**МИНОБРНАУКИ РОССИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА»**

**(НГТУ)**

Кафедра «Вычислительные системы и технологии»

Дисциплина «Методы и средства обработки сигналов»

**Лабораторная работа №2**

Выполнил:

Студент группы 23–ВМз

Чернышев Антон Дмитриевич

Проверил:

доцент, кандидат технических наук

Авербух Михаил Леонидович

Нижний Новгород  
2025

**Задание**

Для каждого изображения построить график функции y(x) = μ1(x), то есть результат наложения фильтра F1 теории активного восприятия (один уровень Q-пирамиды) на часть изображения: ширина фильтра w задается пользователем и является положительной целой величиной кратной 4, шаг сдвига фильтра d задается пользователем. Высота фильтра берётся равной высоте исходного изображения. Сначала фильтр F1 накладывается на часть исходного изображения вдоль левой границы изображения, высотой, как было сказано, в высоту изображения, и шириной w, таким образом вычисляется y(x0), то есть самое первое значение в векторе y, за x0 берется середина фильтра (т.е. w/2); затем область наложения сдвигается вправо на d, и вычисляется следующее значение вектора y, соответствующее x = w/2 + d; так продолжается пока не будет достигнута правая граница исходного изображения.

**Постановка задачи:**

1. Ввод данных: w (ширина фильтра), d (шаг);
2. Применение фильтра и прохождение цикла от левой границы до правой с шагом d, на каждом шаге вычисляется Y - разница между суммарной яркостью всех пикселей правой половины части изображения и левой половины части изображения, и X – координата центра части изображения по горизонтали.
3. Вывод таблицы: отобразить на экране два столбца: X и Y.
4. Построение графика: визуализировать зависимость y(x) с помощью точек или линии.

**Код программы**

import numpy as np

import matplotlib.pyplot as plt

from PIL import Image

def compute\_profile(image\_array: np.ndarray, w: int, d: int):

"""

Вычисляет векторы x и y(x) для фильтра F1:

y(x) = сумма яркостей правой половины окна - сумма яркостей левой половины.

Окно имеет ширину w и высоту во весь кадр.

"""

height, width = image\_array.shape # height сейчас не используется, но явно показывает смысл

x\_values = []

y\_values = []

half\_width = w // 2

# Бежим от левой границы до правой с шагом d

for left in range(0, width - w + 1, d):

right = left + w

window = image\_array[:, left:right]

left\_half = window[:, :half\_width]

right\_half = window[:, half\_width:]

sum\_left = int(left\_half.sum())

sum\_right = int(right\_half.sum())

y = sum\_right - sum\_left

x = left + w / 2.0

x\_values.append(x)

y\_values.append(y)

return x\_values, y\_values

def print\_table(x, y):

"""

Печатает первые и последние 10 значений x и y(x) без pandas.

"""

print("Таблица значений x и y(x):")

n = len(x)

if n == 0:

print("Нет данных для отображения.")

return

limit = 10

def print\_rows(start, end, label=None):

if label:

print(label)

print(f"{'i':>4} {'x':>12} {'y(x)':>20}")

print("-" \* 38)

for i in range(start, end):

print(f"{i:>4} {x[i]:>12.2f} {y[i]:>20.2f}")

if n <= 2 \* limit:

print\_rows(0, n)

else:

print\_rows(0, limit, "Первые значения:")

print("...")

print\_rows(n - limit, n, "Последние значения:")

def plot\_profile(image: Image.Image, x, y, w: int, d: int, title\_suffix: str = ""):

"""

Отображает исходное изображение и график y(x) в одном окне.

Явно закрывает фигуру после вывода, чтобы не было подвисаний.

"""

fig, (ax\_img, ax\_plot) = plt.subplots(2, 1, figsize=(14, 10))

# Исходное изображение (ожидается уже в "L" режиме)

ax\_img.imshow(image, cmap="gray")

ax\_img.set\_title(f"Исходное изображение {title\_suffix}")

ax\_img.axis("off")

# График y(x)

ax\_plot.plot(x, y, linewidth=2, label="y(x) = μ1(x)", zorder=3)

# Ось X

ax\_plot.axhline(y=0, linewidth=2, linestyle="-", label="Ось X", zorder=2)

# Ось Y, если 0 попадает в диапазон x

if min(x) <= 0 <= max(x):

ax\_plot.axvline(x=0, linewidth=1, linestyle="--", alpha=0.7, label="x = 0")

ax\_plot.set\_xlabel("Координата x (пиксели)")

ax\_plot.set\_ylabel("y(x) = разница яркостей")

ax\_plot.set\_title(f"График функции y(x) = μ1(x) {title\_suffix}\n(w = {w}, d = {d})")

ax\_plot.grid(True, alpha=0.3)

ax\_plot.legend()

# Симметричный диапазон по Y

y\_arr = np.asarray(y)

if len(y\_arr) > 0:

y\_abs\_max = float(max(abs(y\_arr.min()), abs(y\_arr.max())))

if y\_abs\_max > 0:

ax\_plot.set\_ylim(-y\_abs\_max \* 1.1, y\_abs\_max \* 1.1)

plt.tight\_layout()

plt.show()

plt.close(fig) # явное закрытие фигуры уменьшает задержки после закрытия окна

def ask\_width() -> int:

"""

Запрашивает у пользователя ширину фильтра w (положительное число, кратное 4).

"""

while True:

try:

value = int(input("Введите ширину фильтра (положительное число, кратное 4): "))

if value > 0 and value % 4 == 0:

return value

print("Число должно быть положительным и кратным 4.")

except ValueError:

print("Введено некорректное значение, требуется целое число.")

def ask\_step() -> int:

"""

Запрашивает у пользователя шаг d (целое число больше 0).

"""

while True:

try:

value = int(input("Введите шаг (целое число больше 0): "))

if value > 0:

return value

print("Число должно быть больше 0.")

except ValueError:

print("Введено некорректное значение, требуется целое число.")

def ask\_continue() -> bool:

"""

Спрашивает пользователя, хочет ли он продолжить.

Возвращает True для 'y' и False для 'n' и любого другого ввода.

"""

answer = input("Вы хотите ввести новые данные (y/n)? - ").strip().lower()

if answer not in {"y", "n"}:

print("Некорректный ввод. Программа будет завершена.")

return False

return answer == "y"

def main():

# Пути к изображениям

image\_paths = ["03.png", "10.jpg"]

# Загружаем изображения один раз, сразу конвертируем в оттенки серого

images = []

for path in image\_paths:

img = Image.open(path).convert("L")

img\_array = np.array(img)

images.append((path, img, img\_array))

# Основной цикл

continue\_flag = True

while continue\_flag:

w = ask\_width()

d = ask\_step()

for path, img, img\_array in images:

print(f"\nОбработка изображения: {path}")

x, y = compute\_profile(img\_array, w, d)

print\_table(x, y)

plot\_profile(img, x, y, w, d, title\_suffix=f"({path})")

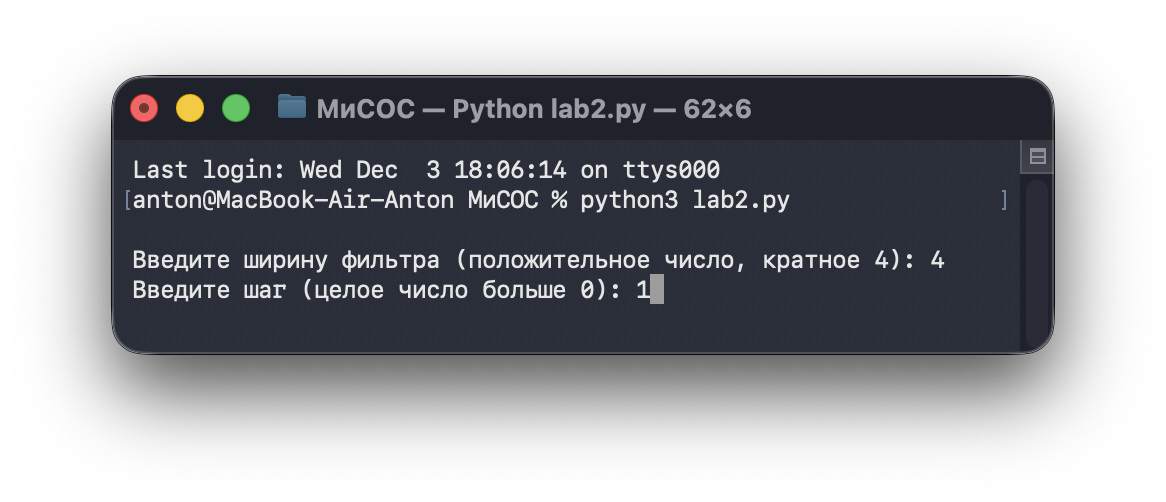
continue\_flag = ask\_continue()

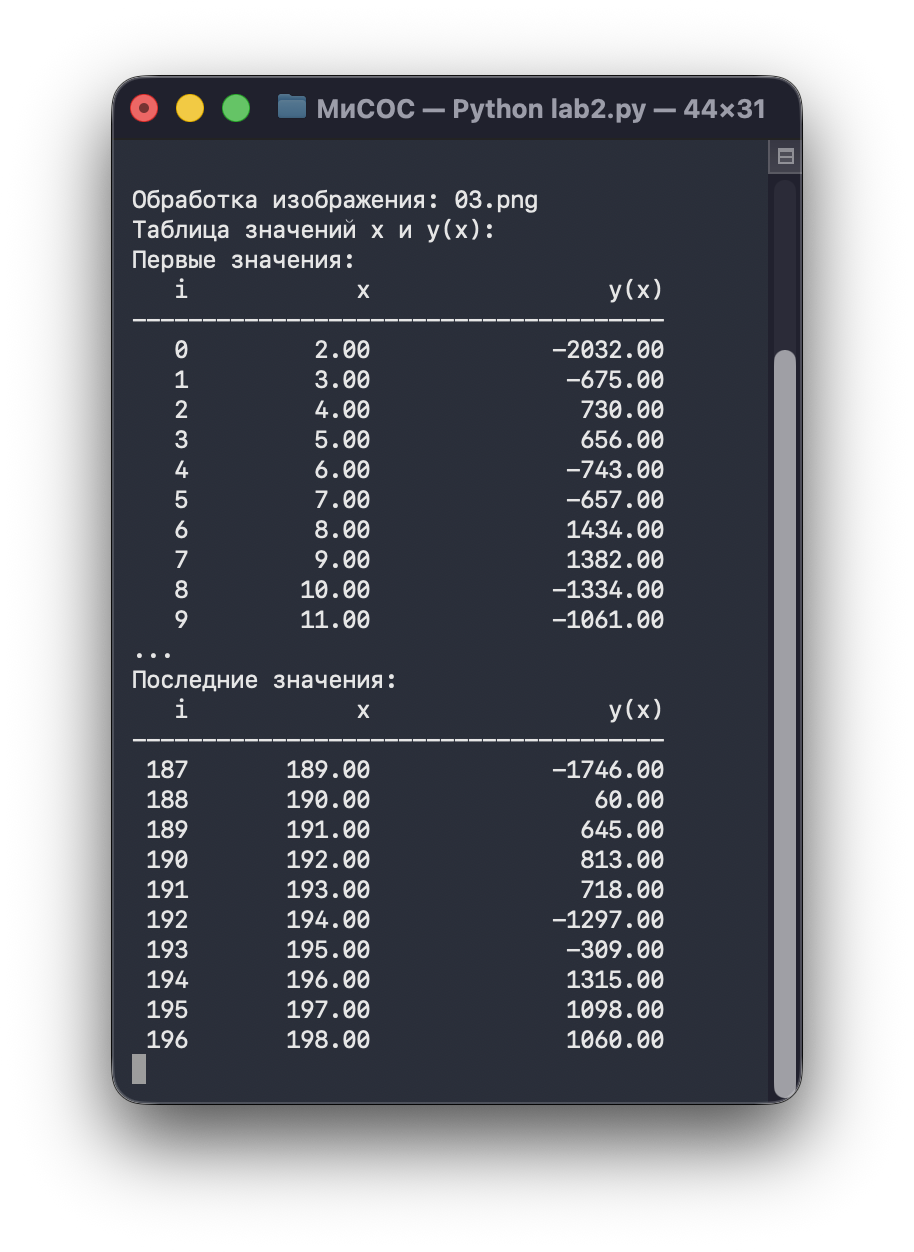
print("\nПрограмма завершена.")

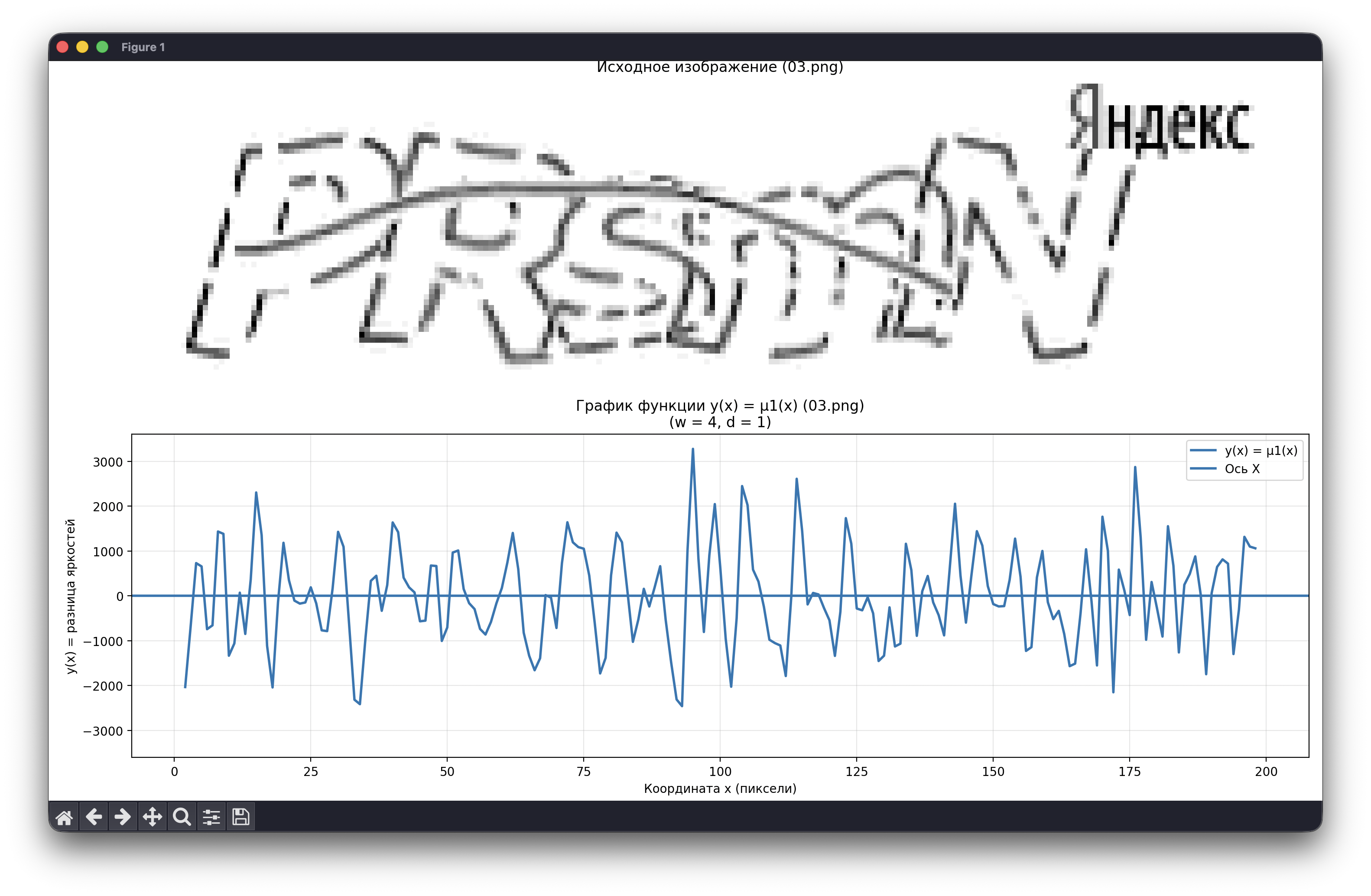
if \_\_name\_\_ == "\_\_main\_\_":

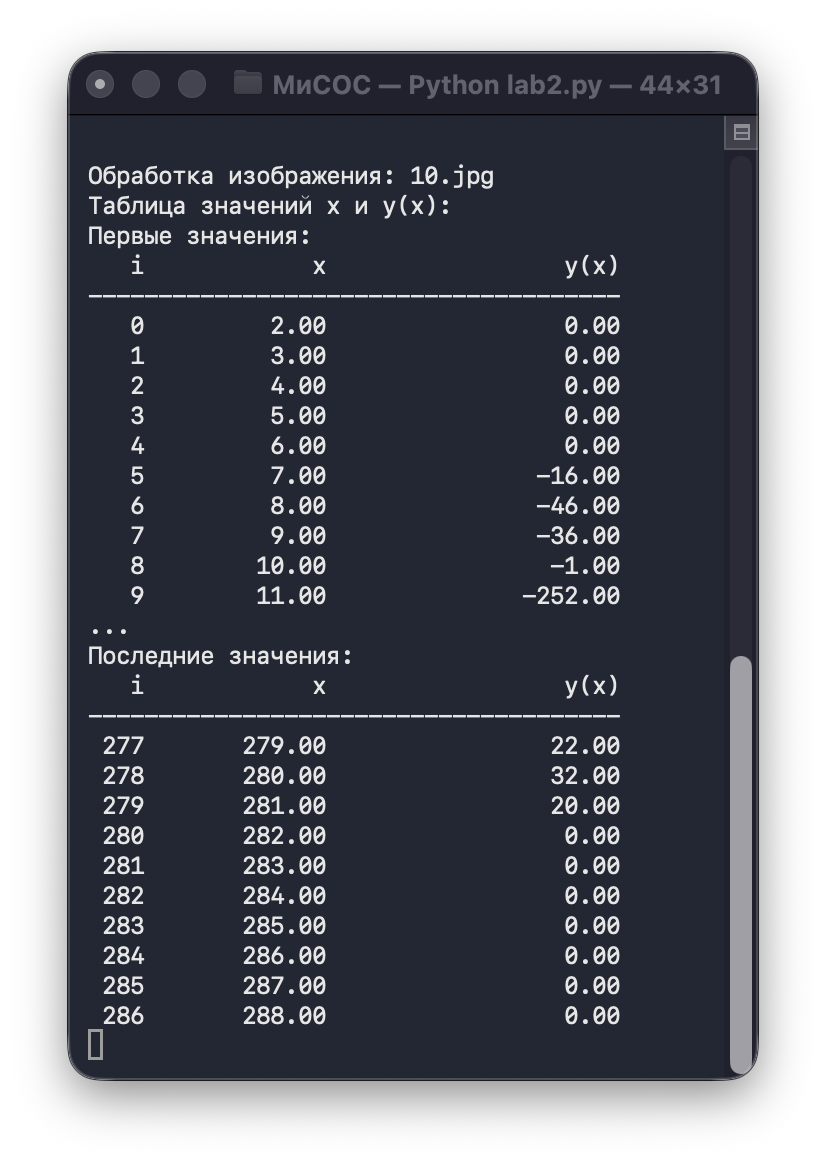
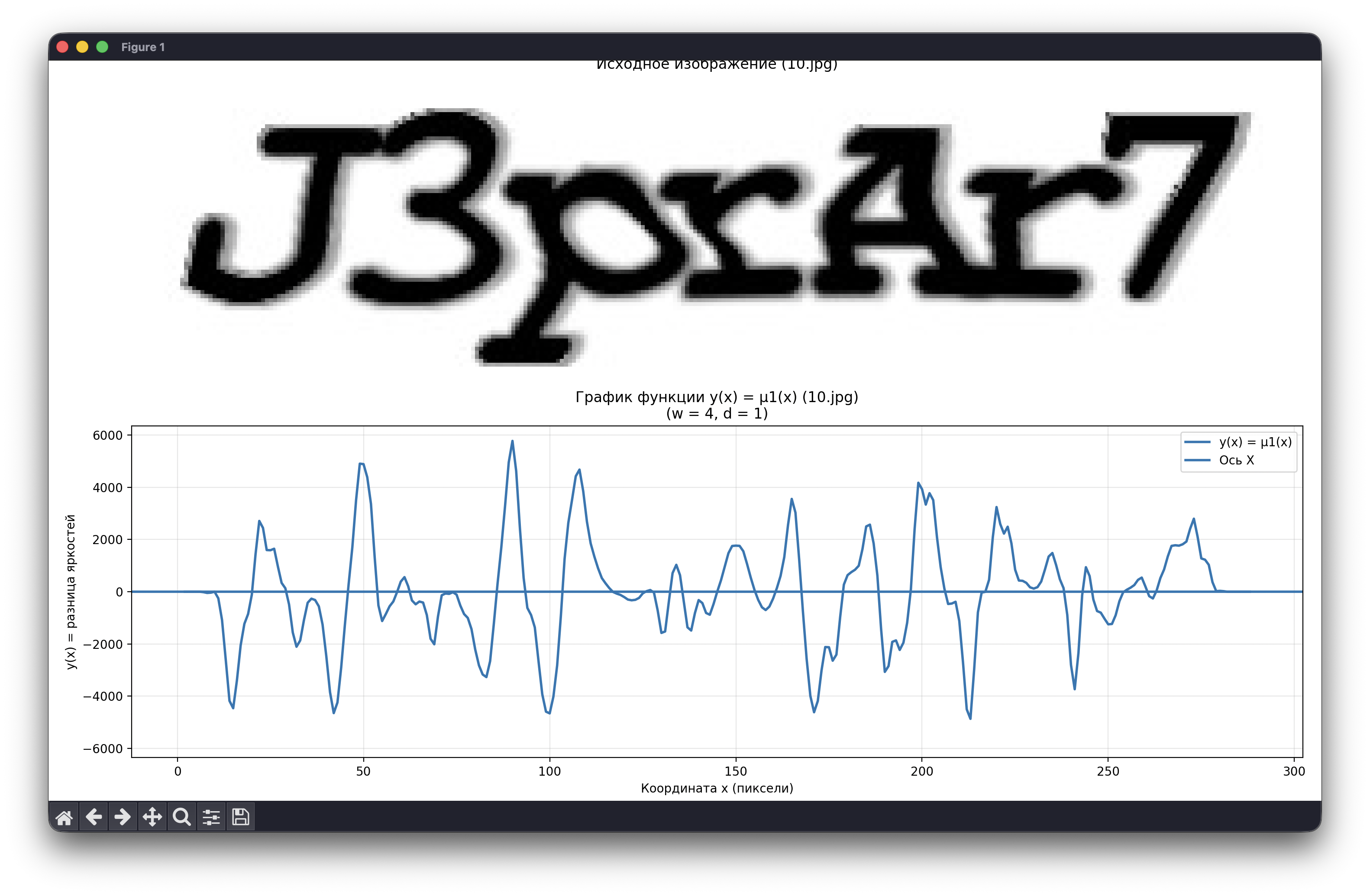
main()

**Демонстрация работы**

**Входные данные:**  
  
**Выходные данные:**





**Вывод**

В ходе выполнения лабораторной работы была успешно решена поставленная задача: разработана программа на языке Python для вычисления и визуализации графика функции , представляющей собой результат наложения линейного фильтра на изображение. Фильтр реализован в соответствии с методикой, описанной в задании: для скользящего окна шириной , охватывающего изображение по вертикали, вычисляется разность суммарной яркости правой и левой половин окна, а полученные значения откладываются как функция от координаты центра окна .

Программа корректно обрабатывает входные изображения, выполняет их преобразование в оттенки серого, обеспечивает ввод параметров фильтра (ширина окна и шаг ) и для каждого набора параметров формирует набор значений и на заданном интервале. Результаты вычислений выводятся в виде таблицы с выборкой первых и последних значений, а также визуализируются в форме графика в требуемом формате. Проведенные вычисления и построенные графики подтверждают правильность реализации алгоритма и соответствие полученных результатов теоретическому описанию фильтра .