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 №4.7
дисциплины «Объектно-ориентированное программирование»
Вариант 9

Выполнил:
Иващенко Олег Андреевич
3 курс, группа ИВТ-б-о-22-1,
09.03.02 «Информационные и вычислительные машины»,
направленность (профиль)
«Программное обеспечение средств вычислительной техники и автоматизированных систем»

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович, доцент департамента цифровых, робототехнических систем и электроники

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: «Основы работы с Tkinter»

Цель: Приобретение навыков построения графического интерфейса пользователя GUI с помощью пакета Tkinter языка программирования Python версии 3.x.

Порядок выполнения работы:

Задача 1. Напишите простейший калькулятор, состоящий из двух текстовых полей, куда пользователь вводит числа, и четырех кнопок "+", "-", "*", "/". Результат вычисления должен отображаться в метке. Если арифметическое действие выполнить невозможно (например, если были введены буквы, а не числа), то в метке должно появляться слово "ошибка".

Таблица 1 – Листинг программы

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

"""
Напишите простейший калькулятор, состоящий из двух текстовых полей, куда
пользователь вводит числа, и четырех кнопок "+", "-", "*", "/". Результат
вычисления должен отображаться в метке. Если арифметическое действие выполнить
невозможно (например, если были введены буквы, а не числа), то в метке должно
появляться слово "ошибка".
"""

from tkinter import Tk, Entry, Frame, Button, Label, LEFT


def calculate(operation):
    """
    Выполняет арифметическую операцию над числами из текстовых полей.
    Результат отображается в метке.
    """
    try:
        num1 = float(entry1.get())
        num2 = float(entry2.get())

        if operation == '+':
            result = num1 + num2
        elif operation == '-':
            result = num1 - num2
        elif operation == '*':
            result = num1 * num2
        elif operation == '/':
            if num2 == 0:
```

```
            result = "ошибка"
        else:
            result = num1 / num2

        result_label.config(text=f"Результат: {result}")

    except ValueError:
        result_label.config(text="ошибка")


def main():
    """Основная функция для создания графического интерфейса."""
    global entry1, entry2, result_label

    root = Tk()
    root.title("Калькулятор")
    root.geometry("250x200")

    entry1 = Entry(root, width=20)
    entry1.pack(pady=5)

    entry2 = Entry(root, width=20)
    entry2.pack(pady=5)

    button_frame1 = Frame(root)
    button_frame1.pack()

    button_add = Button(button_frame1, text="+", width=10,
                        command=lambda: calculate('+'))
    button_add.pack(side=LEFT, padx=5, pady=5)

    button_sub = Button(button_frame1, text="-", width=10,
                        command=lambda: calculate('-'))
    button_sub.pack(side=LEFT, padx=5, pady=5)

    button_frame2 = Frame(root)
    button_frame2.pack()

    button_mul = Button(button_frame2, text="*", width=10,
                        command=lambda: calculate('*'))
    button_mul.pack(side=LEFT, padx=5, pady=5)

    button_div = Button(button_frame2, text="/", width=10,
                        command=lambda: calculate('/'))
    button_div.pack(side=LEFT, padx=5, pady=5)

    result_label = Label(root, text="Результат: ")
    result_label.pack(pady=10)

    root.mainloop()
```

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

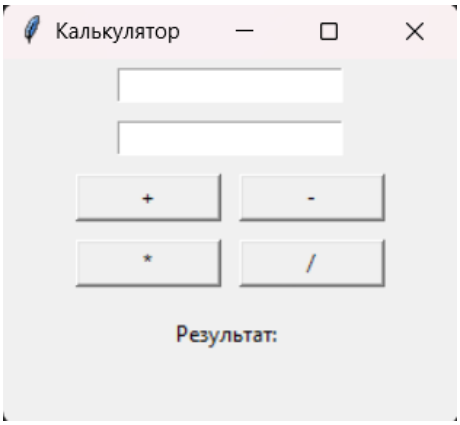

Рисунок 1.1 – Окно калькулятора

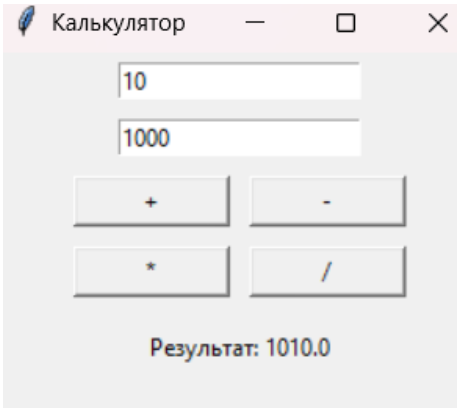

Рисунок 1.2 – Выполнение сложения

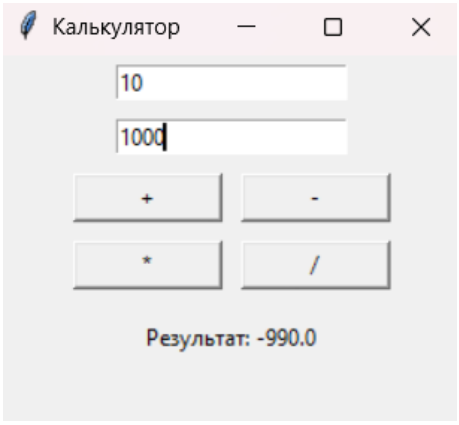

Рисунок 1.3 – Выполнение вычитания

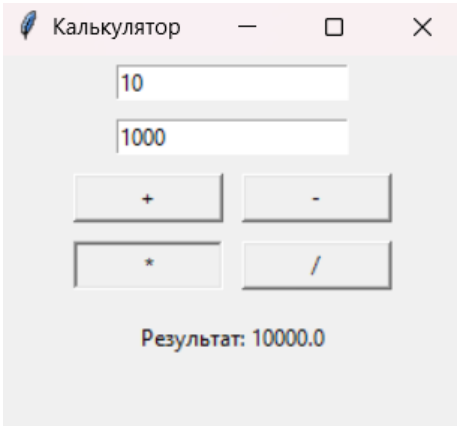

Рисунок 1.4 – Выполнение умножения

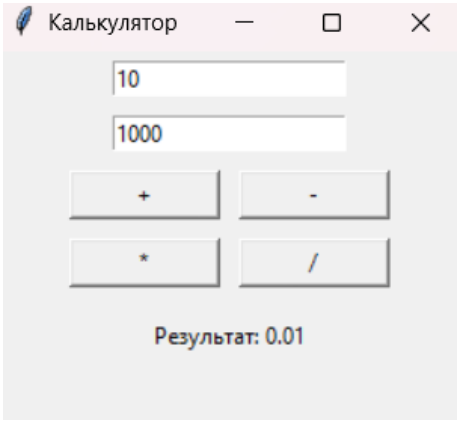

Рисунок 1.5 – Выполнение деления

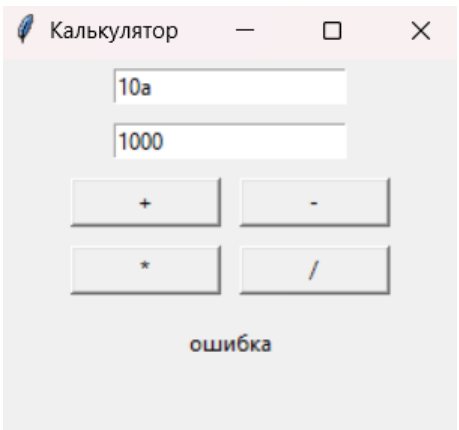

Рисунок 1.6 – Возникновении ошибки

Задача 2. Напишите программу, состоящую из семи кнопок, цвета которых соответствуют цветам радуги. При нажатии на ту или иную кнопку в текстовое поле должен вставляться код цвета, а в метку – название цвета.

Коды цветов в шестнадцатеричной кодировке: #ff0000 – красный, #ff7d00 – оранжевый, #ffff00 – желтый, #00ff00 – зеленый, #007dff – голубой, #0000ff – синий, #7d00ff – фиолетовый.

Таблица 2 – Листинг программы

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

"""
Напишите программу, состоящую из семи кнопок, цвета которых соответствуют
цветам радуги. При нажатии на ту или иную кнопку в текстовое поле должен
вставляться код цвета, а в метку – название цвета. Коды цветов в
шестнадцатеричной кодировке: #ff0000 – красный, #ff7d00 – оранжевый,
#ffff00 – желтый, #00ff00 – зеленый, #007dff – голубой, #0000ff – синий,
# #7d00ff – фиолетовый.
"""

from tkinter import Tk, Label, Entry, Button, END


def set_color(color_name, color_code):
    """
    Устанавливает код цвета в текстовое поле и название цвета в метку.
    """
    text_field.delete(0, END)
    text_field.insert(0, color_code)
    color_label.config(text=color_name)


def main():
    """Основная функция для создания графического интерфейса."""
    global text_field, color_label

    root = Tk()
    root.title("Цвета радуги")
    root.geometry("150x300")

    color_label = Label(root, text="Выберите цвет", font=("Arial", 14))
    color_label.pack(pady=10)

    text_field = Entry(root, width=15)
    text_field.pack(pady=10)

    colors = [
        ("Красный", "#ff0000"),
        ("Оранжевый", "#ff7f00"),
        ("Желтый", "#ffff00"),
        ("Зеленый", "#00ff00"),
        ("Голубой", "#007dff"),
        ("Синий", "#0000ff"),
```

```
        ("Фиолетовый", "#7400ff")
    ]

    for color_name, color_code in colors:
        button = Button(
            root,
            text="",
            bg=color_code,
            fg="black",
            width=15,
            command=lambda name=color_name, code=color_code: set_color(name, code)
        )
        button.pack(pady=2)

    root.mainloop()


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

Рисунок 2.1 – Окно программы при запуске

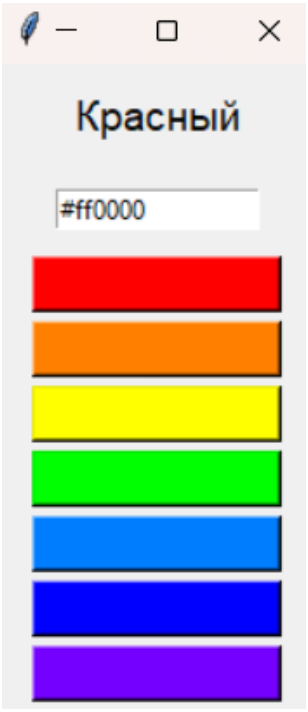

Рисунок 2.2 – Вывод программы при нажатии на кнопку «Красный»

Задача 3. Перепишите программу из пункта 8 так, чтобы интерфейс выглядел примерно следующим образом:

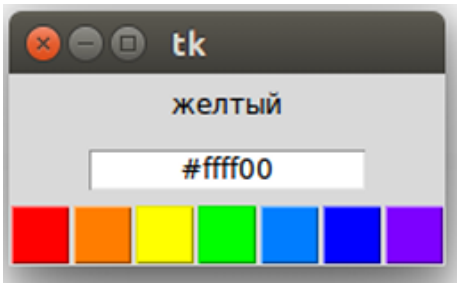

Таблица 3 – Листинг программы

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

"""
Перепишите программу из пункта 8 так, чтобы кнопки были в ряд.
"""

from tkinter import Tk, Label, Entry, Button, Frame, END, LEFT


def set_color(color_name, color_code):
    """
    Устанавливает код цвета в текстовое поле и название цвета в метку.
    """
```

```
    text_field.delete(0, END)
    text_field.insert(0, color_code)
    color_label.config(text=color_name)


def main():
    """Основная функция для создания графического интерфейса."""
    global text_field, color_label

    root = Tk()
    root.title("Цвета радуги")
    root.geometry("300x100")

    color_label = Label(root, text="Выберите цвет", font=("Arial", 12))
    color_label.pack(pady=5)

    text_field = Entry(root, width=20)
    text_field.pack(pady=5)

    colors = [
        ("Красный", "#ff0000"),
        ("Оранжевый", "#ff7f00"),
        ("Желтый", "#ffff00"),
        ("Зеленый", "#00ff00"),
        ("Голубой", "#007dff"),
        ("Синий", "#0000ff"),
        ("Фиолетовый", "#7400ff")
    ]

    button_frame = Frame(root)
    button_frame.pack()

    for color_name, color_code in colors:
        button = Button(
            button_frame,
            text="",
            bg=color_code,
            fg="black",
            width=3,
            command=lambda name=color_name, code=color_code: set_color(name, code)
        )
        button.pack(side=LEFT, padx=5, pady=5)

    root.mainloop()


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

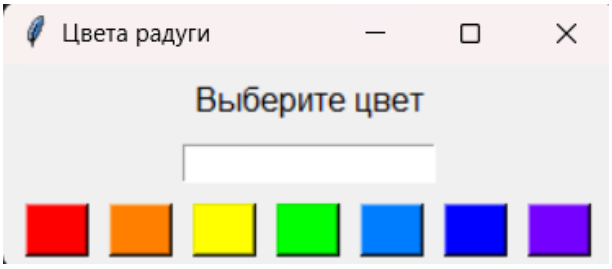

Рисунок 3.1 – Окно программы при запуске

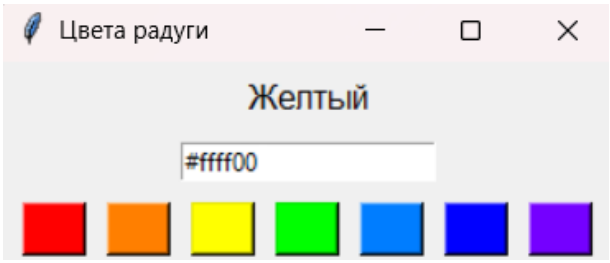

Рисунок 3.2 – Вывод программы

Задача 4. Напишите программу, состоящую из однострочного и многострочного текстовых полей и двух кнопок "Открыть" и "Сохранить". При клике на первую должен открываться на чтение файл, чье имя указано в поле класса Entry, а содержимое файла должно загружаться в поле типа Text. При клике на вторую кнопку текст, введенный пользователем в экземпляр Text, должен сохраняться в файле под именем, которое пользователь указал в однострочном текстовом поле. Файлы будут читаться и записываться в том же каталоге, что и файл скрипта, если указывать имена файлов без адреса. Для выполнения практической работы вам понадобится функция open языка Python и методы файловых объектов чтения и записи.

Таблица 4 – Листинг программы

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

"""
Напишите программу, состоящую из однострочного и многострочного текстовых
полей и двух кнопок "Открыть" и "Сохранить". При клике на первую должен
открываться на чтение файл, чье имя указано в поле класса Entry, а содержимое
файла должно загружаться в поле типа Text. При клике на вторую кнопку текст
введенный пользователем в экземпляр Text, должен сохраняться в файле под
именем, которое пользователь указал в однострочном текстовом поле. Файлы будут
читаться и записываться в том же каталоге, что и файл скрипта, если указывать
```

```
имена файлов без адреса. Для выполнения практической работы вам понадобится
функция open языка Python и методы файловых объектов чтения и записи.
"""

from tkinter import Tk, Frame, Entry, Button, Text, END, LEFT
from tkinter import messagebox


def open_file():
    """
    Открывает файл и загружает его содержимое в текстовое поле.
    """
    try:
        filename = entry.get()
        with open(filename, "r", encoding="utf-8") as file:
            content = file.read()
            text_field.delete(1.0, END)
            text_field.insert(1.0, content)
    except FileNotFoundError:
        messagebox.showerror("Ошибка", "Файл не найден!")
    except Exception as e:
        messagebox.showerror("Ошибка", f"Произошла ошибка: {e}")


def save_file():
    """
    Сохраняет содержимое текстового поля в файл.
    """
    try:
        filename = entry.get()
        content = text_field.get(1.0, END)
        with open(filename, "w", encoding="utf-8") as file:
            file.write(content)
        messagebox.showinfo("Успех", "Файл успешно сохранен!")
    except Exception as e:
        messagebox.showerror("Ошибка", f"Произошла ошибка: {e}")


def main():
    """Основная функция для создания графического интерфейса."""
    global entry, text_field

    root = Tk()
    root.title("Редактор файлов")
    root.geometry("600x400")

    input_frame = Frame(root)
    input_frame.pack(pady=10)

    entry = Entry(input_frame, width=40)
    entry.pack(side=LEFT, padx=5)
```

```
    open_button = Button(input_frame, text="Открыть", width=10,
                         command=open_file)
    open_button.pack(side=LEFT, padx=5)

    save_button = Button(input_frame, text="Сохранить", width=10,
                         command=save_file)
    save_button.pack(side=LEFT, padx=5)

    text_field = Text(root, width=60, height=20)
    text_field.pack(pady=10)

    root.mainloop()


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

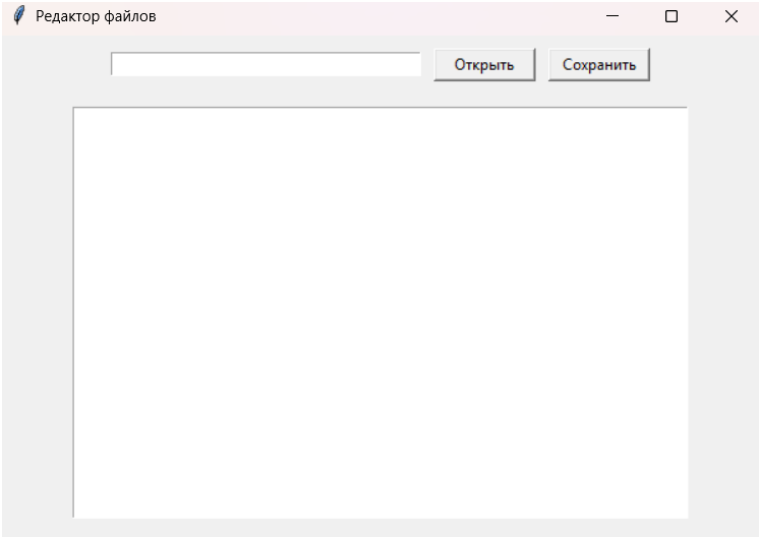

Рисунок 4.1 – Окно редактора файлов

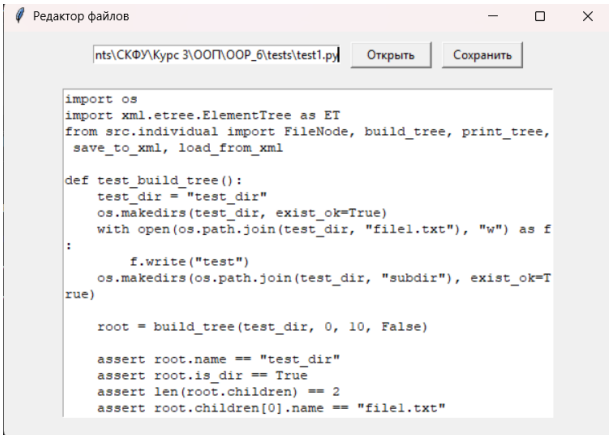

Рисунок 4.2 – Открытие выбранного файла

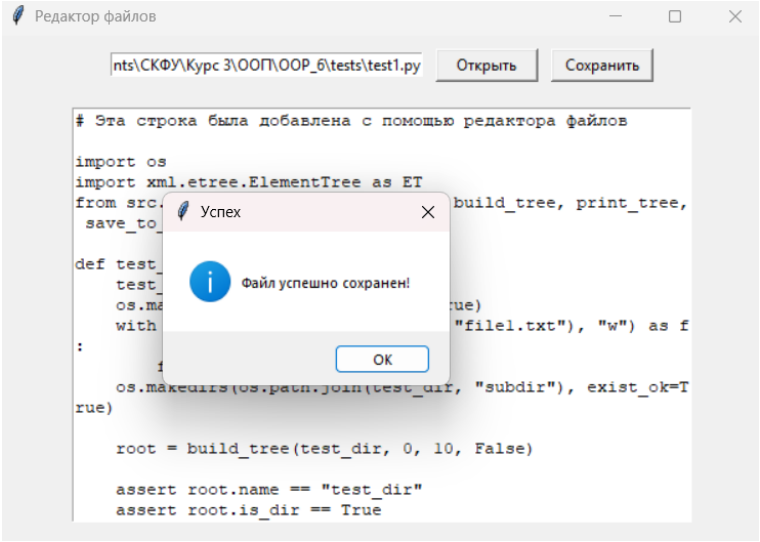

Рисунок 4.3 – Сохранение файла

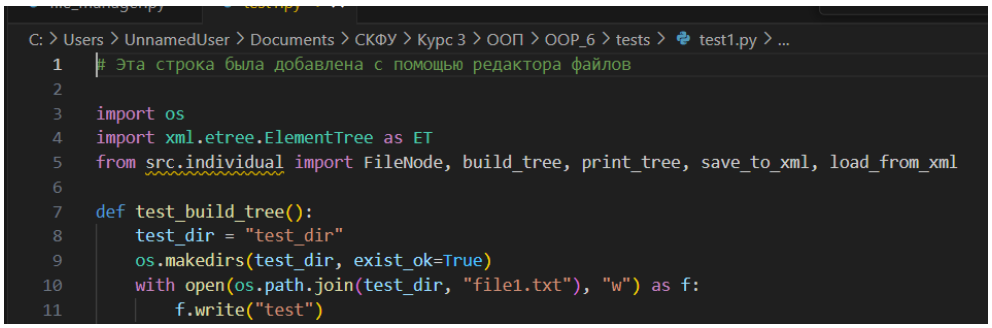

Рисунок 4.4 – Сохранённая строка

Задача 5. Виджеты Radiobatton и Checkbutton поддерживают большинство свойств оформления внешнего вида, которые есть у других элементов графического интерфейса. При этом у Radiobutton есть особое свойство indicatoron. По-умолчанию он равен единице, в этом случае радиокнопка выглядит как нормальная радиокнопка. Однако если присвоить этой опции ноль, то виджет Radiobutton становится похожим на обычную кнопку по внешнему виду. Но не по смыслу. Напишите программу, в которой имеется несколько объединенных в группу радиокнопок, индикатор которых выключен (indicatoron=0). Если какая-нибудь кнопка включается, то в метке должна отображаться соответствующая ей информация. Обычных кнопок в окне быть не должно.

Таблица 5 – Листинг программы

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

"""
Виджеты Radiobatton и Checkbutton поддерживают большинство свойств
оформления внешнего вида, которые есть у других элементов графического
интерфейса. При этом у Radiobutton есть особое свойство indicatoron.
По-умолчанию он равен единице, в этом случае радиокнопка выглядит как
нормальная радиокнопка. Однако если присвоить этой опции ноль, то виджет
Radiobutton становится похожим на обычную кнопку по внешнему виду. Но не по
смыслу. Напишите программу, в которой имеется несколько объединенных в группу
радиокнопок, индикатор которых выключен (indicatoron=0). Если какая-нибудь
кнопка включается, то в метке должна отображаться соответствующая ей
информация. Обычных кнопок в окне быть не должно.
"""

from tkinter import Tk, Frame, Radiobutton, Label, IntVar, W, LEFT, RIGHT


def update_label():
    """
    Обновляет метку в зависимости от выбранной радиокнопки.
    """
    selected_option = var.get()
    if selected_option == 1:
        label.config(text="+7 (912) 345-67-89")
    elif selected_option == 2:
        label.config(text="+7 (923) 456-78-90")
    elif selected_option == 3:
        label.config(text="+7 (934) 567-89-01")
    elif selected_option == 4:
        label.config(text="+7 (945) 678-90-12")


def main():
    """Основная функция для создания графического интерфейса."""
    global var, label

    root = Tk()
    root.title("Радиокнопки без индикатора")
    root.geometry("400x200")

    radio_frame = Frame(root)
    radio_frame.pack(side=LEFT, padx=10, pady=10)

    var = IntVar()

    radio1 = Radiobutton(
        radio_frame, text="Алексей", variable=var, value=1,
        indicatoron=0, command=update_label, width=10, height=2
```

```
    )
    radio1.pack(anchor=W, pady=0)

    radio2 = Radiobutton(
        radio_frame, text="Мария", variable=var, value=2,
        indicatoron=0, command=update_label, width=10, height=2
    )
    radio2.pack(anchor=W, pady=0)

    radio3 = Radiobutton(
        radio_frame, text="Дмитрий", variable=var, value=3,
        indicatoron=0, command=update_label, width=10, height=2
    )
    radio3.pack(anchor=W, pady=0)

    radio4 = Radiobutton(
        radio_frame, text="Анна", variable=var, value=4,
        indicatoron=0, command=update_label, width=10, height=2
    )
    radio4.pack(anchor=W, pady=0)

    info_frame = Frame(root)
    info_frame.pack(side=RIGHT, padx=10, pady=10)

    label = Label(info_frame, text="", font=("Arial", 12))
    label.pack()

    root.mainloop()


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

Рисунок 5.1 – Окно программы с радиокнопками

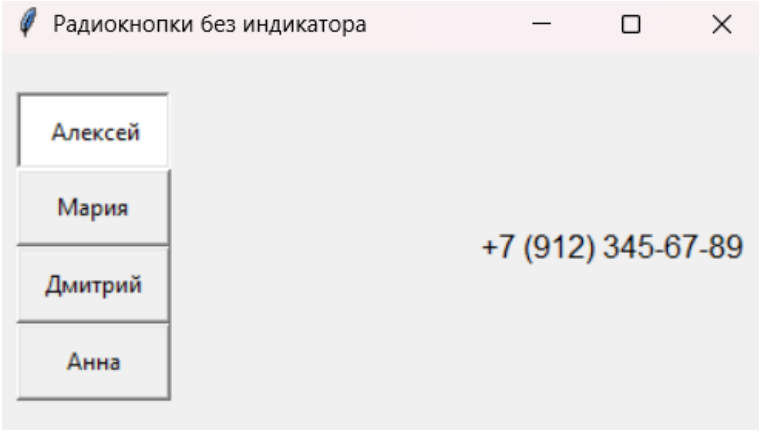

Рисунок 5.2 – Результат выполнения программы при выборе кнопки

Ответы на контрольные вопросы:

1. Какие существуют средства в стандартной библиотеке Python для построения графического интерфейса пользователя?

В стандартной библиотеке Python для создания графического интерфейса пользователя (GUI) доступны следующие средства:

- Tkinter: Стандартный модуль для создания GUI, основанный на библиотеке Tk.
- tkinter.ttk: Расширение Tkinter, предоставляющее улучшенные виджеты с современным внешним видом.
- tkinter.messagebox: Модуль для создания диалоговых окон с сообщениями.
- tkinter.filedialog: Модуль для работы с диалоговыми окнами выбора файлов.
- tkinter.colorchooser: Модуль для выбора цвета.

2. Что такое Tkinter?

Tkinter — это стандартный модуль Python для создания графического интерфейса пользователя (GUI). Он предоставляет интерфейс для работы с библиотекой Tk, которая является кроссплатформенной и позволяет создавать окна, кнопки, текстовые поля и другие элементы интерфейса.

3. Какие требуется выполнить шаги для построения графического интерфейса с помощью Tkinter?

Основные шаги для создания GUI с помощью Tkinter:

- Импортировать модуль tkinter.
- Создать главное окно с помощью класса Tk.
- Создать и настроить виджеты (например, кнопки, метки, текстовые поля).
- Разместить виджеты в окне с помощью менеджеров геометрии (pack, grid, place).
- Запустить главный цикл обработки событий с помощью метода mainloop().

4. Что такое цикл обработки событий?

Цикл обработки событий — это бесконечный цикл, который ожидает действий пользователя (например, нажатия кнопки, ввода текста) и реагирует на них. В Tkinter этот цикл запускается методом mainloop(). Он обрабатывает события, такие как клики мыши, нажатия клавиш и обновления интерфейса.

5. Каково назначение экземпляра класса Tk при построении графического интерфейса с помощью Tkinter?

Экземпляр класса Tk представляет главное окно приложения. Он является корневым элементом, на котором размещаются все остальные виджеты. Без создания экземпляра Tk невозможно построить графический интерфейс.

6. Для чего предназначены виджеты Button, Label, Entry и Text?

Button: Кнопка, которая выполняет действие при нажатии.

Label: Метка для отображения текста или изображения.

Entry: Однострочное текстовое поле для ввода данных.

Text: Многострочное текстовое поле для ввода и отображения текста.

7. Каково назначение метода pack() при построении графического интерфейса пользователя?

Метод pack() используется для автоматического размещения виджетов в окне. Он упорядочивает виджеты в соответствии с их размерами и параметрами, такими как side, fill, expand.

8. Как осуществляется управление размещением виджетов с помощью метода pack()?

Управление размещением виджетов с помощью pack() осуществляется через параметры:

- side: Определяет сторону, к которой прижимается виджет (LEFT, RIGHT, TOP, BOTTOM).
- fill: Определяет, как виджет заполняет доступное пространство (X, Y, BOTH, NONE).
- expand: Определяет, должен ли виджет расширяться при изменении размеров окна (True или False).
- padx/pady: Задает отступы вокруг виджета.

9. Как осуществляется управление полосами прокрутки в виджете Text?

Для управления полосами прокрутки в виджете Text используется виджет Scrollbar. Его связывают с Text через параметры:

- yscrollcommand: Связывает вертикальную полосу прокрутки с виджетом Text.
- xscrollcommand: Связывает горизонтальную полосу прокрутки с виджетом Text.

10. Для чего нужны тэги при работе с виджетом Text?

Тэги в виджете Text используются для форматирования текста. Они позволяют:

- Изменять цвет, шрифт и стиль текста.
- Добавлять гиперссылки.
- Выделять определенные участки текста.

11. Как осуществляется вставка виджетов в текстовое поле?

Вставка виджетов в текстовое поле осуществляется с помощью метода window_create().

12. Для чего предназначены виджеты Radiobutton и Checkbutton?

Radiobutton: Используется для выбора одного варианта из нескольких (переключатель).

Checkbutton: Используется для выбора нескольких вариантов (флажок).

13. Что такое переменные Tkinter и для чего они нужны?

Переменные Tkinter — это специальные объекты (IntVar, StringVar, BooleanVar и др.), которые используются для хранения и управления значениями виджетов. Они позволяют:

- Связывать значения виджетов с переменными
- Автоматически обновлять виджеты при изменении значений переменных.

14. Как осуществляется связь переменных Tkinter с виджетами Radiobutton и Checkbutton?

Связь переменных Tkinter с Radiobutton и Checkbutton осуществляется через параметры:

- variable: Связывает виджет с переменной.
- value: Определяет значение, которое будет присвоено переменной при выборе виджета.

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

Ссылка на репозиторий GitHub:

IUnnamedUserI/OOP_7: About Объектно-ориентированное программирование. Лабораторная работа №7