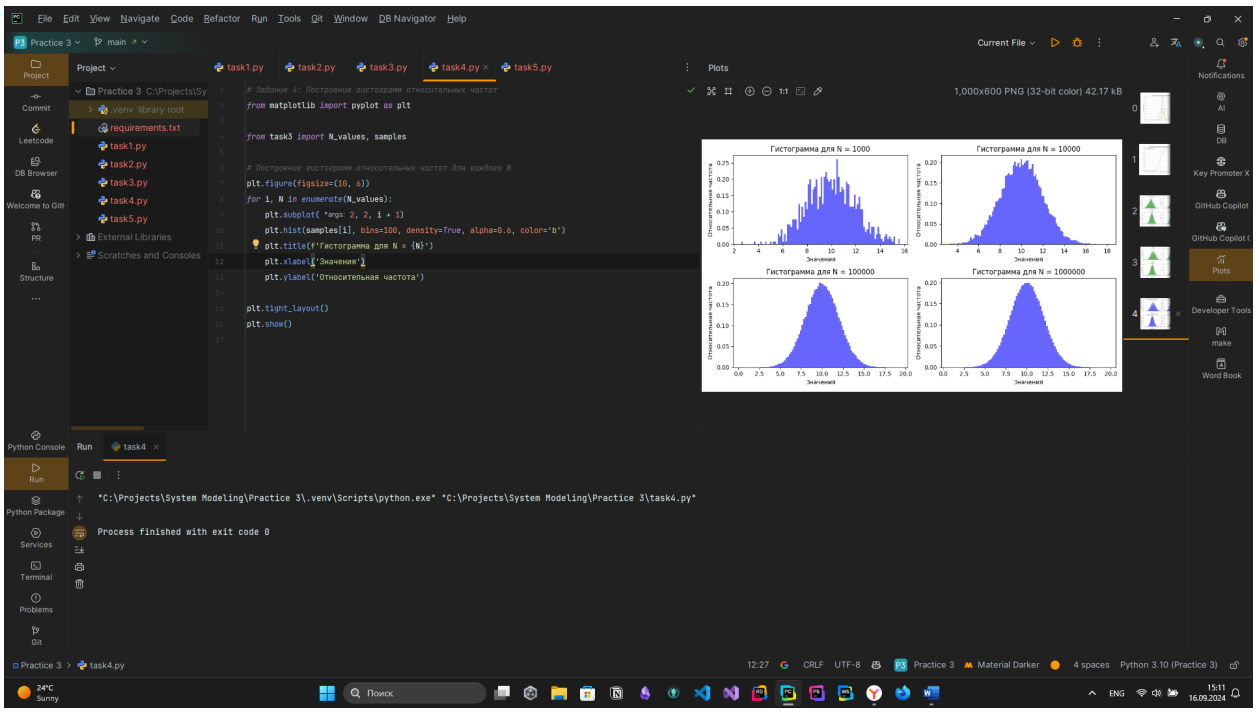
Шутов Кирилл Сергеевич

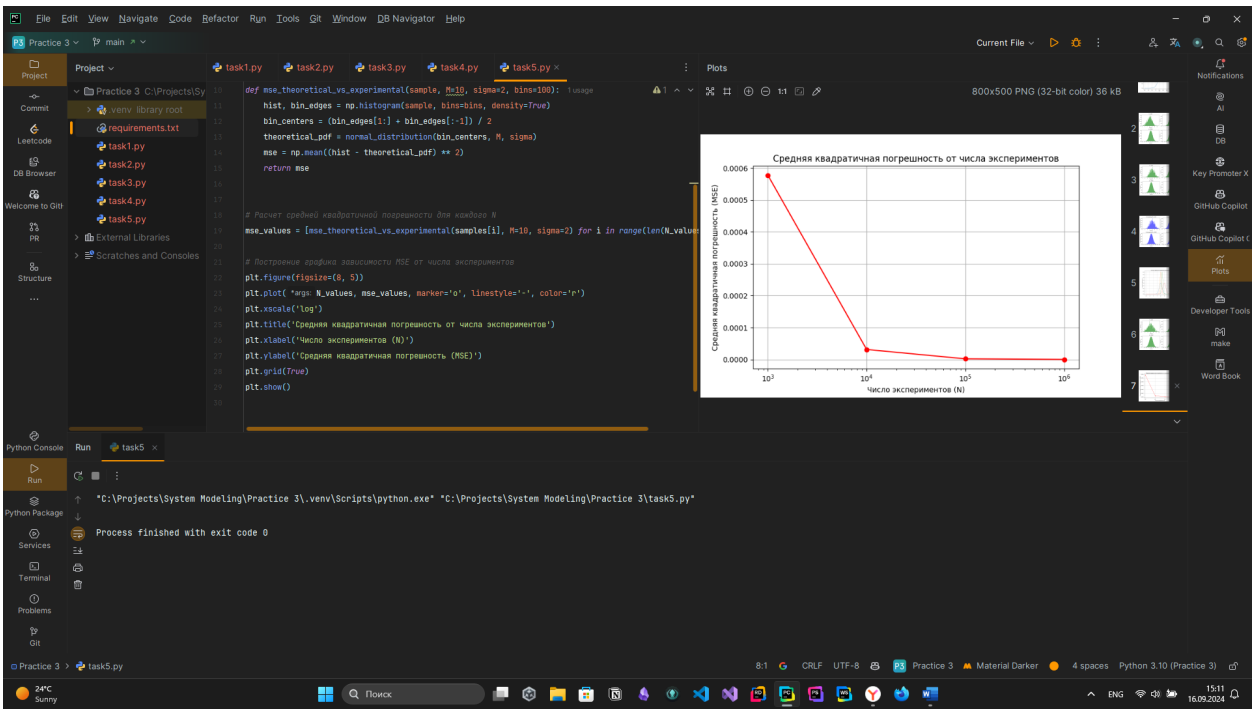
Пример работы






File Edit View Navigate Code Refactor Run Tools Git Window DB Navigator Help
Practice 3
main
Current File
Project
task1.py
task2.py
task3.py
task4.py
task5.py
Practice 3 C:\Projects\Sy
venv library root
requirements.txt
task1.py
task2.py
task3.py
task4.py
task5.py
External Libraries
Scratches and Consoles
Метод обратной функции для моделирования нормального распределения
def inverse_transform_method(N, M=10, sigma=2): 1 usage
u = np.random.rand(N)
return norm.ppf(u, loc=M, scale=sigma)
# Моделирование для разных N
N_values = [10 ** 3, 10 ** 4, 10 ** 5, 10 ** 6]
samples = [inverse_transform_method(N) for N in N_values]
# Построение гистограмм для каждого N
plt.figure(figsize=(10, 6))
for i, N in enumerate(N_values):
plt.subplot(*args: 2, 2, i + 1)
plt.hist(samples[i], bins=100, density=True, alpha=0.6, color='g')
plt.title(f'N = {N}')
plt.xlabel('Значения')
plt.ylabel('Частота')
plt.tight_layout()
plt.show()
Plots
1,000x600 PNG (32-bit color) 34.3 kB
N = 1000
N = 10000
N = 100000
N = 1000000
Частота
Значения
Run
task3
"C:\Projects\System Modeling\Practice 3\.venv\Scripts\python.exe" "C:\Projects\System Modeling\Practice 3\task3.py"
Process finished with exit code 0
Python Console
Python Package
Services
Terminal
Problems
Git
Practice 3 > task3.py
28:1 CRLF UTF-8 Practice 3 Material Darker 4 spaces Python 3.10 (Practice 3)
24°C Sunny
Поиск
ENG
15:11 16.09.2024


File Edit View Navigate Code Refactor Run Tools Git Window DB Navigator Help
Practice 3
main
Current File
Project
task1.py
task2.py
task3.py
task4.py
task5.py
Practice 3 C:\Projects\Sy
venv library root
requirements.txt
task1.py
task2.py
task3.py
task4.py
task5.py
External Libraries
Scratches and Consoles
# Задание 4: Построение гистограмм относительных частот
from matplotlib import pyplot as plt
from task3 import N_values, samples
# Построение гистограмм относительных частот для каждого N
plt.figure(figsize=(10, 6))
for i, N in enumerate(N_values):
plt.subplot(*args: 2, 2, i + 1)
plt.hist(samples[i], bins=100, density=True, alpha=0.6, color='b')
plt.title(f'Гистограмма для N = {N}')
plt.xlabel('Значения')
plt.ylabel('Относительная частота')
plt.tight_layout()
plt.show()
Plots
1,000x600 PNG (32-bit color) 42.17 kB
Гистограмма для N = 1000
Гистограмма для N = 10000
Гистограмма для N = 100000
Гистограмма для N = 1000000
Относительная частота
Значения
Run
task4
"C:\Projects\System Modeling\Practice 3\.venv\Scripts\python.exe" "C:\Projects\System Modeling\Practice 3\task4.py"
Process finished with exit code 0
Python Console
Python Package
Services
Terminal
Problems
Git
Practice 3 > task4.py
12:27 CRLF UTF-8 Practice 3 Material Darker 4 spaces Python 3.10 (Practice 3)
24°C Sunny
Поиск
ENG
15:11 16.09.2024



Листинг кода

```
# Задание 1: Плотность вероятности нормального распределения

import numpy as np
import matplotlib.pyplot as plt


# Функция плотности вероятности нормального распределения
def normal_distribution(x, M, sigma):
    return (1 / (sigma * np.sqrt(2 * np.pi))) * np.exp(-((x - M) ** 2) / (2 * sigma ** 2))


# Параметры
x = np.linspace(-10, 30, 1000)

# Параметры для графиков
params = [
    (10, 2),  # M = 10, σ = 2
    (10, 1),  # M = 10, σ = 1
    (12, 1),  # M = 12, σ = 1
    (15, 2),  # M = 15, σ = 2
]

# Построение графиков
plt.figure(figsize=(10, 6))
for M, sigma in params:
    plt.plot(x, normal_distribution(x, M, sigma), label=f'M={M}, σ={sigma}')

plt.title('Плотность вероятности нормального распределения')
plt.xlabel('x')
plt.ylabel('Плотность вероятности')
plt.legend()
```

```
plt.grid(True)
plt.show()
```

```
# Задание 2: Функция распределения (CDF) для нормального закона

import numpy as np
from matplotlib import pyplot as plt
from scipy.stats import norm

# Параметры
M = 10
sigma = 2
x = np.linspace(0, 20, 1000)

# Функция распределения (CDF)
cdf = norm.cdf(x, loc=M, scale=sigma)

# Построение графика
plt.figure(figsize=(8, 5))
plt.plot(x, cdf, label=f'CDF с M={M}, σ={sigma}')
plt.title('Функция распределения для нормального закона')
plt.xlabel('x')
plt.ylabel('CDF')
plt.grid(True)
plt.legend()
plt.show()
```

```
# Задание 3: Моделирование нормального распределения методом обратной функции
import numpy as np
from matplotlib import pyplot as plt
from scipy.stats import norm


# Метод обратной функции для моделирования нормального распределения
def inverse_transform_method(N, M=10, sigma=2):
    u = np.random.rand(N)
    return norm.ppf(u, loc=M, scale=sigma)


# Моделирование для разных N
N_values = [10 ** 3, 10 ** 4, 10 ** 5, 10 ** 6]
samples = [inverse_transform_method(N) for N in N_values]

# Построение гистограмм для каждого N
plt.figure(figsize=(10, 6))
for i, N in enumerate(N_values):
    plt.subplot(2, 2, i + 1)
    plt.hist(samples[i], bins=100, density=True, alpha=0.6, color='g')
    plt.title(f'N = {N}')
    plt.xlabel('Значения')
    plt.ylabel('Частота')

plt.tight_layout()
plt.show()
```

```
# Задание 4: Построение гистограмм относительных частот
from matplotlib import pyplot as plt

from task3 import N_values, samples

# Построение гистограмм относительных частот для каждого N
plt.figure(figsize=(10, 6))
for i, N in enumerate(N_values):
    plt.subplot(2, 2, i + 1)
    plt.hist(samples[i], bins=100, density=True, alpha=0.6, color='b')
    plt.title(f'Гистограмма для N = {N}')
    plt.xlabel('Значения')
    plt.ylabel('Относительная частота')

plt.tight_layout()
plt.show()
```

```
# Задание 5: Средняя квадратичная погрешность
import numpy as np
from matplotlib import pyplot as plt

from task1 import normal_distribution
from task3 import samples, N_values


# Функция для вычисления средней квадратичной погрешности
def mse_theoretical_vs_experimental(sample, M=10, sigma=2, bins=100):
    hist, bin_edges = np.histogram(sample, bins=bins, density=True)
    bin_centers = (bin_edges[1:] + bin_edges[:-1]) / 2
    theoretical_pdf = normal_distribution(bin_centers, M, sigma)
    mse = np.mean((hist - theoretical_pdf) ** 2)
    return mse


# Расчет средней квадратичной погрешности для каждого N
mse_values = [mse_theoretical_vs_experimental(samples[i], M=10, sigma=2) for i in
range(len(N_values))]

# Построение графика зависимости MSE от числа экспериментов
plt.figure(figsize=(8, 5))
plt.plot(N_values, mse_values, marker='o', linestyle='-', color='r')
plt.xscale('log')
plt.title('Средняя квадратичная погрешность от числа экспериментов')
plt.xlabel('Число экспериментов (N)')
plt.ylabel('Средняя квадратичная погрешность (MSE)')
plt.grid(True)
plt.show()
```