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 №220дисциплины «Основы программной инженерии»**

Выполнил:

Джараян Арег Александрович

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**: Лабораторная работа 4.7 Основы работы с Tkinter.

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

1. Создание нового репозитория с лицензией MIT.

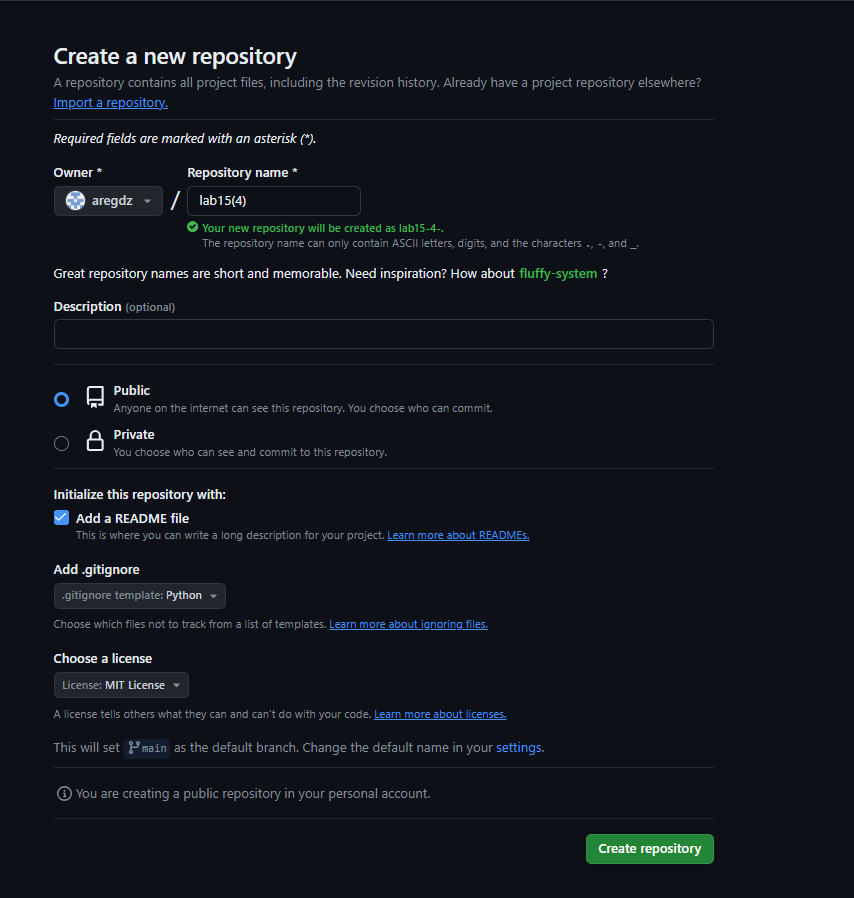


Рисунок 1 – создание репозитория

1. Клонировал репозиторий на рабочий ПК.

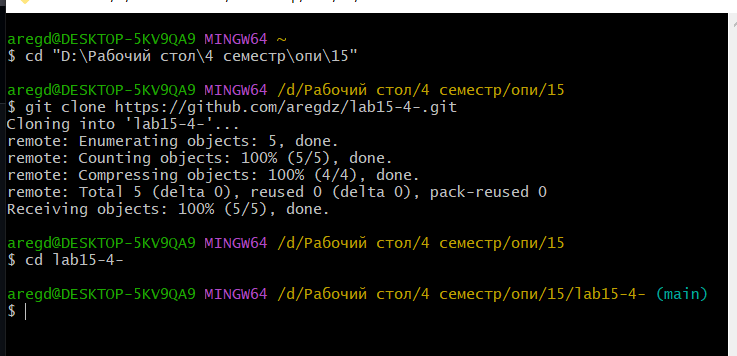
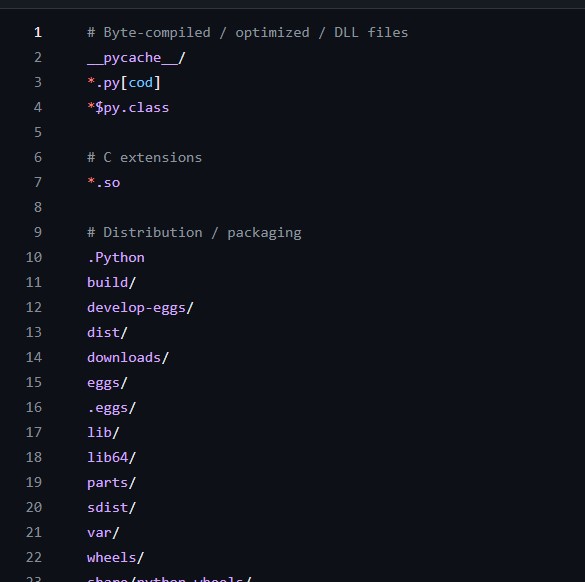


Рисунок 2 – клонирование репозитория

1. Дополнил файл .gitignore необходимыми инструкциями.

Рисунок 4 – Файл .gitignore

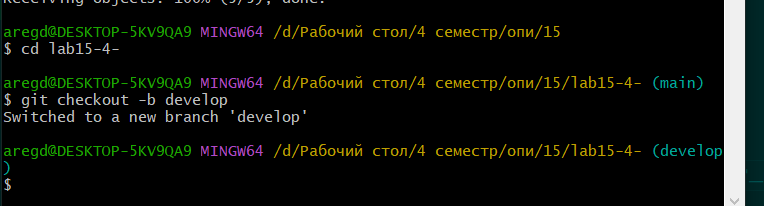


Рисунок 4 – организация ветки

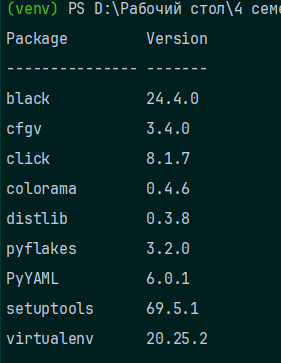


Рисунок 5 – создание виртуального окружения

4.Отработал примеры лабораторной работы.

5. Решите задачу: 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".

*#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
  
from* tkinter *import* \*  
  
*def* calculate():  
 *try*:  
 num1 = *float*(entry1.get())  
 num2 = *float*(entry2.get())  
 operator = operator\_var.get()  
  
 *if* operator == '+':  
 result = num1 + num2  
 *elif* operator == '-':  
 result = num1 - num2  
 *elif* operator == '\*':  
 result = num1 \* num2  
 *elif* operator == '/':  
 *if* num2 != 0:  
 result = num1 / num2  
 *else*:  
 result = "ошибка: деление на ноль"  
 *else*:  
 result = "ошибка: некорректный оператор"  
  
 result\_label.config(text=*str*(result))  
 *except ValueError*:  
 result\_label.config(text="ошибка: введите числа")  
  
 *def* set\_operator(op):  
 operator\_var.set(op)  
  
*if* \_\_name\_\_ == "\_\_main\_\_":  
 root = Tk()  
 root.title("Калькулятор")  
  
 frame = Frame(root)  
 frame.pack(padx=10, pady=10)  
  
 entry1 = Entry(frame, width=10)  
 entry1.pack()  
  
 operator\_var = StringVar()  
 operator\_var.set('+')  
 operator\_menu = OptionMenu(frame, operator\_var, '+', '-', '\*', '/')  
 operator\_menu.pack()  
  
 entry2 = Entry(frame, width=10)  
 entry2.pack()  
  
 calculate\_button = Button(frame, text='Вычислить', command=calculate)  
 calculate\_button.pack(pady=5)  
  
 equal\_label = Label(frame, text='=')  
 equal\_label.pack()  
  
 result\_label = Label(frame, text='')  
 result\_label.pack()  
  
 root.mainloop()

Рисунок 6 – Задание 1

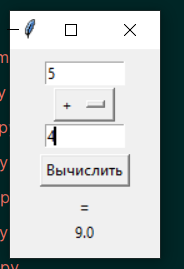


Рисунок 7 – пример выполнения примера 2

6. Решите задачу: 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.

*#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
  
from* tkinter *import* \*  
  
*def* change\_color1():  
 l1["text"] = "Красный"  
 l2["text"] = "#ff0000"  
  
*def* change\_color2():  
 l1["text"] = "Оранжевый"  
 l2["text"] = "#ff7d00"  
  
*def* change\_color3():  
 l1["text"] = "Желтый"  
 l2["text"] = "#ffff00"  
  
*def* change\_color4():  
 l1["text"] = "Зеленый"  
 l2["text"] = "#00ff00"  
  
*def* change\_color5():  
 l1["text"] = "Голубой"  
 l2["text"] = "#007dff"  
  
*def* change\_color6():  
 l1["text"] = "Синий"  
 l2["text"] = "#0000ff"  
  
*def* change\_color7():  
 l1["text"] = "Фиолетовый"  
 l2["text"] = "#7d00ff"  
  
  
*if* \_\_name\_\_ == "\_\_main\_\_":  
 root = Tk()  
  
 l1 = Label(text="")  
 l2 = Label(text="")  
 l1.pack()  
 l2.pack()  
  
 rainbow\_colors = ["red", "orange", "yellow", "green", "blue", "indigo", "violet"]  
  
 b1 = Button(text="Изменить", width=15, height=3, bg=rainbow\_colors[0], command=change\_color1)  
 b1.pack()  
  
 b2 = Button(text="Изменить", width=15, height=3, bg=rainbow\_colors[1], command=change\_color2)  
 b2.pack()  
  
 b3 = Button(text="Изменить", width=15, height=3, bg=rainbow\_colors[2], command=change\_color3)  
 b3.pack()  
  
 b4 = Button(text="Изменить", width=15, height=3, bg=rainbow\_colors[3], command=change\_color4)  
 b4.pack()  
  
 b5 = Button(text="Изменить", width=15, height=3, bg=rainbow\_colors[4], command=change\_color5)  
 b5.pack()  
  
 b6 = Button(text="Изменить", width=15, height=3, bg=rainbow\_colors[5], command=change\_color6)  
 b6.pack()  
  
 b7 = Button(text="Изменить", width=15, height=3, bg=rainbow\_colors[6], command=change\_color7)  
 b7.pack()  
  
 root.mainloop()

Рисунок 10 – код для выполнения задания 2

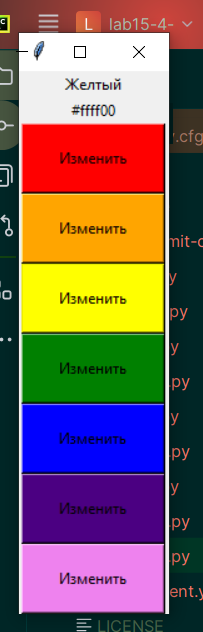


Рисунок 11 – пример выполнения задания 2

7.Решите задачу: перепишите программу из пункта 8 так, чтоб интерфейс выглядел примерно следующим образом.

*#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
  
from* tkinter *import* \*  
  
*def* change\_color1():  
 l1["text"] = "Красный"  
 l2["text"] = "#ff0000"  
  
*def* change\_color2():  
 l1["text"] = "Оранжевый"  
 l2["text"] = "#ff7d00"  
  
*def* change\_color3():  
 l1["text"] = "Желтый"  
 l2["text"] = "#ffff00"  
  
*def* change\_color4():  
 l1["text"] = "Зеленый"  
 l2["text"] = "#00ff00"  
  
*def* change\_color5():  
 l1["text"] = "Голубой"  
 l2["text"] = "#007dff"  
  
*def* change\_color6():  
 l1["text"] = "Синий"  
 l2["text"] = "#0000ff"  
  
*def* change\_color7():  
 l1["text"] = "Фиолетовый"  
 l2["text"] = "#7d00ff"  
  
  
*if* \_\_name\_\_ == "\_\_main\_\_":  
 root = Tk()  
  
 l1 = Label(text="")  
 l2 = Label(text="")  
 l1.pack()  
 l2.pack()  
  
 rainbow\_colors = ["red", "orange", "yellow", "green", "blue", "indigo", "violet"]  
  
 b1 = Button(width=3, height=3, bg=rainbow\_colors[0], command=change\_color1)  
 b1.pack(side=LEFT)  
  
 b2 = Button(width=3, height=3, bg=rainbow\_colors[1], command=change\_color2)  
 b2.pack(side=LEFT)  
  
 b3 = Button(width=3, height=3, bg=rainbow\_colors[2], command=change\_color3)  
 b3.pack(side=LEFT)  
  
 b4 = Button(width=3, height=3, bg=rainbow\_colors[3], command=change\_color4)  
 b4.pack(side=LEFT)  
  
 b5 = Button(width=3, height=3, bg=rainbow\_colors[4], command=change\_color5)  
 b5.pack(side=LEFT)  
  
 b6 = Button(width=3, height=3, bg=rainbow\_colors[5], command=change\_color6)  
 b6.pack(side=LEFT)  
  
 b7 = Button(width=3, height=3, bg=rainbow\_colors[6], command=change\_color7)  
 b7.pack(side=LEFT)  
  
 root.mainloop()

Рисунок 12 – код для выполнения задания 3

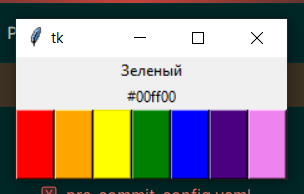


Рисунок 13 – пример выполнения задания 3

8. Решите задачу: 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.

*#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
  
from* tkinter *import* \*  
  
*def* open\_file():  
 filename = entry.get()  
 *try*:  
 *with open*(filename, 'r') *as* file:  
 content = file.read()  
 text.delete('1.0', END)  
 text.insert('1.0', content)  
 *except FileNotFoundError*:  
 text.delete('1.0', END)  
 text.insert('1.0', "Файл не найден.")  
  
*def* save\_file():  
 filename = entry.get()  
 content = text.get('1.0', END)  
 *with open*(filename, 'w') *as* file:  
 file.write(content)  
  
*if* \_\_name\_\_ == "\_\_main\_\_":  
 root = Tk()  
 root.title("Редактор файлов")  
  
 entry = Entry(root, width=50)  
 entry.pack(pady=5)  
  
 text = Text(root, width=50, height=20)  
 text.pack(pady=5)  
  
 open\_button = Button(root, text="Открыть", command=open\_file)  
 open\_button.pack(side=LEFT, padx=5)  
  
 save\_button = Button(root, text="Сохранить", command=save\_file)  
 save\_button.pack(side=LEFT, padx=5)  
  
 root.mainloop()

Рисунок 14 – код для выполнения задания 4

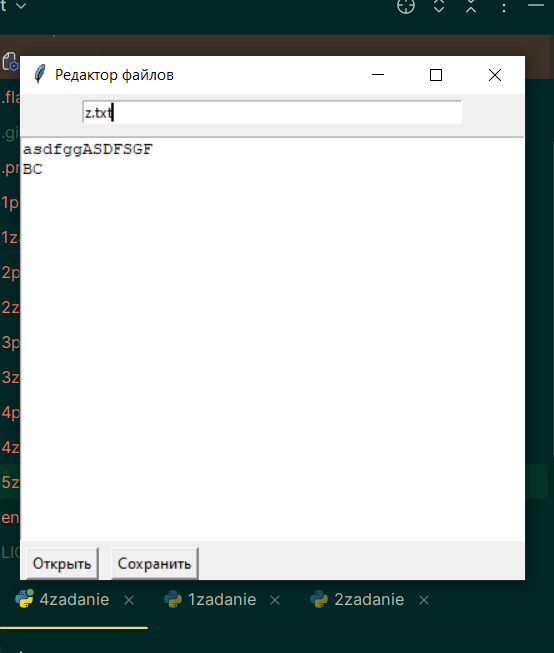


Рисунок 15 – пример выполнения задания 4

9. Решите задачу: 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.

*#!/usr/bin/env python3  
# -\*- coding: utf-8 -\*-  
  
import* tkinter *as* tk  
  
*def* show\_phone\_number():  
 selected\_person = var.get()  
 phone\_number = people\_data[selected\_person]  
 label.config(text=f"Номер телефона: {phone\_number}")  
  
*if* \_\_name\_\_ == "\_\_main\_\_":  
 people\_data = {  
 "Иван": "123-456-789",  
 "Мария": "987-654-321",  
 "Алексей": "555-555-555"  
 }  
  
 root = tk.Tk()  
 root.title("Информация о людях")  
  
 var = tk.StringVar()  
  
 max\_name\_length = *max*(*len*(person) *for* person *in* people\_data)  
  
 *for* i, person *in enumerate*(people\_data, start=1):  
 rb = tk.Radiobutton(root, text=person.ljust(max\_name\_length), variable=var, value=person, indicatoron=0, width=20, command=show\_phone\_number)  
 rb.grid(row=i, column=0, sticky=tk.W)  
  
 label = tk.Label(root, text="Выберите человека", font=("Arial", 12))  
 label.grid(row=0, column=1, padx=15, pady=(10, 0))  
  
 phone\_label = tk.Label(root, text="", font=("Arial", 12))  
 phone\_label.grid(row=*len*(people\_data)+1, columnspan=2, padx=15, pady=(10, 10))  
  
 root.mainloop()

Рисунок 16 – код для выполнения задания 5

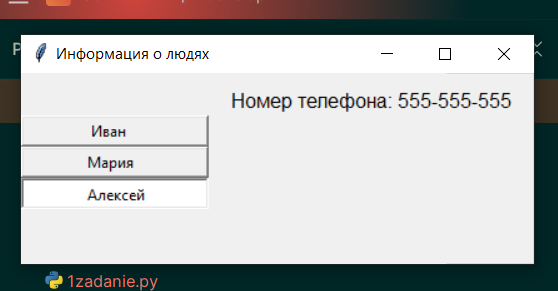


Рисунок 17 – пример выполнения задания 5

Контрольные вопросы:

1.В стандартной библиотеке Python для построения графического интерфейса пользователя используется модуль Tkinter.

2.Tkinter - это стандартный модуль Python для создания графического пользовательского интерфейса (GUI). Он предоставляет инструменты для создания различных виджетов и управления ими.

3.Для построения графического интерфейса с помощью Tkinter требуется выполнить следующие шаги:

* Создать экземпляр класса Tk.
* Добавить виджеты (кнопки, текстовые поля и т. д.).
* Разместить виджеты на окне с помощью методов pack(), grid() или place().
* Запустить цикл обработки событий методом mainloop().

4.Цикл обработки событий - это основной механизм, который ожидает события (например, нажатия кнопок, перемещения мыши) и вызывает соответствующие обработчики событий для выполнения соответствующих действий.

5.Экземпляр класса Tk используется для создания основного окна приложения и управления основными параметрами приложения.

6.Виджеты Button, Label, Entry и Text используются для следующих целей:

* Button: для создания кнопок.
* Label: для отображения текста или изображений.
* Entry: для ввода текста одной строкой.
* Text: для ввода и отображения многострочного текста.

7.Метод pack() используется для размещения виджетов в родительском контейнере (например, окне приложения). Он автоматически управляет распределением и размерами виджетов.

8.Управление размещением виджетов с помощью метода pack() осуществляется путем передачи определенных параметров, таких как side (сторона), fill (заполнение), expand (расширение) и др.

9.Управление полосами прокрутки в виджете Text осуществляется с помощью создания объектов Scrollbar и привязки их к виджету Text с помощью методов xscrollcommand и yscrollcommand.

10.Тэги используются для добавления атрибутов к определенным частям текста в виджете Text, таким как цвет, шрифт или стиль.

11.Виджеты можно вставлять в текстовое поле с помощью метода insert(), который позволяет указывать место вставки и содержимое.

12.Виджеты Radiobutton и Checkbutton используются для выбора одного или нескольких вариантов из предложенного списка соответственно.

13.Переменные Tkinter представляют собой специальные объекты, которые связываются с виджетами и хранят их текущее состояние или значение.

14.Связь переменных Tkinter с виджетами Radiobutton и Checkbutton осуществляется с помощью параметра variable, который указывает на переменную, с которой связывается состояние виджета.