Министерство образования и науки России

Рыбинский государственный авиационный технический университет

имени П.А. Соловьева

КАФЕДРА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

ДИСЦИПЛИНА:

Объектно-ориентированное программирование

ОТЧЁТ   
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

Выполнил Киселев И.Э.

(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](#_Toc185030340)

[2. Спецификация 4](#_Toc185030341)

[3. Теоретические сведения 6](#_Toc185030342)

[4. Организация данных 8](#_Toc185030343)

[5. Текст программы 11](#_Toc185030344)

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 — методами;
* код выкладывать на GitHub.

1. **Спецификация**

Данная программа представляет собой графический редактор, созданный с использованием библиотеки «tkinter» для языка программирования Python. Основное назначение программы — предоставление пользователю возможности рисовать графические элементы (точки, линии, круги, квадраты) а также произвольно рисовать с помощью мыши.

**Функциональные возможности:**

1. **Рисование точек:**
   * Пользователь может выбрать инструмент «Нарисовать точку» и размещать точки произвольно на холсте;
   * Точки создаются в месте нажатия мыши на холсте;
2. **Рисование линий:**
   * Линии создаются автоматически по уже нарисованным точкам;
   * По нажатию кнопки «Соединить точки» все точки автоматически соединяются в том порядке, в каком были поставлены, образуя ломаную линию.
3. **Рисование кругов:**
   * Пользователь может создавать круги произвольного размера по клику в любом месте холста;
   * Радиус круга указывается в отдельной форме ввода.
4. **Рисование квадратов:**
   * Пользователь может создавать квадраты произвольного размера в любом месте холста по клику;
   * Размер стороны квадрата указывается в отдельной форме ввода.
5. **Функция очистки холста:**

* В случае, если пользователю требуется очистить холст, это делается по нажатию кнопки «Очистить холст»

**Основные интерфейсные элементы:**

* **Холст** –рабочая область, где пользователь может рисовать;
* **Панель инструментов:**
  1. Кнопка **«Нарисовать точку»** для размещения точек.
  2. Кнопка **«Нарисовать линию»** для построения прямых.
  3. Кнопка **«Рисовать круг»** для рисования кругов.
  4. Кнопка **«Рисовать квадрат»** для создания квадратов.
  5. Кнопка **«Очистить холст»** для удаления всех действий на холсте.

**Технические особенности:**

* Каждый графический элемент (точка, линия, круг, квадрат, треугольник) реализован в виде отдельного класса (Point, Line, Circle, Square), что обеспечивает модульность и упрощает расширение программы;
* Для рисования используется объектный подход: каждый инструмент рисования (например, точка или линия) создается и обрабатывается методами соответствующего класса;
* История действий (список всех созданных элементов) сохраняется в памяти, что позволяет пользователю шаг за шагом отменять выполненные действия.

**Ограничения:**

* Программа предназначена для базовых операций и не поддерживает сохранение или экспорт нарисованного в виде файла;
* Функционал рассчитан на интуитивное использование и не включает сложные инструменты редактирования.

1. **Теоретические сведения**
   1. **Принцип управления программой**

**Запуск программы:**

1. **Исполняемый файл**

Чтобы запустить программу, нужно открыть файл в IDLE и в нем выбрать запустить модуль (F5)

**Окно программы**

После запуска программы откроется главное окно, содержащее:

* + Рабочую область (холст), на которой будут рисоваться элементы;
  + Панель инструментов, позволяющую выбирать тип рисуемых элементов и очистить холст.

**Работа с графическим интерфейсом**

1. **Инструмент «****Нарисовать точку»**
   * Выберите инструмент «Нарисовать точку», нажав на соответствующую кнопку на панели инструментов;
   * Кликните левой кнопкой мыши в любом месте холста, чтобы создать точку заданного цвета и толщины;
   * Чтобы нарисовать следующую точку, просто повторите клик в другом месте холста.
2. **Инструмент «****Нарисовать линию»**
   * Выберите инструмент «Нарисовать линию»;
   * При выборе инструмента все нарисованные точки на холсте автоматически соединятся линиями в том порядке, в каком они были поставлены, плюс соединяется первая и последняя точки
3. **Инструмент «Построить круг»**
   * Выберите инструмент «Построить круг»;
   * Укажите радиус круга;
   * После этого в центре холста отрисуется круг указанного размера.
4. **Инструмент «Построить квадрат»**
   * Выберите инструмент «Построить квадрат» ;
   * Укажите длину стороны квадрата.
   * После этого в центре холста отрисуется квадрат указанного размера.
5. **Инструмент «Свободное рисование»**
   * Выберите кнопку «Свободное рисование» ;
   * Зажмите левую кнопку мыши на холсте и начните двигать курсор. Произвольная линия будет рисоваться вдоль движения вашего курсора;
6. **Функция «Очистить окно»**
   * Если рабочая область слишком переполнена, нажмите кнопку «Очистить окно»;
   * Окно будет полностью очищено;
   * Процесс необратим.

#### Завершение работы

После завершения работы просто закройте окно программы. Программа не сохраняет результаты работы.

1. **Организация данных**

Программа построена на принципах объектно-ориентированного программирования (ООП), что делает её структуру гибкой и легко расширяемой. Код состоит из классов и методов, которые определяют логику работы графического редактора, его интерфейс и функционал.

**Структура программы**

Программа разделена на два основных компонента:

1. **Классы рисуемых объектов**

Представляют графические элементы (точка, линия, круг, квадрат, треугольник). Каждый объект содержит свойства (цвет, толщину, координаты) и логику для создания заданной фигуры.

1. **Класс графического редактора**

* Управляет взаимодействием пользователя с интерфейсом;
* Реализует обработку событий (нажатие кнопок, рисование на холсте);

**Описание классов и методов**

1. **Абстрактный класс «Shape»**

Это базовый класс для всех рисуемых объектов.

**Атрибуты:**

* + - canvas – ссылка на холст, где происходит отрисовка;

**Методы:**

* draw()– абстрактный метод, который должен быть реализован в каждом подклассе для отрисовки конкретного элемента.

1. **Подклассы «Shape»**

**Point** – класс для рисования точки:

* + - Хранит координаты x и y;
    - Реализует метод «draw()», который создаёт точку на холсте с помощью метода «canvas.create\_oval».

**Line** – класс для рисования линии:

* Сохраняет координаты стартовой и конечной точки (x1, y1, x2, y2) ;
* Метод «draw()»» создаёт линию на холсте с помощью метода «canvas.create\_line».

**Circle**, **Square**– классы для рисования круга и квадрата соответственно.

* + - Оба класса используют координаты начальной и конечной точек для определения размеров фигуры;
    - Каждый класс реализует метод draw() для соответствующего элемента (create\_oval, create\_rectangle).

1. **Класс «GraphicEditor»**

Это основной класс программы, который управляет взаимодействием между пользователем, графическим интерфейсом и рисуемыми объектами.

**Атрибуты:**

* + - canvas – рабочая область для рисования;
    - drawing\_mode – текущий режим рисования;
    - shapes[] – массив фигур на экране

**Методы:**

* + - \_init\_ () – создаёт панель инструментов;
    - on\_canvas\_click –вызывает рисование по щелчку мыши
    - clear\_canvas() – очистка экрана

**Режимы рисования:**

Например: draw\_circle\_mode(), draw\_square\_mode(), add\_point\_mode(), connect\_points(), – устанавливают текущий режим рисования.

**Обработка событий:**

* on\_canvas\_click(event) – вызывается при нажатии кнопки мыши. Реализует начальную логику для рисования объекта или начала произвольной линии;

**Логика работы программы**

1. **Запуск и интерфейс:**

При запуске программы создаётся главное окно с холстом и панелью инструментов. Панель инструментов позволяет задавать цвет, толщину и выбирать тип рисуемого элемента.

1. **Выбор инструмента:**

Пользователь выбирает инструмент (точка, линия, фигура, рисование) с панели инструментов. Программа переключается в соответствующий режим с помощью методо.

1. **Рисование:**

При нажатии на холст (on\_click) создаётся объект выбранного класса (Point, Line, Circle и т.д.), и вызывается метод draw() для его отрисовки. В случае сложных фигур (линия, круг, квадрат) временный объект создаётся в «on\_click», обновляется в «on\_drag и фиксируется в «on\_release».

1. **Отмена действий:**

Все нарисованные элементы сохраняются в списках shapes и points по мере их создания. Кнопка «Очистить окно» удаляет все объекты с холста и списков.

1. **Текст программы**

import tkinter as tk

from tkinter import simpledialog

class Shape:

def draw(self,canvas):

raise NotImplementedError("Метод draw должен быть переопределен в подклассе")

class Point(Shape):

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

self.x = x

self.y = y

def draw(self,canvas):

canvas.create\_oval(self.x - 2, self.y - 2, self.x + 2, self.y + 2, fill='black')

class Circle(Shape):

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

self.x = x

self.y = y

self.radius = radius

def draw(self,canvas):

canvas.create\_oval(self.x - self.radius, self.y - self.radius,

self.x + self.radius, self.y + self.radius,

outline='black')

class Square(Shape):

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

self.x = x

self.y = y

self.side = side

def draw(self,canvas):

canvas.create\_rectangle(self.x - self.side / 2, self.y - self.side / 2,

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

outline='black')

class GraphicEditor:

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

self.root = root

self.root.title("Графический редактор")

self.canvas = tk.Canvas(self.root, width=600, height=400, bg='white')

self.canvas.pack()

self.button\_frame = tk.Frame(self.root)

self.button\_frame.pack()

self.dot\_button = tk.Button(self.button\_frame, text="Рисовать точку", command=self.add\_point\_mode)

self.dot\_button.pack(side=tk.LEFT)

self.circle\_button = tk.Button(self.button\_frame, text="Рисовать круг", command=self.draw\_circle\_mode)

self.circle\_button.pack(side=tk.LEFT)

self.square\_button = tk.Button(self.button\_frame, text="Рисовать квадрат", command=self.draw\_square\_mode)

self.square\_button.pack(side=tk.LEFT)

self.connect\_button = tk.Button(self.root, text="Соединить точки", command=self.connect\_points)

self.connect\_button.pack()

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

self.btn\_clear.pack(side=tk.LEFT)

self.mode = 'point'

self.shapes = []

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

def add\_point\_mode(self):

self.mode = 'point'

def draw\_circle\_mode(self):

self.mode = 'circle'

def draw\_square\_mode(self):

self.mode = 'square'

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

if self.mode == 'point':

point = Point(self.canvas, event.x, event.y)

point.draw(self.canvas)

self.shapes.append(point)

elif self.mode == 'circle':

radius = simpledialog.askinteger("Введите радиус", "Радиус круга:", minvalue=1)

if radius is not None:

circle = Circle(self.canvas, event.x, event.y, radius)

circle.draw(self.canvas)

self.shapes.append(circle)

elif self.mode == 'square':

side = simpledialog.askinteger("Введите размер", "Размер квадрата:", minvalue=1)

if side is not None:

square = Square(self.canvas, event.x, event.y, side)

square.draw(self.canvas)

self.shapes.append(square)

def connect\_points(self):

points\_to\_connect = [(shape.x, shape.y) for shape in self.shapes if isinstance(shape, Point)]

if len(points\_to\_connect) > 1:

for i in range(len(points\_to\_connect) - 1):

self.canvas.create\_line(points\_to\_connect[i], points\_to\_connect[i + 1], fill='black')

def clear\_canvas(self):

self.canvas.delete("all")

self.shapes.clear()

self.points.clear()

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

root = tk.Tk()

app = GraphicEditor(root)

root.mainloop()