Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8
дисциплины
«Объектно-ориентированное программирование»
Вариант 15

Выполнил:
Степанов Леонид Викторович
3 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Проверил:
Богданов Сергей Сергеевич,
Ассистент департамента цифровых, робототехнических систем и электроники

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: Обработкасобытий и рисование в Tkinter

Цель: приобретение навыков улучшения графического интерфейса пользователя GUI с помощью обработки событий и рисования, реализованных в пакете Tkinter языка программирования Python версии 3.x.

Порядок выполнения работы:

Решите задачу: напишите программу, состоящую из двух списков Listbox . В первом будет, например, перечень товаров, заданный программно. Второй изначально пуст, пусть это будет перечень покупок. При клике на одну кнопку товар должен переходить из одного списка в другой. При клике на вторую кнопку – возвращаться (человек передумал покупать). Предусмотрите возможность множественного выбора элементов списка и их перемещения.

prog > task1.py > ...

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import tkinter as tk
from tkinter import Listbox, Button, END, MULTIPLE


def main():
    # Функция для перемещения выбранных товаров из одного списка в другой
    def move_items(source_listbox, target_listbox):
        selected_items = source_listbox.curselection()
        for index in selected_items[
            ::-1
        ]:  # Перемещение с конца, чтобы не нарушить индексы
            item = source_listbox.get(index)
            target_listbox.insert(END, item)
            source_listbox.delete(index)

    # Создание основного окна
    root = tk.Tk()
    root.title("Перемещение товаров")

    # Список товаров
    products = ["Яблоки", "Бананы", "Молоко", "Хлеб", "Яйца", "Сыр"]

    # Создание Listbox для товаров
    product_listbox = Listbox(root, selectmode=MULTIPLE)
    for product in products:
        product_listbox.insert(END, product)
    product_listbox.pack(side=tk.LEFT, padx=10, pady=10)

    # Создание пустого Listbox для покупок
    shopping_listbox = Listbox(root, selectmode=MULTIPLE)
    shopping_listbox.pack(side=tk.RIGHT, padx=10, pady=10)

    # Кнопки для перемещения товаров
    add_button = Button(
        root,
        text="Добавить в покупки",
        command=lambda: move_items(product_listbox, shopping_listbox),
    )
    add_button.pack(pady=5)

    remove_button = Button(
        root,
        text="Вернуть в товары",
        command=lambda: move_items(shopping_listbox, product_listbox),
    )
    remove_button.pack(pady=5)

    # Запуск основного цикла приложения
    root.mainloop()


if __name__ == "__main__":
    main()
```

Рисунок 1 – Код программы

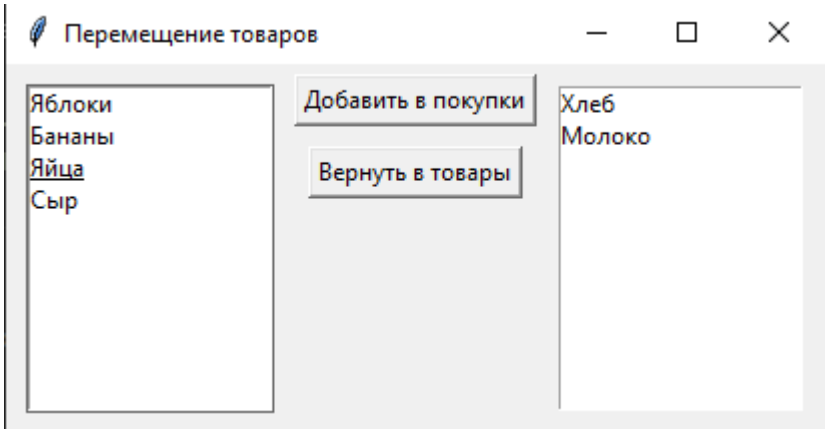

Рисунок 2 – Результат работы программы

Решите задачу: напишите программу по следующему описанию. Нажатие Enter в однострочном текстовом поле приводит к перемещению текста из него в список (экземпляр Listbox).

При двойном клике (<Double-Button-1>) по элементу-строке списка, она должна копироваться в текстовое поле.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import tkinter as tk


def main():
    # Функция для добавления текста из Entry в Listbox
    def add_to_list(event):
        text = entry.get()
        if text:  # Проверяем, что текст не пустой
            listbox.insert(tk.END, text)
            entry.delete(0, tk.END)  # Очищаем поле ввода

    # Функция для копирования текста из Listbox в Entry
    def copy_to_entry(event):
        selected_item_index = listbox.curselection()
        if selected_item_index:  # Проверяем, что выбран элемент
            selected_item = listbox.get(selected_item_index)
            entry.delete(0, tk.END)  # Очищаем поле ввода
            entry.insert(0, selected_item)  # Вставляем выбранный элемент

    # Создание основного окна
    root = tk.Tk()
    root.title("Список товаров")

    # Создание однострочного текстового поля
    entry = tk.Entry(root)
    entry.pack(pady=10)
    entry.bind("<Return>", add_to_list)  # Привязываем нажатие Enter к функции

    # Создание Listbox для отображения списка
    listbox = tk.Listbox(root)
    listbox.pack(pady=10)
    listbox.bind(
        "<Double-Button-1>", copy_to_entry
    )  # Привязываем двойной клик к функции

    # Запуск основного цикла приложения
    root.mainloop()


if __name__ == "__main__":
    main()

```

Рисунок 3 – Код программы

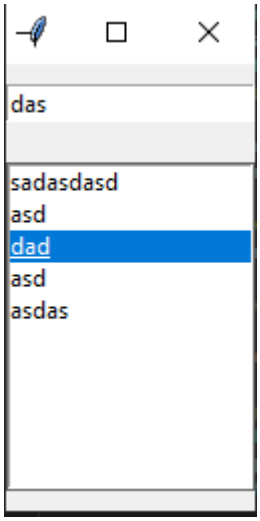

Рисунок 4 – Результат работы программы

Решите задачу: напишите программу по описанию. Размеры многострочного текстового поля определяются значениями, введенными в однострочные текстовые поля. Изменение размера происходит при нажатии мышью на кнопку, а также при нажатии клавиши Enter.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import tkinter as tk


def main():
    def update_text_area():
        try:
            rows = int(rows_entry.get())
            cols = int(cols_entry.get())
            text_area.config(height=rows, width=cols)
        except ValueError:
            pass  # Игнорируем ошибки преобразования

    def on_focus_in(event):
        text_area.config(bg="white")

    def on_focus_out(event):
        text_area.config(bg="lightgrey")

    # Создаем главное окно
    root = tk.Tk()
    root.title("Изменение размера текстового поля")

    # Создаем однострочные текстовые поля для ввода размеров
    rows_entry = tk.Entry(root)
    rows_entry.pack(pady=5)
    rows_entry.insert(0, "Введите количество строк")

    cols_entry = tk.Entry(root)
    cols_entry.pack(pady=5)
    cols_entry.insert(0, "Введите количество столбцов")

    # Создаем кнопку для изменения размера
    resize_button = tk.Button(root, text="Изменить размер", command=update_text_area)
    resize_button.pack(pady=5)

    # Создаем многострочное текстовое поле
    text_area = tk.Text(root, bg="lightgrey", height=5, width=30)
    text_area.pack(pady=5)

    # Привязываем события фокуса
    text_area.bind("<FocusIn>", on_focus_in)
    text_area.bind("<FocusOut>", on_focus_out)

    # Обработка нажатия клавиши Enter
    def on_enter(event):
        update_text_area()

    text_area.bind("<Return>", on_enter)

    # Запускаем главный цикл
    root.mainloop()


if __name__ == "__main__":
    main()
```

Рисунок 5 – Код программы

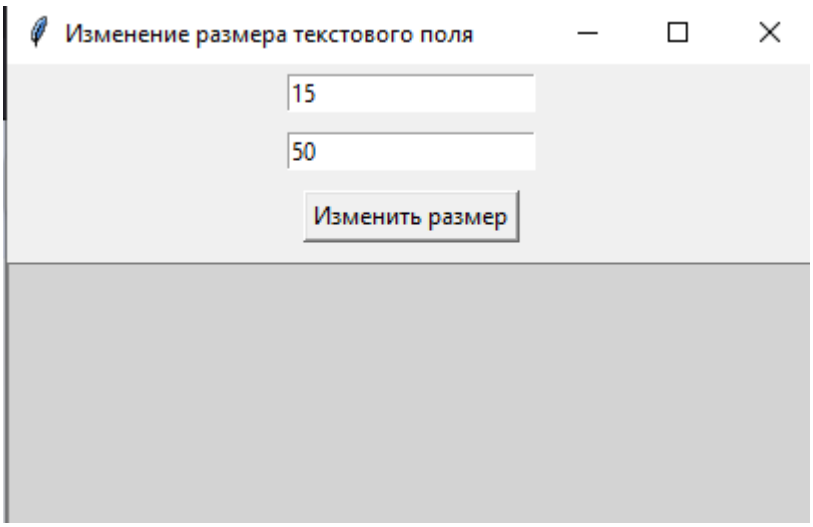

Рисунок 6– Результат работы программы

Решите задачу: Создайте на холсте подобное изображение (рис. 7), для создания травы используется цикл.

Рисунок 7 – Примерный результат

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import tkinter as tk


def main():
    # Создаем окно
    root = tk.Tk()
    root.title("Рисунок с домом и травой")

    # Создаем холст
    canvas = tk.Canvas(root, width=300, height=300, bg="white")
    canvas.pack()

    # Рисуем дом (основа и крыша)
    canvas.create_rectangle(
        100, 150, 200, 250, fill="lightblue", outline="lightblue"
    )  # Основа дома
    canvas.create_polygon(
        60, 150, 150, 100, 240, 150, fill="lightblue", outline="lightblue"
    )  # Крыша

    # Рисуем солнце
    canvas.create_oval(220, 20, 270, 70, fill="yellow", outline="yellow")  # Солнце

    # Рисуем траву с использованием цикла
    for x in range(0, 300, 10):  # С шагом 10 пикселей
        canvas.create_line(x, 300, x + 10, 260, fill="green", width=2)

    # Запускаем приложение
    root.mainloop()


if __name__ == "__main__":
    main()
```

Рисунок 8 – Код программы

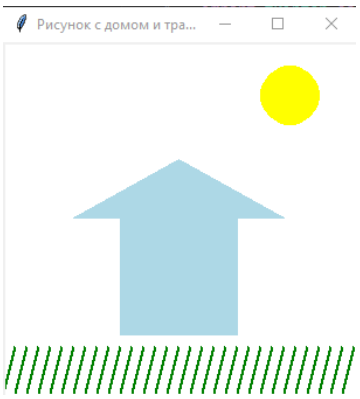

Рисунок 9 – Результат работы программы

Решите задачу: в данной программе создается анимация круга, который движется от левой границы холста до правой: Выражение c.coords(ball) возвращает список текущих координат объекта (в данном случае это ball). Третий элемент списка соответствует его второй координате x. Метод after вызывает функцию, переданную вторым аргументом, через количество миллисекунд, указанных первым аргументом. Изучите приведенную программу и самостоятельно запрограммируйте постепенное движение фигуры в ту точку холста, где пользователь кликает левой кнопкой мыши. Координаты события хранятся в его атрибутах x и y (event.x, event.y).

```
prog > tasks.py > main
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from tkinter import *


def main():
    def move_to_click(event):
        """Запускаем движение шара к месту клика."""
        global target_x, target_y
        target_x, target_y = event.x, event.y
        move_ball()

    def move_ball():
        """Перемещаем шар к указанным координатам."""
        global target_x, target_y
        current_coords = c.coords(ball)
        current_x, current_y = (current_coords[0] + current_coords[2]) / 2, (
            current_coords[1] + current_coords[3]
        ) / 2

        # Вычисляем шаги движения
        dx = target_x - current_x
        dy = target_y - current_y
        step_x = 1 if dx > 0 else -1 if dx < 0 else 0
        step_y = 1 if dy > 0 else -1 if dy < 0 else 0

        # Если шар не достиг цели, двигаем его
        if abs(dx) > 1 or abs(dy) > 1:
            c.move(ball, step_x, step_y)
            root.after(10, move_ball)

    # Создаем окно и холст
    root = Tk()
    c = Canvas(root, width=300, height=200, bg="white")
    c.pack()

    # Создаем круг
    ball = c.create_oval(0, 100, 40, 140, fill="green")

    # Обрабатываем клик мыши
    c.bind("<Button-1>", move_to_click)

    # Инициализация
    target_x, target_y = 0, 0

    # Запускаем приложение
    root.mainloop()


if __name__ == "__main__":
    main()

```

Рисунок 10 – Код программы

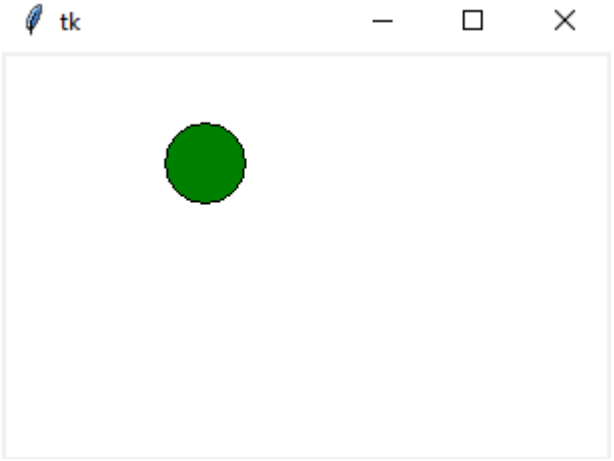

Рисунок 11 – Результат работы программы

Ссылка на github: https://github.com/FiaLDI/lab4.8

Контрольные вопросы

1. Каково назначение виджета ListBox?

Виджет Listbox используется для отображения списка элементов, из которого пользователь может выбирать один или несколько элементов. Это удобно для создания интерфейсов, где требуется выбор из множества вариантов.

2. Каким образом осуществляется связывание событие или действие с виджетом Tkinter?

Связывание событий с виджетами осуществляется с помощью метода bind. Этот метод связывает конкретное событие (например, щелчок мыши) с функцией-обработчиком, которая вызывается при возникновении этого события.

3. Какие существуют типы событий в Tkinter? Приведите примеры.

События в Tkinter включают:

Мышь:

1) <Button-1> — левый щелчок мыши.
2) <Button-3> — правый щелчок мыши.
3) <Double-Button-1> — двойной щелчок левой кнопкой мыши.
4) <Motion> — движение мыши.

Клавиатура:

1) <Key> — нажатие любой клавиши.
2) <KeyPress-a> — нажатие конкретной клавиши (например, "a").
3) <Return> — нажатие клавиши Enter.

Фокус:

1) <FocusIn> — получение фокуса виджетом.
2) <FocusOut> — потеря фокуса.

Окно:

1) <Configure> — изменение размера окна.
2) <Destroy> — закрытие окна.

4. Как обрабатываются события в Tkinter?

События обрабатываются с помощью функций-обработчиков, которые привязываются к конкретным событиям с помощью метода bind. При срабатывании события в обработчик передается объект события (event), содержащий информацию о типе события, его координатах, кнопках мыши и т.д.

5. Как обрабатываются события мыши в Tkinter?

Для обработки событий мыши используются идентификаторы событий, такие как:

1) <Button-1> — левый щелчок.
2) <Button-2> — средний щелчок.
3) <Button-3> — правый щелчок.
4) <Motion> — движение мыши. Эти события можно привязывать к виджетам с помощью bind.

6. Каким образом можно отображать графические примитивы в Tkinter?

Для отображения графических примитивов используется виджет Canvas. На холсте можно рисовать линии, прямоугольники, овалы, многоугольники и текст с помощью методов Canvas.

7. Перечислите основные методы для отображения графических примитивов в Tkinter.

create_line(x1, y1, x2, y2, **options) — рисует линию. create_rectangle(x1, y1, x2, y2, **options) — рисует прямоугольник. create_oval(x1, y1, x2, y2, **options) — рисует овал. create_polygon(x1, y1, ..., xn, yn, **options) — рисует многоугольник. create_text(x, y, text="Текст", **options) — рисует текст.

8. Каким образом можно обратиться к ранее созданным фигурам на холсте?

Каждая фигура, созданная на холсте, имеет уникальный идентификатор, который возвращается методом создания. К фигурам можно обращаться по этому идентификатору.

9. Каково назначение тэгов в Tkinter?

Тэги используются для группировки объектов на холсте и управления ими одновременно. Тэги задаются с помощью параметра tags при создании фигуры.

Вывод: в ходе выполнения работы были приобретены навыки улучшения графического интерфейса пользователя GUI с помощью обработки событий и рисования, реализованных в пакете Tkinter языка программирования Python версии 3.x.