Министерство образования и науки России

Рыбинский государственный авиационный технический университет

имени П.А. Соловьева

КАФЕДРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

ДИСЦИПЛИНА:

Объектно-ориентированное программирование

ОТЧЁТ   
ПО ЛАБАРАТОРНОЙ РАБОТЕ № 1

Выполнил Воробина В.М.

(фамилия, имя, отчество)

студент гр. ЗИС-23

Преподаватель Пруктишина В.А.

(фамилия, имя, отчество)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

г. Рыбинск

2024 г.

**Содержание**

[1. Постановка задачи 3](#_Toc184225809)

[2. Спецификация 4](#_Toc184225810)

[3. Теоретические сведения 7](#_Toc184225811)

[4. Организация данных 10](#_Toc184225812)

[5. Текст программы 14](#_Toc184225813)

1. **Постановка задачи**

На языке программирования Python разработать графический редактор Требования:

* должны использоваться визуальные компоненты, с кнопками и формами ввода;
* должны соблюдаться основные принципы ООП, элементы точка и линии должны быть классами, рисование — методы;
* код выкладывать на GitHub.

1. **Спецификация**

**Цель программы:**

Программа предоставляет пользователю возможность рисовать на холсте, размещать точки, соединять их линиями и рисовать произвольные линии с помощью мыши. Она поддерживает два основных режима:

* **Режим «Точки и линии»**: позволяет добавлять точки на холст, соединяя каждую последующую точку с предыдущей линией;
* **Режим «Рисование»**: позволяет свободно рисовать линии с зажатой левой кнопкой мыши.

**Основные функции программы:**

1. **Добавление точки:**
   * Пользователь может щёлкнуть левой кнопкой мыши по холсту, чтобы добавить точку;
   * Точки имеют фиксированную толщину 6 пикселей и рисуются в месте клика черным цветом.
2. **Соединение точек линиями:**
   * Если пользователь добавляет более одной точки, новая точка автоматически соединяется с предыдущей линией;
   * Линии имеют фиксированную толщину 3 пикселя и отображаются синим цветом.
3. **Свободное рисование:**

* В этом режиме пользователь может рисовать произвольные линии, удерживая левую кнопку мыши;
* Линии рисуются красным цветом и толщиной 2 пикселя.

1. **Очистка холста:**
   * Кнопка «Очистить», расположенная в левом верхнем углу программы, позволяет удалить все точки и линии с холста;
   * После очистки пользователь может начать новое рисование.
2. **Интерфейс программы:**
   * Программа включает графический интерфейс реализованный с использованием библиотеки «tkinter»;
   * Холст размером 800x600 пикселей занимает центральную часть окна;
   * Элементы управления режимами и очисткой холста располагаются в верхней части окна.
3. **Теоретические сведения**

**Программа включает два основных режима:**

* Режим «Точки и линии» **-** в этом режиме пользователи могут добавлять точки, которые автоматически соединяются линиями. Каждая новая точка соединяется с предыдущей, образуя линию.
* Режим «Рисование» **-** в этом режиме пользователи могут свободно рисовать линии на холсте, удерживая левую кнопку мыши.

**Основные компоненты программы**:

**Холст -** основной элемент графического интерфейса, на котором выполняется рисование.

**Кнопки управления**:

**Очистить**: удаляет все нарисованные объекты на холсте.

**Точки и линии**: переключает программу в режим рисования точек и соединения их линиями.

**Рисование**: переключает программу в режим свободного рисования линий.

**Нажатие на левую кнопку мыши** (Button-1): определяет точку, которую нужно добавить или с которой нужно начать рисовать.

**Перемещение мыши с зажатой левой кнопкой** (B1-Motion): используется в режиме свободного рисования для рисования произвольных линий.

**Принципы работы с каждым элементом программы:**

**Класс Point:** Точка в графическом редакторе представлена как объект класса (Point). Этот класс использует координаты точки и предоставляет метод для её рисования.

Конструктор (\_\_init\_\_): принимает координаты x и y, которые задают положение точки на холсте.

Метод (draw): рисует точку на холсте с размером 6 пикселей и черным цветом. При каждом добавлении точки на холст, она отображается с заданным размером. Если точка является первой, она просто отображается. Если точка не первая, она автоматически соединяется с предыдущей точкной линией.

**Класс Line:** Линия — это объект, который соединяет две точки. Класс (Line) содержит логику рисования линий между точками.

Конструктор (\_\_init\_\_): принимает два объекта класса (Point) - начальную и конечную точку.

Метод (draw): рисует линию на холсте между этими точками с толщиной 3 пикселя и синим цветом. Линии автоматически рисуются между точками при использовании режима «Точки и линии».

**Режим «Рисование» -** этот режим позволяет пользователю рисовать произвольные линии, удерживая левую кнопку мыши.

**Свободное рисование**: при перемещении мыши с зажатой левой кнопкой программа рисует линию между текущей и предыдущей позицией мыши. Линии рисуются красным цветом с толщиной 2 пикселя. В этом режиме нет привязки к точкам — пользователь может рисовать любые линии, создавая произвольные изображения.

**Кнопка «Очистить» -** при нажатии на эту кнопку программа очищает холст, удаляя все нарисованные точки, линии и другие объекты. Это позволяет начать рисование заново.

Очистка холста: метод (clear\_canvas) удаляет все элементы, нарисованные на холсте, включая точки и линии. Он сбрасывает состояние программы, очищая список точек и последнюю позицию мыши для свободного рисования.

**Переключение между режимами:** программа позволяет пользователю переключаться между двумя режимами:

* Режим «Точки и линии»: активируется при нажатии кнопки «Точки и линии». В этом режиме пользователь добавляет точки и соединяет их линиями.
* Режим «Рисование»: активируется при нажатии кнопки «Рисование». В этом режиме пользователь может рисовать произвольные линии.

Режимы переключаются с помощью соответствующих кнопок интерфейса, и пользователь может в любой момент сменить режим рисования.

1. **Организация данных**

В программе для графического редактора данные организованы с использованием объектно-ориентированных принципов программирования. Каждый графический элемент (точка, линия) представлен отдельным классом, который инкапсулирует данные и методы для работы с этими данными. Программа использует холст (Canvas) для рисования, а данные о точках и линиях хранятся в списках.

**Классы и их объекты:**

Основными классами, которые используются в программе, являются:

* **Point** (Точка)
* **Line** (Линия)

Каждый объект этих классов содержит данные, необходимые для рисования точек и линий на холсте.

**Класс Point:**

Этот класс описывает координаты (x, y) точки на хосте.

**Атрибуты:**

* + x, y: координаты точки на холсте.

**Методы:**

* (draw): рисует точку на холсте с заданным размером и цветом.

Пример хранения данных для точки:

|  |
| --- |
| class Point:  def \_\_init\_\_(self, x, y):  self.x = x  self.y = y  def draw(self, canvas):  canvas.create\_oval(self.x - 3, self.y - 3, self.x + 3, self.y + 3, fill="black", outline="black") |

**Класс Line:** Этот класс описывает линию, соединяющую две точки.

Атрибуты:

* point1, point2: две точки, которые должны быть соединены линией.
* thickness: толщина линии - 3 пикселя.
* color: цвет линии - синий.

Методы:

(draw): рисует линию между двумя точками на холсте.

Пример хранения данных для линии:

|  |
| --- |
| class Line:  def \_\_init\_\_(self, point1, point2):  self.point1 = point1  self.point2 = point2  self.thickness = 3  self.color = "blue"  def draw(self, canvas):  canvas.create\_line(self.point1.x, self.point1.y, self.point2.x, self.point2.y, width=self.thickness, fill=self.color) |

**Списки данных:**

Для управления состоянием программы и хранения информации о рисуемых элементах используются списки.

* **points**: список, в котором хранятся все объекты класса (Point). Каждая точка хранится как объект с координатами (x, y).
* **lines**: список, в котором хранятся все объекты класса (Line). Каждая линия соединяет две точки и хранит информацию о своей толщине и цвете.

Пример организации данных:

|  |
| --- |
| points = []  lines = [] |

**Организация данных для режимов рисования:**

Программа поддерживает два режима рисования:

* Режим «Точки и линии»: в этом режиме создаются объекты точек и линии. Каждая новая точка сохраняется в списке (points), а линия, соединяющая её с предыдущей точкой, сохраняется в списке (lines).
* Режим «Рисование»: в этом режиме программа позволяет рисовать произвольные линии. Эти линии рисуются, пока пользователь удерживает левую кнопку мыши, но не сохраняются в списках, так как они не соединяют точки. Вместо этого просто рисуется линия на холсте, без сохранения её данных в программе.

**Хранение данных о текущем состоянии программы:**

Для удобства работы с текущим состоянием холста, программа использует дополнительные переменные:

* (last\_point): переменная, которая хранит ссылку на последнюю добавленную точку. Эта точка необходима для рисования линии между точками;
* (drawing\_mode): переменная, указывающая, какой режим рисования активен (линии и точки или свободное рисование).

Пример хранения состояния:

|  |
| --- |
| last\_point = None # Последняя добавленная точка  drawing\_mode = "points\_lines" # Текущий режим рисования |

**Алгоритм работы программы с данными:**

* При добавлении точки на холст программа создаёт объект (Point) и добавляет его в список (points). Если это не первая точка, то создаётся объект (Line), соединяющий новую точку с предыдущей, и добавляется в список (lines).
* В режиме рисования программа отслеживает перемещение мыши и рисует линии между текущей позицией мыши и предыдущей, но эти линии не сохраняются в списке, а рисуются только в момент перемещения мыши.
* Кнопка «Очистить» очищает холст, удаляя все элементы из списка (points) и (lines), а также сбрасывает переменные состояния, например, (last\_point).

**Взаимодействие с пользователем:**

Каждый пользовательский ввод (например, клик по холсту или перемещение мыши с зажатой кнопкой) обрабатывается с помощью событий «tkinter», которые вызывают соответствующие методы для создания точек, линий или рисования. Все эти действия влияют на данные, хранящиеся в списках (points) и (lines), а также на состояние программы.

1. **Текст программы**

import tkinter as tk

class Point:

"""Класс Точки"""

def \_\_init\_\_(self, x, y):

self.x = x

self.y = y

def draw(self, canvas, size=6, color="black"):

"""Создание точки"""

canvas.create\_oval(

self.x - size // 2, self.y - size // 2,

self.x + size // 2, self.y + size // 2,

fill=color, outline=color

)

class Line:

"""Класс Линии"""

def \_\_init\_\_(self, start\_point, end\_point):

self.start\_point = start\_point

self.end\_point = end\_point

def draw(self, canvas, width=3, color="blue"):

"""Создание линии"""

canvas.create\_line(

self.start\_point.x, self.start\_point.y,

self.end\_point.x, self.end\_point.y,

width=width, fill=color

)

class GraphicEditorApp:

"""Класс всего приложения"""

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Графический редактор")

# Режим Рисование

self.drawing\_mode = "points\_and\_lines" # "points\_and\_lines" или "free\_draw"

# Кнопки для управления

self.clear\_button = tk.Button(self.root, text="Очистить", command=self.clear\_canvas)

self.clear\_button.pack(side=tk.TOP, padx=5, pady=5)

self.points\_lines\_button = tk.Button(self.root, text="Точки и линии", command=self.set\_points\_and\_lines\_mode)

self.points\_lines\_button.pack(side=tk.TOP, padx=5, pady=5)

self.free\_draw\_button = tk.Button(self.root, text="Рисование", command=self.set\_free\_draw\_mode)

self.free\_draw\_button.pack(side=tk.TOP, padx=5, pady=5)

# Создание Канваса (холста)

self.canvas = tk.Canvas(self.root, bg="white", width=800, height=600)

self.canvas.pack(fill=tk.BOTH, expand=True)

# Перечень точек

self.points = []

# События на канвасе

self.canvas.bind("<Button-1>", self.on\_mouse\_click)

self.canvas.bind("<B1-Motion>", self.on\_mouse\_drag)

# Последняя позиция мыши

self.last\_x, self.last\_y = None, None

def set\_points\_and\_lines\_mode(self):

"""Режим 'Точки и линии'"""

self.drawing\_mode = "points\_and\_lines"

def set\_free\_draw\_mode(self):

"""Режим 'Рисование'"""

self.drawing\_mode = "free\_draw"

def on\_mouse\_click(self, event):

"""ЛКМ - нажатие"""

if self.drawing\_mode == "points\_and\_lines":

self.add\_point(event)

def on\_mouse\_drag(self, event):

"""Движение Мыши с зажатой ЛКМ"""

if self.drawing\_mode == "free\_draw":

self.free\_draw(event)

def add\_point(self, event):

"""Точка и соединение с линией"""

new\_point = Point(event.x, event.y)

new\_point.draw(self.canvas)

if self.points:

# Соединение линией с последней точкой

previous\_point = self.points[-1]

line = Line(previous\_point, new\_point)

line.draw(self.canvas)

self.points.append(new\_point)

def free\_draw(self, event):

"""Свободное рисование"""

if self.last\_x is not None and self.last\_y is not None:

self.canvas.create\_line(

self.last\_x, self.last\_y, event.x, event.y,

width=2, fill="red"

)

self.last\_x, self.last\_y = event.x, event.y

def clear\_canvas(self):

"""Очищает холст полностью"""

self.canvas.delete("all")

self.points = []

self.last\_x, self.last\_y = None, None

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

root = tk.Tk()

app = GraphicEditorApp(root)

root.mainloop()