МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №4.8

Обработка событий и рисование в Tkinter

По дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ИВТ-б-о-20-1

Новикова В.И. « »_____20__г.

Подпись студента _____

Работа защищена « »_____20__г.

Проверил Воронкин Р. А. _____

(подпись)

Ставрополь 2022

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

Ход работы:

1. Создала общедоступный репозиторий и клонировала на его на локальный сервер.
   https://github.com/Valentina1502/4_8_Tkinter
2. После изучения теоретического материала и методических рекомендаций приступила к выполнению заданий.

Задание 1. 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

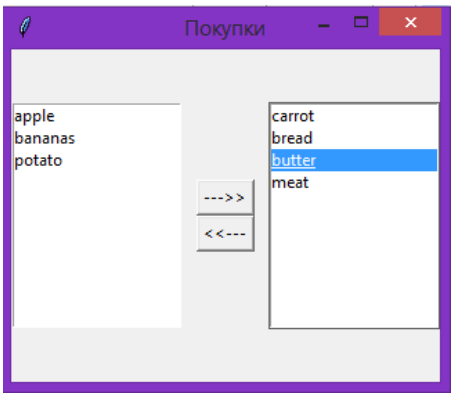

Рисунок 1 – Задание 1

Код:

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

from tkinter import *


def add():
    product = []
    select = list(box.curselection())
    select.reverse()
    for item in select:
        ad = box.get(item)
```

```
        product.append(ad)
    for val in product:
        box2.insert(0, val)
    for n in select:
        box.delete(n)


def back():
    product = []
    select = list(box2.curselection())
    select.reverse()
    for item in select:
        ad = box2.get(item)
        product.append(ad)
    for val in product:
        box.insert(0, val)
    for n in select:
        box2.delete(n)


if __name__ == '__main__':
    root = Tk()
    root.title('Покупки')
    root.geometry('310x240')

    products = [
        'apple',
        'bananas',
        'carrot',
        'bread',
        'butter',
        'meat',
        'potato'
    ]

    box = Listbox(selectmode=EXTENDED)
    box.pack(side=LEFT)
    box2 = Listbox(selectmode=EXTENDED)
    box2.pack(side=RIGHT)

    f = Frame()
    f.pack(side=LEFT, padx=10)
    Button(f, text="--->>", command=add).pack(fill=X)
    Button(f, text="<<---", command=back).pack(fill=X)

    for i in products:
        box.insert(END, i)
    root.mainloop()
```

Задание 2. Напишите программу по следующему описанию. Нажатие Enter в однострочном текстовом поле приводит к перемещению текста из него в список (экземпляр Listbox). При двойном клике по элементу-строке списка, она должна копироваться в текстовое поле

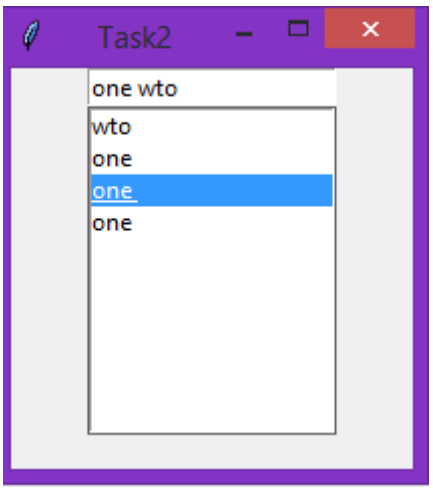

Рисунок 2 – Задание 2

Код:

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

from tkinter import Tk, Listbox, Entry


if __name__ == "__main__":
    root = Tk()
    root.geometry("200x200+90+90")
    root.title("Task2")

    ent1 = Entry()
    lb1 = Listbox()

    ent1.pack()
    lb1.pack()
    ent1.bind('<Return>', lambda e:  lb1.insert(0, ent1.get()))
    lb1.bind('<Double-Button-1>', lambda e: ent1.insert(0,
lb1.get(lb1.curselection())))

    root.mainloop()
```

Задание 3. 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.

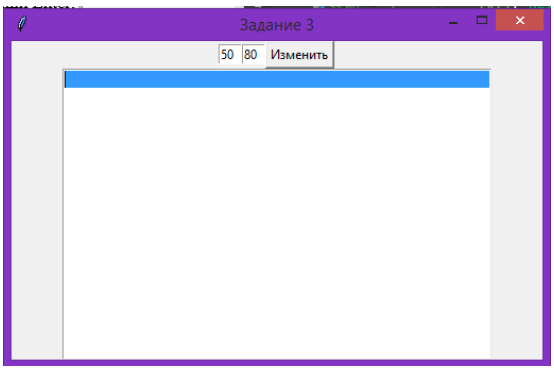

Рисунок 3 – Задание 3

Код:

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

from tkinter import Tk, Frame, Entry, Button, Text, LEFT, RIGHT


def button_click(event):
    txt1['width'] = ent1.get()
    txt1['height'] = ent2.get()


def focus_change(event, color):
    txt1['bg'] = color


if __name__ == "__main__":
    root = Tk()
    root.geometry("500x300+300+300")
    root.title("Задание 3")

    f = Frame()
    f.pack()
    ent1 = Entry(f, width=3)
    ent2 = Entry(f, width=3)
    bt1 = Button(f, text='Изменить')
    txt1 = Text(width=10, height=10, bg='lightgrey')

    bt1.bind('<Button-1>', button_click)
    ent1.bind('<Return>', button_click)
    ent2.bind('<Return>', button_click)
    bt1.bind('<Return>', button_click)
    root.bind('<Return>', button_click)
    txt1.bind('<Return>', button_click)
    txt1.bind('<FocusIn>', lambda e, c="white": focus_change(e, c))
    txt1.bind('<FocusOut>', lambda e, c="lightgrey": focus_change(e, c))

    ent1.pack(side=LEFT)
    bt1.pack(side=RIGHT)
    ent2.pack(side=LEFT)
    txt1.pack()

    root.mainloop()
```

Задание 4. Создайте на холсте изображение

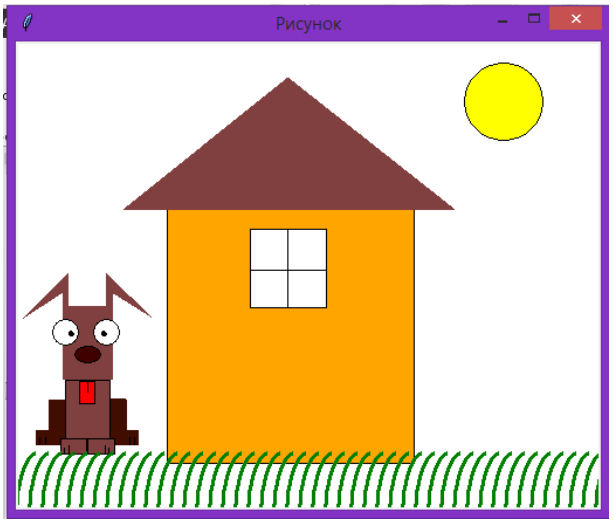

Рисунок 4 – Задание 4

Код:

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

from tkinter import Tk, Canvas, ARC


if __name__ == '__main__':

    root = Tk()
    root.title('Рисунок')
    root.geometry('500x400')

    c = Canvas(root, width=500, height=400, bg='white')
    c.pack()

    # Создаем основу дома и солнце
    circle = c.create_oval(383, 18, 450, 84, fill="#FFFF00")
    rectangle = c.create_rectangle(129, 143, 340, 360, fill='orange')
    triangle = c.create_polygon(232, 30, 90, 144, 375, 143, fill='#804040')

    # Создаем собаку
    triangle2 = c.create_polygon(3, 238, 45, 197, 45, 226,
                                 77, 226, 76, 197, 118, 237,
                                 83, 215, 83, 289, 40, 289,
                                 40, 214, fill='#804040')
    triangle3 = c.create_polygon(28, 305, 28, 332, 17, 332,
                                 17, 345, 42, 345, 42, 306, fill="#400d00")
    triangle4 = c.create_polygon(80, 305, 94, 305, 95, 332,
                                 105, 332, 105, 345, 80, 345, fill="#400d00")
    rectangle2 = c.create_rectangle(42, 289, 80, 345, fill='#804040')
    rectangle3 = c.create_rectangle(54, 290, 67, 309, fill='red')
    rectangle4 = c.create_rectangle(38, 339, 60, 352, fill='#804040')
    rectangle5 = c.create_rectangle(61, 339, 84, 352, fill='#804040')
    line1 = c.create_line(20, 338, 20, 344)
    line2 = c.create_line(25, 338, 25, 344)
    line3 = c.create_line(41, 347, 41, 353)
    line4 = c.create_line(45, 345, 45, 353)
    line5 = c.create_line(76, 345, 76, 353)
    line6 = c.create_line(79, 347, 79, 353)
    line7 = c.create_line(97, 338, 97, 345)
    line8 = c.create_line(101, 338, 101, 345)
    line9 = c.create_line(61, 289, 61, 300)
```

```
    circle2 = c.create_oval(50, 260, 72, 274, fill="#400000")
    circle5 = c.create_oval(31, 237, 53, 259, fill="white")
    circle6 = c.create_oval(45, 247, 49, 251, fill="black")
    circle7 = c.create_oval(66, 237, 88, 259, fill="white")
    circle8 = c.create_oval(73, 247, 77, 251, fill="black")

    # Детали к дому
    rectangle7 = c.create_rectangle(200, 160, 265, 227, fill='white')
    line10 = c.create_line(232, 160, 232, 227)
    line11 = c.create_line(200, 195, 265, 195)

    # Создаем цикл, для вывода травы
    x = 0
    while x < 500:
        c.create_arc(x, 455, x+40, 350, start=180, extent=-80, style=ARC,
width=3, outline='green')
        x += 11

    root.mainloop()
```

Задание 5. 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)

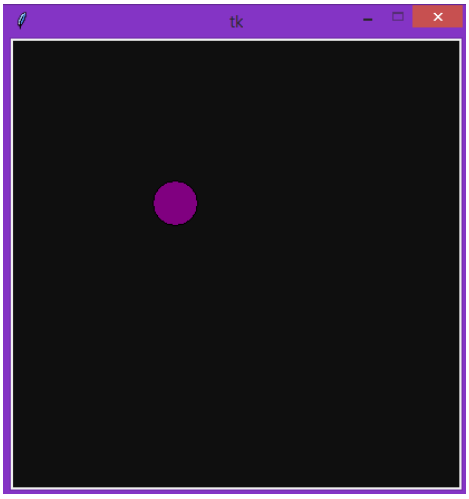

Рисунок 5 – Задание 5

Код:

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

from tkinter import Tk, Canvas
import math


class GameBall:
    def __init__(self, width=400, height=400):
        self.vx = 1
        self.vy = 1
        self.t = 0
```

```
        self.dt = 1
        self.i = 0
        self.height = height
        self.width = width
        self.c = Canvas(root, width=self.width, height=self.height,
bg="#0f0f0f")
        self.c.pack()
        self.rad = 20
        self.ball = self.c.create_oval(self.width // 2 - self.rad,
self.height // 2 - self.rad,
                                       self.width // 2 + self.rad,
self.height // 2 + self.rad, fill='purple')

        self.c.bind('<Button-1>', self.onclick)
        root.after(10, self.onframe)

    def onclick(self, event):
        self.p = self.c.coords(self.ball)
        self.dx = event.x - self.p[0]
        self.dy = event.y - self.p[1]

        self.r = math.sqrt(self.dx ** 2 + self.dy ** 2)

        self.vx = self.dx / self.r
        self.vy = self.dy / self.r

        self.t += self.dt
        self.p[0] += self.vx + 10
        self.p[1] += self.vy + 10

        if 200 > self.dx >= 100 or 200 > self.dy >= 100 or -200 < self.dx <=
-100 or -200 < self.dy <= -100:
            self.dt = 2
        elif 300 > self.dx >= 200 or 300 > self.dy >= 200 or -300 < self.dx
<= -200 or -300 < self.dy <= -200:
            self.dt = 4
        elif 500 > self.dx >= 300 or 500 > self.dy >= 300 or -500 < self.dx
<= -300 or -500 < self.dy <= -300:
            self.dt = 7
        elif self.dx >= 500 or self.dy >= 500 or self.dx <= -500 or self.dy
<= -500:
            self.dt = 10
        else:
            self.dt = 1
        print(self.dx, self.dy, self.dt)

    def onframe(self):
        self.c.move(self.ball, self.dt * self.vx, self.dt * self.vy)
        self.p = self.c.coords(self.ball)

        if self.p[1] < 0 or self.p[1] > self.height - 2 * self.rad:
            self.vy = -self.vy

        if self.p[0] < 0 or self.p[0] > self.width - 2 * self.rad:
            self.vx = -self.vx

        root.after(10, self.onframe)


if __name__ == '__main__':
    root = Tk()
    root.resizable(False, False)
    GameBall()
    root.mainloop()
```

Контрольные вопросы:

1. Каково назначение виджета ListBox?

От класса Listbox создаются списки – виджеты, внутри которых в столбик перечисляются элементы. При этом можно выбирать один или множество элементов списка

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

Events - события. В tkinte с помощью метода bind() между собой связываются виджет, событие и действие. Например, виджет – кнопка, событие – клик по ней левой кнопкой мыши, действие – отправка сообщения. Другой пример: виджет – текстовое поле, событие – нажатие Enter, действие – получение текста из поля методом get() для последующей обработки программой.

3. Какие существуют типы событий в Tkinter?

Можно выделить три основных типа событий: производимые мышью, нажатиями клавиш на клавиатуре, а также события, возникающие в результате изменения виджетов. Нередко обрабатываются сочетания. Например, клик мышью с зажатой клавишей на клавиатуре.

4. Как обрабатываются события в Tkinter?

Все события обрабатываются методом класса print_event(), который выводит тип события и положение мыши в консоли. Можете поэкспериментировать, нажимая на зеленую рамку мышью и двигая ею, пока она будет выводить сообщения события.

5. Как обрабатываются события мыши в Tkinter?

– <Button-1> – клик левой кнопкой мыши;

– <Button-2> – клик средней кнопкой мыши;

– <Button-3> – клик правой кнопкой мыши;

– <Double-Button-1> – двойной клик левой кнопкой мыши;

– <Motion> – движение мыши.

6. Каким образом можно отображать графические примитивы в Tkinter?

В tkinter от класса Canvas создаются объекты-холсты, на которых можно "рисовать", размещая различные фигуры и объекты. Делается это с помощью вызовов соответствующих методов. При создании экземпляра Canvas необходимо указать его ширину и высоту. При размещении геометрических примитивов и других объектов указываются их координаты на холсте. Точкой отсчета является верхний левый угол.

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

Методом create_polygon рисуется произвольный многоугольник путем задания координат каждой его точки. Метод create_oval создает эллипсы. При этом задаются координаты гипотетического прямоугольника, описывающего эллипс. Если нужно получить круг, то соответственно описываемый прямоугольник должен быть квадратом. Более сложные для понимания фигуры получаются при использовании метода create_arc . В зависимости от значения опции style можно получить сектор (по умолчанию), сегмент (CHORD) или дугу (ARC). Также как в случае create_oval координаты задают прямоугольник, в который вписана окружность (или эллипс), из которой "вырезают" сектор, сегмент или дугу. Опции start присваивается градус начала фигуры, extent определяет угол поворота.

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

В Tkinter существует два способа "пометить" фигуры, размещенные на холсте, – это идентификаторы и теги. Первые всегда уникальны для каждого объекта. Два объекта не могут иметь одни и тот же идентификатор. Теги не уникальны. Группа объектов на холсте может иметь один и тот же тег. Это дает возможность менять свойства всей группы. Отдельно взятая фигура на Canvas может иметь как идентификатор, так и тег.

9. Каково назначение тэгов в Tkinter?

В отличие от идентификаторов, которые являются уникальными для каждого объекта, один и тот же тег может присваиваться разным объектам. Дальнейшее обращение к такому тегу позволит изменить все объекты, в которых он был указан. Метод tag_bind позволяет привязать событие (например, щелчок кнопкой мыши) к определенной фигуре на Canvas. Таким образом, можно реализовать обращение к различным областям холста с помощью одного и того же события

Вывод: в ходе выполнения лабораторной работы были приобретены навыки построения графического интерфейса пользователя GUI с помощью пакета Tkinter языка программирования Python версии 3.x.