Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №30
дисциплины «Основы программной инженерии»

Выполнил:
Плугатырев Владислав Алексеевич
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и сопровождение программного обеспечения», очная форма обучения

_____
(подпись)

Проверил Воронкин Роман Александрович
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: Основы работы с Tkinter

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

Ход работы.

1. Создание репозитория.

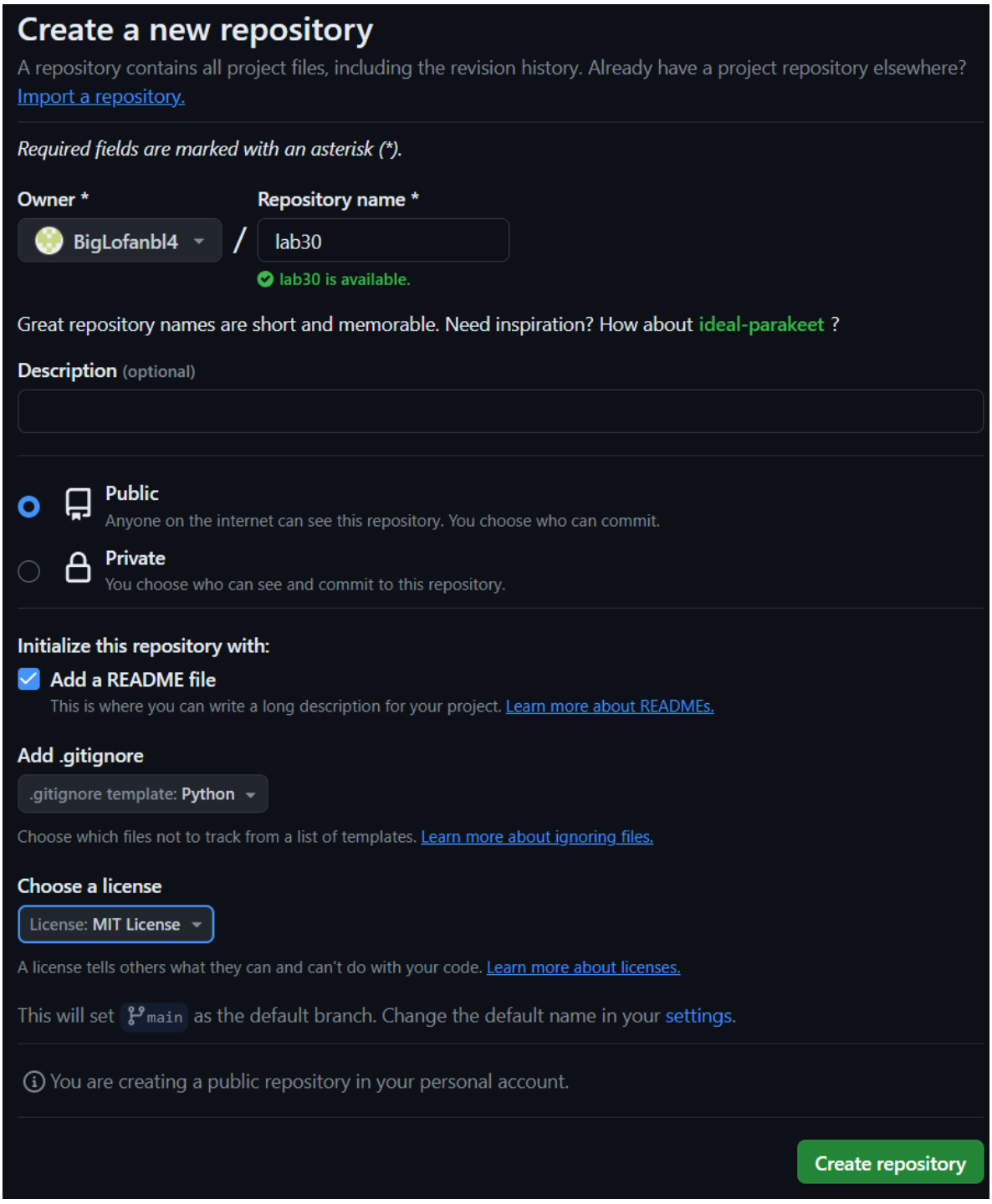

Рисунок 1.1 – Создание репозитория

2. Выполнение первого задания: 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".

```python
import tkinter as tk

def calculate(operation):
    try:
        num1 = float(entry_num1.get())
        num2 = float(entry_num2.get())

        match operation:
            case "+":
                result = num1 + num2
            case "-":
                result = num1 - num2
            case "*":
                result = num1 * num2
            case "/":
                result = num1 / num2
        label_result.config(text=f"Результат: {result}")
    except (ValueError, ZeroDivisionError):
        label_result.config(text="Ошибка")


if __name__ == "__main__":
    root = tk.Tk()
    root.title("Калькулятор")

    entry_num1 = tk.Entry(root)
    entry_num2 = tk.Entry(root)
    entry_num1.pack()
    entry_num2.pack()

    btn_add = tk.Button(root, text="+", command=lambda: calculate('+'))
    btn_sub = tk.Button(root, text="-", command=lambda: calculate('-'))
    btn_mul = tk.Button(root, text="*", command=lambda: calculate('*'))
    btn_div = tk.Button(root, text="/", command=lambda: calculate('/'))
    btn_add.pack()
    btn_sub.pack()
    btn_mul.pack()
    btn_div.pack()

    label_result = tk.Label(root, text="Результат: ")
    label_result.pack()

    root.mainloop()
```

Рисунок 2.1 – Код программы

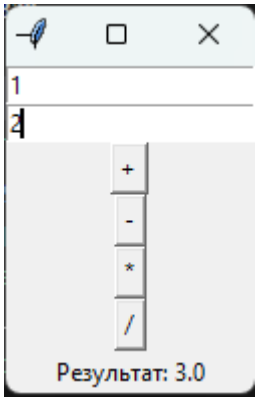

Рисунок 2.2 – Результат

3. Выполнение второго задания: 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.

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

import tkinter as tk

def set_color(hex_color):
    color_code_entry.delete(0, tk.END)
    color_code_entry.insert(0, hex_color)

if __name__ == "__main__":
    root = tk.Tk()
    root.title("Цвета радуги")

    color_code_entry = tk.Entry(root, justify="center")
    color_code_entry.pack()

    rainbow_colors = {
        "Красный": "#ff0000",
        "Оранжевый": "#ff7d00",
        "Желтый": "#ffff00",
        "Зеленый": "#00ff00",
        "Голубой": "#007dff",
        "Синий": "#0000ff",
        "Фиолетовый": "#7d00ff"
    }

    for color_name, hex_color in rainbow_colors.items():
        button = tk.Button(root, bg=hex_color, command=lambda h=hex_color: set_color(h))
        button.pack(fill=tk.X)

    root.mainloop()
```

Рисунок 3.1 – Код программы

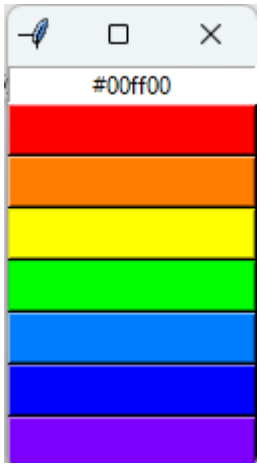

Рисунок 3.2 – Результат

4. Выполнение третьего задания: перепишите программу из предыдущего задания под требуемый интерфейс.

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

import tkinter as tk

def set_color(hex_color):
    color_name_label.config(text=rainbow_colors[hex_color])
    color_code_entry.delete(0, tk.END)
    color_code_entry.insert(0, hex_color)


if __name__ == "__main__":
    root = tk.Tk()
    root.title("Цвета радуги")

    buttons_frame = tk.Frame(root)
    buttons_frame.pack(side=tk.BOTTOM, padx=2, pady=2)

    rainbow_colors = {
        "#ff0000": "Красный",
        "#ff7d00": "Оранжевый",
        "#ffff00": "Желтый",
        "#00ff00": "Зеленый",
        "#007dff": "Голубой",
        "#0000ff": "Синий",
        "#7d00ff": "Фиолетовый"
    }


    for hex_color in rainbow_colors:
        button = tk.Button(buttons_frame, bg=hex_color, width=2, command=lambda c=hex_color: set_color(c))
        button.pack(side=tk.LEFT, padx=1, pady=1)

    color_code_entry = tk.Entry(root, justify='center')
    color_code_entry.pack()

    color_name_label = tk.Label(root, text="")
    color_name_label.pack()

    root.mainloop()
```

Рисунок 4.1 – Код программы

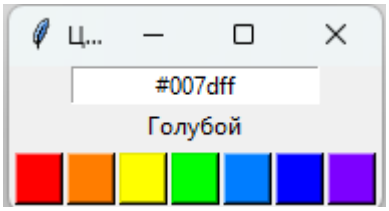

Рисунок 4.2 – Результат

5. Выполнение четвертого задания: 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.

```python
import tkinter as tk

def open_file():
    file_name = entry.get()
    try:
        with open(file_name, 'r', encoding="utf8") as file:
            content = file.read()
            text.delete('1.0', tk.END)
            text.insert(tk.END, content)
    except FileNotFoundError:
        text.delete('1.0', tk.END)
        text.insert(tk.END, "Файл не найден.")

def save_file():
    file_name = entry.get()
    content = text.get('1.0', tk.END)
    with open(file_name, 'w') as file:
        file.write(content)


if __name__ == "__main__":
    root = tk.Tk()
    root.title("Текстовый редактор")

    entry = tk.Entry(root)
    entry.pack(pady=5)

    text = tk.Text(root, height=10, width=50)
    text.pack(pady=5)

    open_button = tk.Button(root, text="Открыть", command=open_file)
    open_button.pack(pady=5)

    save_button = tk.Button(root, text="Сохранить", command=save_file)
    save_button.pack(pady=5)

    root.mainloop()
```

Рисунок 5.1 – Код программы

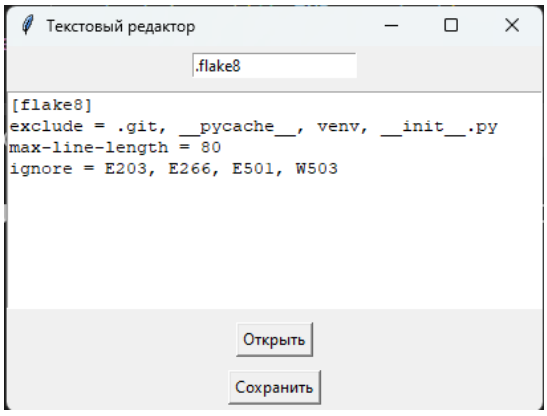

Рисунок 5.2 – Результат

6. Выполнение пятого задания: 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.

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

import tkinter as tk


def select():
    label.config(text=var.get())


if __name__ == "__main__":
    root = tk.Tk()
    root.title("Radiobutton с indicatoron=0")

    frame_radio_buttons = tk.Frame(root)
    frame_radio_buttons.pack(side=tk.LEFT, fill=tk.Y, expand=False)

    frame_output = tk.Frame(root)
    frame_output.pack(side=tk.RIGHT, fill=tk.X, anchor="center", expand=True)

    var = tk.StringVar(value="Выберите опцию")

    label = tk.Label(frame_output, text=var.get(), justify="center")
    label.pack(padx=10)

    radiobuttons = [
        ("Опция 1", "Опция 1"),
        ("Опция 2", "Опция 2"),
        ("Опция 3", "Опция 3"),
    ]

    for text, value in radiobuttons:
        rb = tk.Radiobutton(
            frame_radio_buttons,
            text=text,
            variable=var,
            value=value,
            indicatoron=0,
            command=select,
            width=20,
            height=2,
        )
        rb.pack()

    root.mainloop()
```

Рисунок 6.1 – Код программы

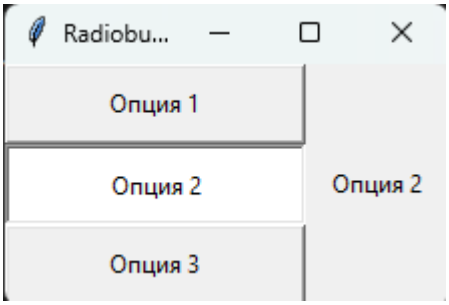

Рисунок 6.2 – Результат

Ответы на контрольные вопросы

1. Средства в стандартной библиотеке Python для построения графического интерфейса пользователя:

- «Tkinter»: наиболее известная библиотека для создания графического интерфейса пользователя (GUI).

- «turtle»: утилита для рисования, которая также может использоваться для создания простого GUI.

- «ttk»: подмодуль «Tkinter», предоставляющий доступ к тематическим виджетам, которые выглядят более современно.

2. Tkinter: это стандартная библиотека GUI Python, которая предоставляет интерфейс к библиотеке Tk GUI toolkit. Это набор инструментов для построения визуальных интерфейсов, который входит в стандартную библиотеку и является наиболее часто используемым для создания GUI в Python.

3. Шаги для построения GUI с помощью Tkinter:

- Импортировать «Tkinter».

- Создать главное окно (инстанс класса «Tk»).

- Добавить необходимые виджеты (например, кнопки, текстовые поля, метки).

- Конфигурировать виджеты с помощью опций (например, размеры, цвета).

- Разместить виджеты в главном окне с помощью менеджеров геометрии («pack», «grid», «place»).

- Запустить главный цикл обработки событий («mainloop»).

4. Цикл обработки событий: это бесконечный цикл, в котором приложение ожидает и отвечает на события, такие как нажатия клавиш, движения мыши или нажатия кнопок GUI. В Tkinter это достигается вызовом метода «mainloop».

5. Экземпляр класса Tk: является главным окном приложения Tkinter. Это корневой элемент GUI, от которого наследуются и в который помещаются все другие виджеты.

6. Назначение виджетов:

- «Button»: кнопка, которую пользователь может нажать для выполнения действия.

- «Label»: метка, которая может отображать текст или изображение.

- «Entry»: текстовое поле для ввода одной строки текста.

- «Text»: текстовое поле для ввода и отображения нескольких строк текста.

7. Метод pack(): это один из менеджеров геометрии в Tkinter, который упаковывает виджеты в контейнер (родительский виджет или окно) и размещает их один за другим в свободном пространстве.

8. Управление размещением виджетов с помощью метода pack(): метод «pack()» может принимать различные параметры для контроля размещения, такие как «side» (сторона размещения), «fill» (заполнение пространства), «expand» (растягивание) и «padx/pady» (отступы).

9. Управление полосами прокрутки в виджете Text: для этого создаются экземпляры «Scrollbar» и связываются с виджетом «Text» через конфигурацию команд прокрутки («xscrollcommand», «yscrollcommand») за счёт вызова методов «set» и «config».

10. Тэги в виджете Text: позволяют стилизовать определенные части текста внутри «Text», можно задать различные атрибуты стиля, такие как шрифт, цвет, выравнивание и другие.

11. Вставка виджетов в текстовое поле: виджет «Text» допускает вставку других виджетов Tkinter с помощью метода «window_create», позволяя разместить, например, кнопки или метки внутри области текста.

12. Виджеты Radiobutton и Checkbutton: используются для предоставления пользователям выбора. «Radiobutton» позволяет выбрать один

вариант из множества, «Checkbutton» — для включения или отключения опции.

13. Переменные Tkinter: это специальные объекты (например, «StringVar», «IntVar», «BooleanVar»), которые могут быть связаны с виджетами для хранения их значения и обеспечения возможности их динамического обновления.

14. Связь переменных Tkinter с Radiobutton и Checkbutton: осуществляется через параметр «variable» при создании экземпляров этих виджетов. Переменная хранит текущее значение и автоматически обновляется при изменении состояния связанных виджетов.