

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

Институт перспективной инженерии
Департамента цифровых, роботехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7
дисциплины «Объектно-ориентированное программирование»

Выполнила:
Гайчук Дарья Дмитриевна
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)

Руководитель практики:
Воронкин Р.А.-доцент департамента
цифровых, роботехнических систем и
электроники института перспективной
инженерии

(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.

Ссылка на репозиторий: https://github.com/Ichizuchi/LR_4.7

Порядок выполнения работы:

1. Создала новый репозиторий и клонировала его на свой компьютер.

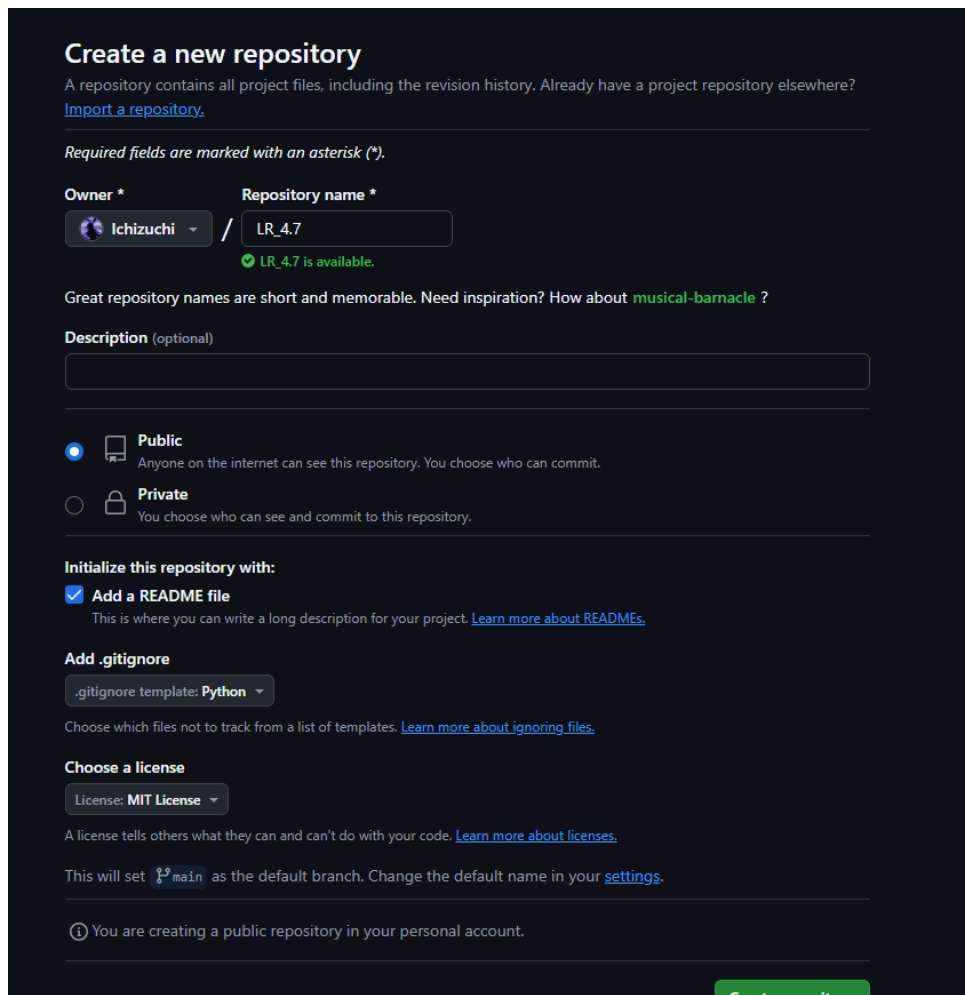


Рисунок 1. Создание репозитория

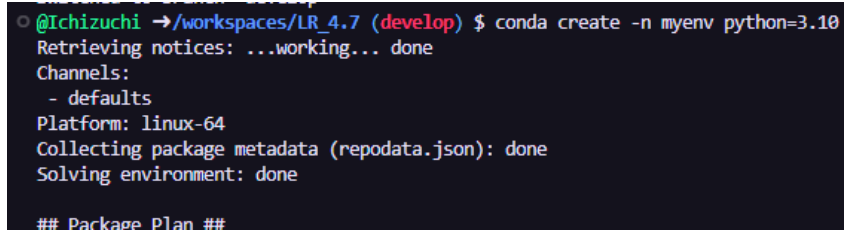
2. Клонировала репозиторий на свой компьютер.

```
@Ichizuchi →/workspaces/LR_4.7 (main) $ git branch develop
@Ichizuchi →/workspaces/LR_4.7 (main) $ git chekout develop
git: 'chekout' is not a git command. See 'git --help'.

The most similar command is
  checkout
@Ichizuchi →/workspaces/LR_4.7 (main) $ git checkout develop
Switched to branch 'develop'
```

Рисунок 2. Модель ветвления git-flow

3. Создала виртуальное окружение Anaconda с именем репозитория.



```
@Ichizuchi →/workspaces/LR_4.7 (develop) $ conda create -n myenv python=3.10
Retrieving notices: ...working... done
Channels:
 - defaults
Platform: linux-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##
```

Рисунок 3. Создание виртуального окружения

Выполнение заданий

Вариант 3

Задание №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».

```
Tasks > Task_1.py > ...
1  import tkinter as tk
2
3  root = tk.Tk()
4  root.title("Калькулятор")
5
6  entry1 = tk.Entry(root, width=10, font=("Arial", 14))
7  entry1.grid(row=0, column=0, padx=5, pady=5)
8
9  entry2 = tk.Entry(root, width=10, font=("Arial", 14))
10 entry2.grid(row=1, column=0, padx=5, pady=5)
11 |
12 # Функция для калькулятора
13 def calculate(operation):
14     try:
15         num1 = float(entry1.get())
16         num2 = float(entry2.get())
17
18         if operation == "+":
19             result = num1 + num2
20         elif operation == "-":
21             result = num1 - num2
22         elif operation == "*":
23             result = num1 * num2
24         elif operation == "/":
25             result = num1 / num2
26         result_label.config(text=f"Результат: {result}")
27     except ValueError:
28         result_label.config(text="Ошибка")
29     except ZeroDivisionError:
30         result_label.config(text="Деление на ноль")
31
32 buttons = ["+", "-", "*", "/"]
33 for i, text in enumerate(buttons):
34     button = tk.Button(root, text=text, width=5, height=2, font=("Arial", 14),
35                        command=lambda t=text: calculate(t))
36     button.grid(row=2+i, column=0, padx=5, pady=5)
37
38 # Метка отображения результата
39 result_label = tk.Label(root, text="Результат", font=("Arial", 14))
40 result_label.grid(row=6, column=0, pady=10)
41
42 # Запуск цикла
43 root.mainloop()
44 ..
ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  КОНСОЛЬ ОТЛАДКИ  ТЕРМИНАЛ  ПОРТЫ
```

Рисунок 6. Код программы

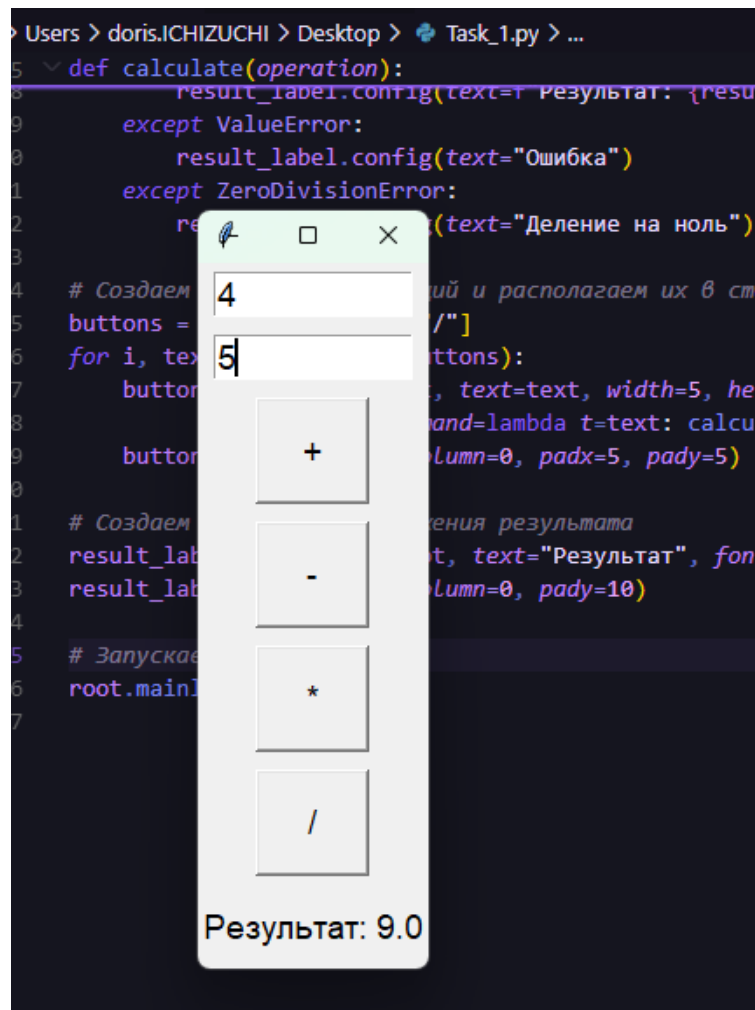


Рисунок 7. Вывод задания №1

Задание №2. 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.

```

Tasks > Task_2.py > ...
1  import tkinter as tk
2
3  colors = [
4      ("Красный", "#ff0000"),
5      ("Оранжевый", "#ff7d00"),
6      ("Желтый", "#ffff00"),
7      ("Зеленый", "#00ff00"),
8      ("Голубой", "#007dff"),
9      ("Синий", "#0000ff"),
10     ("Фиолетовый", "#7d00ff")
11 ]
12
13 root = tk.Tk()
14 root.title("Цвета радуги")
15
16 # Метка названия цвета
17 label = tk.Label(root, text="Выберите цвет", font=("Arial", 14))
18 label.pack(pady=10)
19
20 # Текстовое поле для отображения кода цвета
21 color_code_entry = tk.Entry(root, font=("Arial", 12), width=15, justify="center")
22 color_code_entry.pack(pady=10)
23
24 def show_color(name, code):
25     label.config(text=name)
26     color_code_entry.delete(0, tk.END)
27     color_code_entry.insert(0, code)
28
29 # Кнопки для каждого цвета
30 for name, code in colors:
31     button = tk.Button(
32         root, text=name, bg=code, width=10, height=2,
33         command=lambda n=name, c=code: show_color(n, c)
34     )
35     button.pack(pady=5)
36
37 # Запуск цикла
38 root.mainloop()
39

```

Рисунок 8. Код программы

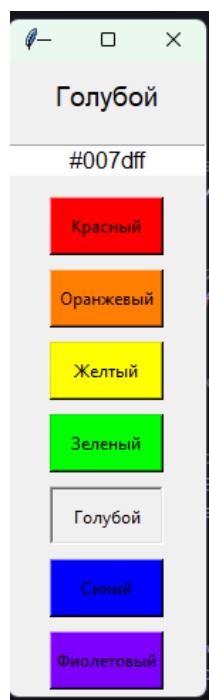


Рисунок 9. Вывод задания №2

Задание №3. Перепишите программу из задания №2 так, чтобы кнопки цветов находились стояли в ряд.

```
Tasks > Task_3.py > on_color_button_click
1  import tkinter as tk
2
3  colors = {
4      "Красный": "#ff0000",
5      "Оранжевый": "#ff7d00",
6      "Желтый": "#ffff00",
7      "Зеленый": "#00ff00",
8      "Голубой": "#007dff",
9      "Синий": "#0000ff",
10     "Фиолетовый": "#7d00ff"
11 }
12
13 def on_color_button_click(color_name, color_code):
14     color_label.config(text=color_name)
15     color_code_entry.delete(0, tk.END)
16     color_code_entry.insert(0, color_code)
17
18 root = tk.Tk()
19 root.title("Цвета радуги")
20
21 color_label = tk.Label(root, text="Название цвета", font=("Arial", 14))
22 color_label.pack(pady=10)
23
24 color_code_entry = tk.Entry(root, font=("Arial", 12), justify="center")
25 color_code_entry.pack(pady=10)
26
27 button_frame = tk.Frame(root)
28 button_frame.pack(pady=10)
29
30 for color_name, color_code in colors.items():
31     button = tk.Button(
32         button_frame,
33         bg=color_code,
34         width=6,
35         height=3,
36         command=lambda name=color_name, code=color_code: on_color_button_click(name, code)
37     )
38     button.pack(side=tk.LEFT, padx=5)
39
40 root.mainloop()
41
```

Рисунок 10. Код программы

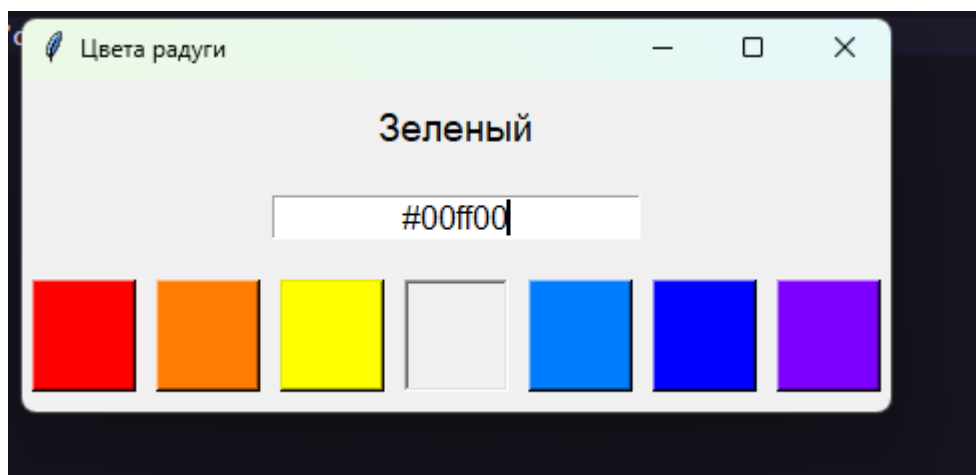


Рисунок 11. Вывод задания №3

Задание №4. 10 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.

```
Tasks > Task_4.py > save_file
1  import tkinter as tk
2  from tkinter import messagebox
3
4  # Функция для открытия файла и загрузки содержимого в Text
5  def open_file():
6      file_name = entry.get()
7      if not file_name:
8          messagebox.showwarning("Ошибка", "Введите имя файла")
9      return
10     try:
11         with open(file_name, 'r', encoding='utf-8') as file:
12             text.delete(1.0, tk.END)
13             text.insert(tk.END, file.read())
14     except FileNotFoundError:
15         messagebox.showerror("Ошибка", f"Файл '{file_name}' не найден")
16
17 # Сохранение содержимого Text в файл
18 def save_file():
19     file_name = entry.get()
20     if not file_name:
21         messagebox.showwarning("Ошибка", "Введите имя файла")
22         return
23     with open(file_name, 'w', encoding='utf-8') as file:
24         file.write(text.get(1.0, tk.END))
25     messagebox.showinfo("Успех", f"Файл '{file_name}' успешно сохранен")
26
27 root = tk.Tk()
28 root.title("Редактор файлов")
29
30 entry_label = tk.Label(root, text="Имя файла:")
31 entry_label.pack(pady=5)
32
33 entry = tk.Entry(root, width=50)
34 entry.pack(padx=10, pady=5)
35
36 text = tk.Text(root, wrap='word', width=60, height=20)
37 text.pack(padx=10, pady=5)
38
39 open_button = tk.Button(root, text="Открыть", command=open_file)
40 open_button.pack(side=tk.LEFT, padx=10, pady=5)
41
42 save_button = tk.Button(root, text="Сохранить", command=save_file)
43 save_button.pack(side=tk.RIGHT, padx=10, pady=5)
```

Рисунок 12. Код программы

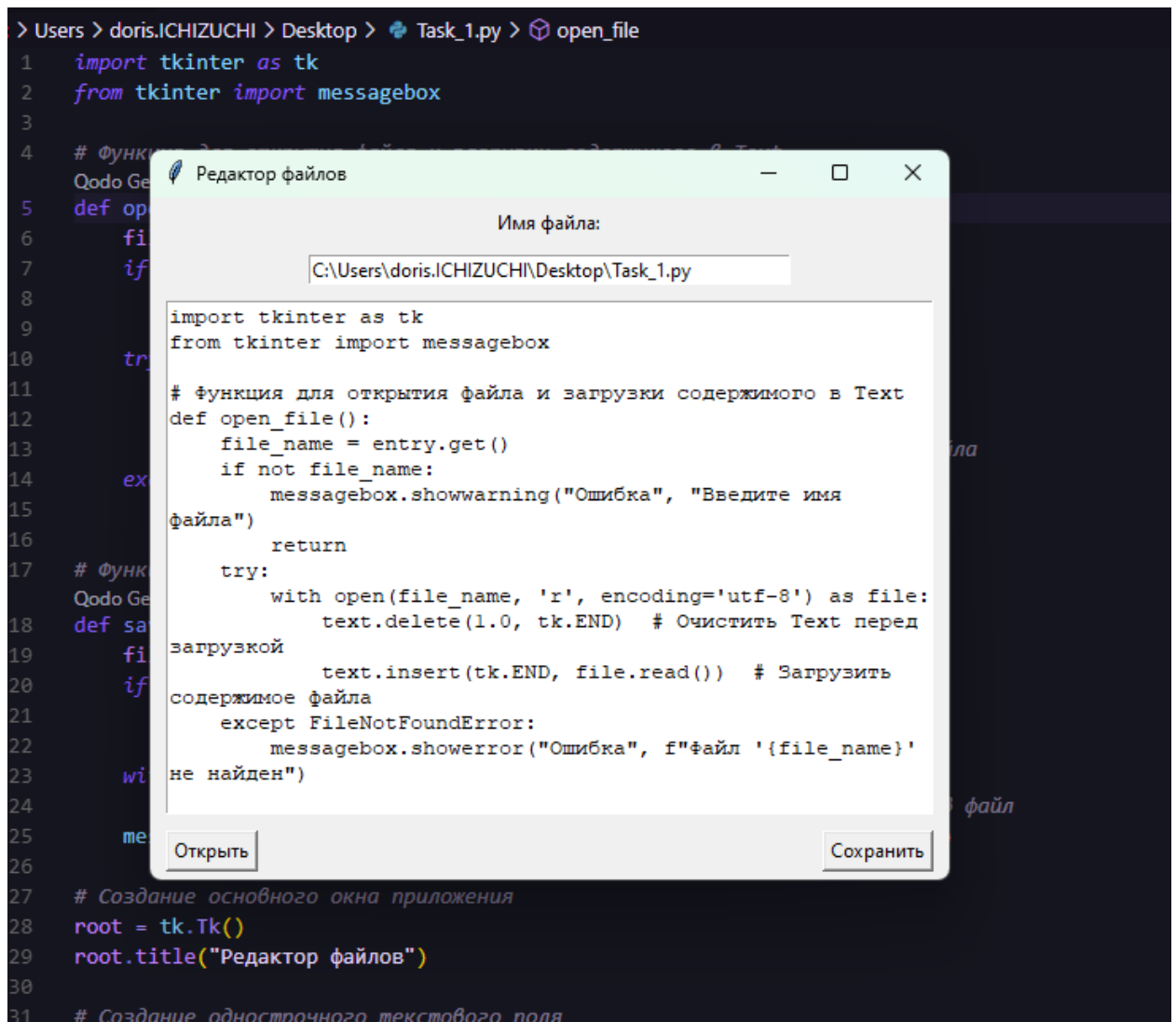


Рисунок 13. Вывод задания №4

Задание №5. 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.

```

1  import tkinter as tk
2
3  def on_radio_button_select():
4      if r_var.get() == 1:
5          info_label.config(text="1 Января")
6      elif r_var.get() == 2:
7          info_label.config(text="4 Ноября")
8      elif r_var.get() == 3:
9          info_label.config(text="9 Мая")
10
11 # главное окно
12 root = tk.Tk()
13 root.title("Радиокнопки")
14
15 radio_frame = tk.Frame(root)
16 radio_frame.pack(side=tk.LEFT, padx=20, pady=20)
17
18 info_label = tk.Label(root, text="Выберите вариант", font=("Arial", 14), width=30, anchor="w")
19 info_label.pack(side=tk.LEFT, padx=20, pady=20)
20
21 r_var = tk.IntVar()
22
23 r1 = tk.Radiobutton(radio_frame, text="Новый Год", variable=r_var, value=1, command=on_radio_button_select, indicatoron=0, width=20, height=2, anchor="w", padx=10, pady=5)
24 r1.pack(pady=5)
25
26 r2 = tk.Radiobutton(radio_frame, text="День Народного Единства", variable=r_var, value=2, command=on_radio_button_select, indicatoron=0, width=20, height=2, anchor="w", padx=10, pady=5)
27 r2.pack(pady=5)
28
29 r3 = tk.Radiobutton(radio_frame, text="День Победы", variable=r_var, value=3, command=on_radio_button_select, indicatoron=0, width=20, height=2, anchor="w", padx=10, pady=5)
30 r3.pack(pady=5)
31
32 root.mainloop()
33

```

Рисунок 14. Код программы

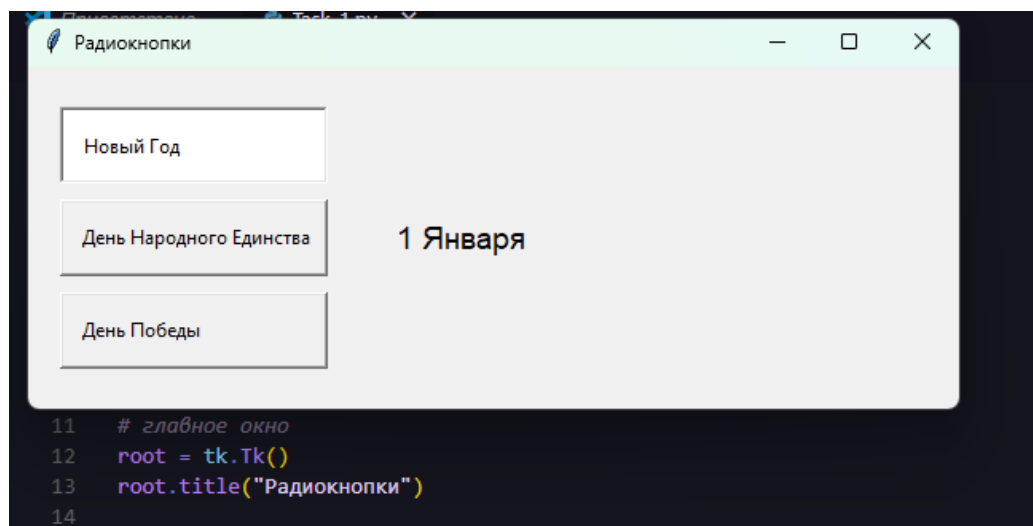


Рисунок 15. Вывод задания №5

```

==> For changes to take effect, close and re-open your current shell. <==

@Ichizuchi →/workspaces/LR_4.7 (develop) $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

@Ichizuchi →/workspaces/LR_4.7 (main) $ git merge develop
Updating b2c500e..a4c435b
Fast-forward
 Tasks/Task_1.py | 43 ++++++
 Tasks/Task_2.py | 38 ++++++
 Tasks/Task_3.py | 40 ++++++

```

Рисунок 16. Слияние веток

Ответы на контрольные вопросы

1. Каково назначение виджета «Listbox»?

«Listbox» — это виджет Tkinter для отображения списка элементов, из которого можно выбирать один или несколько элементов. Используется, когда

необходимо предоставить пользователю возможность выбора из списка, например, для выбора опций, файлов или категорий.

2. Каким образом осуществляется связывание события или действия с виджетом Tkinter?

Связывание события с виджетом осуществляется с помощью метода «`widget.bind(event, handler)`», где «event» — это строка, описывающая событие (например, `<Button-1>` для клика мыши), а handler — функция-обработчик, которая вызывается при возникновении события.

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

Основные типы событий в Tkinter включают:

- Клавиатурные события: `<KeyPress>` (нажатие клавиши), `<KeyRelease>` (отпускание клавиши)
- Мышиные события: `<Button-1>` (левая кнопка мыши), `<Button-3>` (правая кнопка мыши), `<Double-Button-1>` (двойной клик)
- События окна: `<Configure>` (изменение размера окна), `<FocusIn>` и `<FocusOut>` (получение и потеря фокуса)
- Системные события: `<Destroy>` (закрытие окна)

Пример: «`button.bind(<Button-1>, handler_function)`» связывает клик левой кнопки мыши с «button».

4. Как обрабатываются события в Tkinter?

В Tkinter события обрабатываются функциями-обработчиками, которые привязываются к виджетам через «bind». Когда событие происходит, вызовется привязанная функция-обработчик. Аргументом функции будет объект события («event»), содержащий информацию о типе события, координатах и других параметрах.

5. Как обрабатываются события мыши в Tkinter?

События мыши обрабатываются с помощью «bind», указывая тип события (например, `<Button-1>`, `<Motion>` для перемещения курсора). Объект

события передаёт данные, такие как «event.x» и «event.y» — координаты указателя мыши, что позволяет определить точное местоположение курсора при возникновении события.

6. Каким образом можно отображать графические примитивы в Tkinter?

Графические примитивы отображаются с помощью виджета «Canvas», на котором можно рисовать линии, окружности, прямоугольники, многоугольники, текст и изображения. «Canvas» предоставляет методы для создания и управления этими объектами.

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

Основные методы «Canvas» для отображения графических примитивов:

- «create_line()» — рисует линию
- «create_rectangle()» — рисует прямоугольник
- «create_oval()» — рисует эллипс или круг
- «create_polygon()» — рисует многоугольник
- «create_text()» — добавляет текст
- «create_image()» — отображает изображение

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

Для обращения к фигуре используется её идентификатор, который возвращается при создании. Фигурам также можно назначить тэги, что позволяет обращаться к ним группами. Методы, такие как «itemconfig» и «coords», позволяют изменять свойства и положение фигур по их идентификатору или тэгу.

9. Каково назначение тэгов в Tkinter?

Тэги позволяют группировать объекты на «Canvas», чтобы управлять ими вместе. Например, можно применить одинаковое действие ко всем

фигурам с заданным тэгом, изменять их цвет, позицию или удалять их из холста одним вызовом команды.

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