Студент группы ИС-25 Дудков Д.Ю.

Практическое занятие № 17

Тема: составление программ с использованием GUI Tkinter в IDE PyCharm Community, изучение возможностей модуля OS.

Цель: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ с использованием GUI Tkinter в IDE PyCharm Community, изучить возможности модуля OS.

Постановка задачи.
В соответствии с номером варианта перейти по ссылке на прототип. Реализовать его в IDE PyCharm Community с применением пакета tk. Получить интерфейс максимально приближенный к оригиналу. Вариант 8 - https://i.pinimg.com/originals/73/c6/0d/73c60def8c55043f9fd27b370530a9cf.jpg

Текст программы:

```
import tkinter as tk
from tkinter import ttk

from datetime import datetime
from pprint import pprint



def window_deleted():
    print("Окно закрыто")
    root.quit()  # явное указание на выход из программы



def cancel():
    entry_first_name.delete(0, tk.END)
    entry_first_name.config(fg='Black')
    focus_out_entry_box(entry_first_name, entry_first_name_text)

    entry_last_name.delete(0, tk.END)
    entry_last_name.config(fg='Black')
    focus_out_entry_box(entry_last_name, entry_last_name_text)

    entry_screen_name.delete(0, tk.END)
    entry_screen_name.config(fg='Black')
    focus_out_entry_box(entry_screen_name, entry_screen_name_text)

    entry_email.delete(0, tk.END)
    entry_email.config(fg='Black')
    focus_out_entry_box(entry_email, entry_email_text)
```

```
    entry_phone.delete(0, tk.END)
    entry_phone.config(fg='Black')
    focus_out_entry_box(entry_phone, entry_phone_text)

    entry_password.delete(0, tk.END)
    entry_confirm_password.delete(0, tk.END)

    default_month.set(month_list[0])
    default_day.set(day_list[0])
    default_year.set("1985")
    default_country.set(country_list[0])

    radio_button_male.config(variable=tk.StringVar())
    radio_button_female.config(variable=tk.StringVar())
    checkbutton_terms_of_use.config(variable=tk.BooleanVar(value=False))

    print("Форма очищена\n")


def submit():
    gotten_month_num = str(month_list.index(combo_box_month.get()) + 1).rjust(2, "0")
    gotten_year = combo_box_year.get()
    gotten_day = combo_box_day.get().rjust(2, "0")
    try:
        full_date = datetime.strptime(f"{gotten_year}-{gotten_month_num}-{gotten_day}",
"%Y-%m-%d")
    except ValueError:
        full_date = ""
        print("Invalid date was given!\n")

    data = {
        "first_name": entry_first_name.get(),
        "last_name": entry_last_name.get(),
        "screen_name": entry_screen_name.get(),
        "email": entry_email.get(),
        "phone": entry_phone.get(),
        "password": entry_password.get(),
        "confirm_password": entry_confirm_password.get(),
        "date_of_birth": full_date,
        "country": combo_box_country.get(),
        "gender": gender.get(),
        "terms_agree": terms_agree.get(),
    }

    pprint(data, indent=4)
    print("\n")

    cancel()


"""
----------------
```

```
ОСНОВНЫЕ ЭЛЕМЕНТЫ
-----------------
"""



root = tk.Tk()
root.title("Sign Up")
root.geometry("560x655+700+400")  # ширина=500, высота=400, x=300, y=200
root.protocol('WM_DELETE_WINDOW', window_deleted)  # обработчик закрытия окна
root.resizable(False, False)  # размер окна НЕ может быть изменён
root.config(bg="#de8704")

for i in range(10):
    root.columnconfigure(index=i, weight=1)
for i in range(23):
    root.rowconfigure(index=i, weight=1)

main_title_label = tk.Label(root, text='Sign Up', bg="#de8704", fg="#fde82d",
font=("Arial", 16))
main_frame = tk.Frame(root, bg='#222536', bd=5)
cancel_button = tk.Button(root, text='Cancel', font=("Arial", 11, "bold"),
background="red", foreground="white", command=cancel)
submit_button = tk.Button(root, text='Submit', font=("Arial", 11, "bold"),
background="green", foreground="white", command=submit)

main_title_label.place(x=10, y=4)
main_frame.grid(row=1, column=0, columnspan=11, rowspan=22, padx=0, pady=12,
sticky="nsew")
cancel_button.grid(row=23, column=9, columnspan=1, ipadx=4, ipady=4, padx=0,
pady=12)
submit_button.grid(row=23, column=10, columnspan=1, ipadx=4, ipady=4, padx=12,
pady=12)



"""
---------------
ЯРЛЫКИ ДЛЯ ПОЛЕЙ
---------------
"""



first_name_label = tk.Label(root, text='First Name', bg="#222536", fg="#fde82d",
font=("Arial", 11, "bold"))
first_name_label.grid(row=1, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

last_name_label = tk.Label(root, text='Last Name', bg="#222536", fg="#fde82d",
font=("Arial", 11, "bold"))
last_name_label.grid(row=3, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

screen_name_label = tk.Label(root, text='Screen Name', bg="#222536", fg="#fde82d",
font=("Arial", 11, "bold"))
```

```
screen_name_label.grid(row=5, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

date_of_birth_label = tk.Label(root, text='Date Of Birth', bg="#222536", fg="#fde82d",
font=("Arial", 11, "bold"))
date_of_birth_label.grid(row=7, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

gender_label = tk.Label(root, text='Gender', bg="#222536", fg="#fde82d", font=("Arial",
11, "bold"))
gender_label.grid(row=9, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

country_label = tk.Label(root, text='Country', bg="#222536", fg="#fde82d",
font=("Arial", 11, "bold"))
country_label.grid(row=11, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

email_label = tk.Label(root, text='E-mail', bg="#222536", fg="#fde82d", font=("Arial",
11, "bold"))
email_label.grid(row=13, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

phone_label = tk.Label(root, text='Phone', bg="#222536", fg="#fde82d", font=("Arial",
11, "bold"))
phone_label.grid(row=15, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

password_label = tk.Label(root, text='Password', bg="#222536", fg="#fde82d",
font=("Arial", 11, "bold"))
password_label.grid(row=17, column=0, columnspan=4, rowspan=2, padx=0, pady=0,
sticky="e")

confirm_password_label = tk.Label(root, text='Confirm Password', bg="#222536",
fg="#fde82d", font=("Arial", 11, "bold"))
confirm_password_label.grid(row=19, column=0, columnspan=4, rowspan=2, padx=0,
pady=0, sticky="e")



"""
-------------------
ПОЛЯ ДЛЯ ЗАПОЛНЕНИЯ
-------------------
"""



def focus_out_entry_box(widget, widget_text):
    if widget['fg'] == 'Black' and len(widget.get()) == 0:
        widget.delete(0, tk.END)
        widget['fg'] = 'Grey'
        widget.insert(0, widget_text)
```

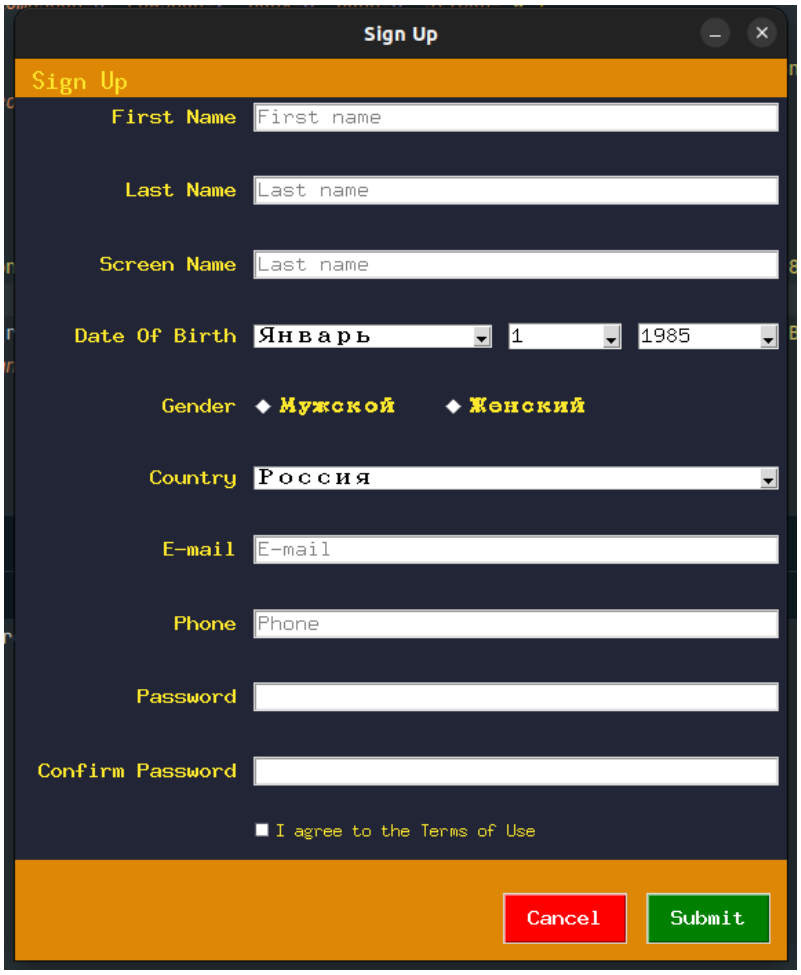
```
def focus_in_entry_box(widget):
    if widget['fg'] == 'Grey':
        widget['fg'] = 'Black'
        widget.delete(0, tk.END)



entry_first_name_text = 'First name'
entry_first_name = tk.Entry(root, font='Arial 11', fg='Grey')
entry_first_name.insert(0, entry_first_name_text)
entry_first_name.grid(row=1, column=5, columnspan=6, rowspan=2, padx=6, pady=0,
sticky="we")
entry_first_name.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_first_name))
entry_first_name.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_first_name,
entry_first_name_text))

entry_last_name_text = 'Last name'
entry_last_name = tk.Entry(root, font='Arial 11', fg='Grey')
entry_last_name.insert(0, entry_last_name_text)
entry_last_name.grid(row=3, column=5, columnspan=6, rowspan=2, padx=6, pady=0,
sticky="we")
entry_last_name.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_last_name))
entry_last_name.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_last_name,
entry_last_name_text))

entry_screen_name_text = 'Last name'
entry_screen_name = tk.Entry(root, font='Arial 11', fg='Grey')
entry_screen_name.insert(0, entry_screen_name_text)
entry_screen_name.grid(row=5, column=5, columnspan=6, rowspan=2, padx=6, pady=0,
sticky="we")
entry_screen_name.bind("<FocusIn>", lambda args:
focus_in_entry_box(entry_screen_name))
entry_screen_name.bind("<FocusOut>", lambda args:
focus_out_entry_box(entry_screen_name, entry_screen_name_text))

entry_email_text = 'E-mail'
entry_email = tk.Entry(root, font='Arial 11', fg='Grey')
entry_email.insert(0, entry_email_text)
entry_email.grid(row=13, column=5, columnspan=6, rowspan=2, padx=6, pady=0,
sticky="we")
entry_email.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_email))
entry_email.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_email,
entry_email_text))

entry_phone_text = 'Phone'
entry_phone = tk.Entry(root, font='Arial 11', fg='Grey')
entry_phone.insert(0, entry_phone_text)
entry_phone.grid(row=15, column=5, columnspan=6, rowspan=2, padx=6, pady=0,
sticky="we")
entry_phone.bind("<FocusIn>", lambda args: focus_in_entry_box(entry_phone))
entry_phone.bind("<FocusOut>", lambda args: focus_out_entry_box(entry_phone,
entry_phone_text))

entry_password = tk.Entry(root, font='Arial 11')
entry_password.grid(row=17, column=5, columnspan=6, rowspan=2, padx=6, pady=0,
```

```
sticky="we")

entry_confirm_password = tk.Entry(root, font='Arial 11')
entry_confirm_password.grid(row=19, column=5, columnspan=6, rowspan=2, padx=6,
pady=0, sticky="we")



"""
--------------
ПОЛЯ С ВЫБОРОМ
--------------
"""

month_list = ["Январь", "Февраль", "Март", "Апрель", "Май", "Июнь", "Июль", "Август",
"Сентябрь", "Октябрь", "Ноябрь", "Декабрь"]
default_month = tk.StringVar(value=month_list[0])

day_list = [str(i) for i in range(1, 32)]
default_day = tk.StringVar(value=day_list[0])

year_list = [str(i) for i in range(1900, 2025)]
default_year = tk.StringVar(value="1985")

country_list = ["Россия", "Украина", "Беларусь", "Казахстан", "Другие"]
default_country = tk.StringVar(value=country_list[0])



combo_box_month = ttk.Combobox(root, values=month_list, textvariable=default_month,
font='Arial 10 bold')
combo_box_month.grid(row=7, column=5, columnspan=4, rowspan=2, padx=6, pady=0,
sticky="we")

combo_box_day = ttk.Combobox(root, values=day_list, textvariable=default_day,
font='Arial 11', width=5)
combo_box_day.grid(row=7, column=9, columnspan=1, rowspan=2, padx=6, pady=0,
sticky="we")

combo_box_year = ttk.Combobox(root, values=year_list, textvariable=default_year,
font='Arial 11', width=5)
combo_box_year.grid(row=7, column=10, columnspan=2, rowspan=2, padx=6, pady=0,
sticky="we")

combo_box_country = ttk.Combobox(root, values=country_list,
textvariable=default_country, font='Arial 10 bold')
combo_box_country.grid(row=11, column=5, columnspan=6, rowspan=2, padx=6,
pady=0, sticky="we")



"""
---------------
РАДИО И ГАЛОЧКИ
---------------
```

```
"""


male = "Мужской"
female = "Женский"
gender = tk.StringVar()
gender_style = ttk.Style()
gender_style.configure("BW.TRadiobutton", font=("Arial", 9), background="#222536",
border=0, foreground="#fde82d")

radio_button_male = ttk.Radiobutton(root, text="Мужской", value="male",
variable=gender, style="BW.TRadiobutton")
radio_button_male.grid(row=9, column=5, columnspan=3, rowspan=2, padx=6, pady=0,
sticky="w")

radio_button_female = ttk.Radiobutton(root, text="Женский", value="female",
variable=gender, style="BW.TRadiobutton")
radio_button_female.grid(row=9, column=8, columnspan=3, rowspan=2, padx=6,
pady=0, sticky="w")



terms_agree = tk.BooleanVar(value=False)
terms_style = ttk.Style()
terms_style.configure("BW.TCheckbutton", font=("Arial", 9), background="#222536",
border=0, foreground="#fde82d")

checkbutton_terms_of_use = ttk.Checkbutton(root, text="I agree to the Terms of Use",
variable=terms_agree, style="BW.TCheckbutton")
checkbutton_terms_of_use.grid(row=21, column=5, columnspan=6, rowspan=1, padx=6,
pady=0, sticky="w")

root.mainloop()
```

Протокол работы программы:

Студент группы ИС-25 Дудков Д.Ю.



{ 'confirm_password': 'password',
'country': 'Россия',
'date_of_birth': datetime.datetime(2011, 4, 5, 0, 0),
'email': 'vasya@mail.ru',
'first_name': 'Вася',
'gender': 'male',
'last_name': 'Хороший',
'password': 'password',
'phone': '89008007060',
'screen_name': 'Вася',
'terms_agree': True}

Форма очищена

Окно закрыто

Process finished with exit code 0

Студент группы ИС-25 Дудков Д.Ю.

Постановка задачи.

Разработать программу с применением пакета tk, взяв в качестве условия одну любую задачу из ПЗ №№ 2 – 9. Выбранная задача: ПЗ 7 задание 1.

Вывести строку длины N, которая состоит из символов C.

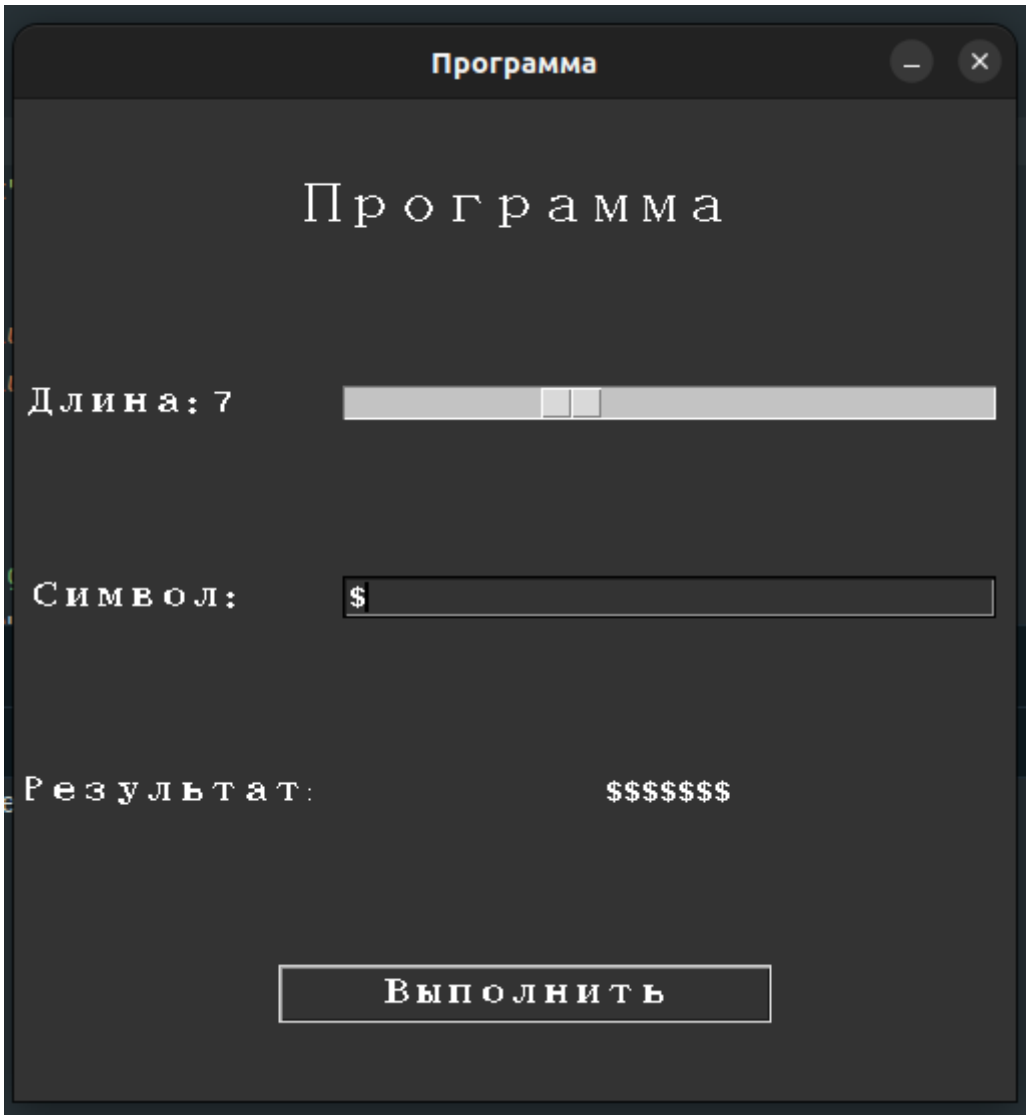
Текст программы:

```
import tkinter as tk
from tkinter import ttk



def design(root: tk.Tk):
    def change(new_value):
        amount_label["text"] = str(round(float(new_value)))

    def check_char(new_value):
        return len(new_value) <= 1

    def calculate():
        char = char_entry.get()
        amount = round(amount_scale.get())

        if char and amount:
            result_value_label["text"] = char * amount

    root.title("Программа")
    root.geometry("500x500+700+400")
    root.resizable(False, False)
    root.config(bg="#333333")

    for i in range(10):
        root.columnconfigure(index=i, weight=1)
    for i in range(5):
        root.rowconfigure(index=i, weight=1)

    label_config = {
        "bg": "#333333",
        "fg": "white",
        "font": ("extra", 11, "bold"),
    }

    main_title_label = tk.Label(root, text='Программа', bg="#333333", fg="white",
font=("extra", 18))
    main_title_label.grid(row=0, column=0, columnspan=10, rowspan=1, padx=0, pady=0,
sticky="we")

    amount_title_label = tk.Label(root, text="Длина:", **label_config)
    amount_title_label.grid(row=1, column=0, columnspan=2, rowspan=1, padx=6,
pady=0, sticky="w")
```

```
    default_amount = 10

    amount_label = tk.Label(root, text=str(default_amount), **label_config)
    amount_label.grid(row=1, column=2, columnspan=2, rowspan=1, padx=0, pady=0,
sticky="w")

    amount_value = tk.IntVar(root, value=default_amount)
    amount_scale = ttk.Scale(root, orient=tk.HORIZONTAL, length=350, from_=1.0,
to=20.0, variable=amount_value, command=change)
    amount_scale.grid(row=1, column=4, columnspan=6, rowspan=1, padx=10, pady=0,
sticky="e")

    char_title_label = tk.Label(root, text="Символ:", **label_config)
    char_title_label.grid(row=2, column=0, columnspan=4, rowspan=1, padx=6, pady=0,
sticky="w")

    char_validator = (root.register(check_char), "%P")

    char_entry = tk.Entry(root, **label_config, width=38, validate="key",
validatecommand=char_validator)
    char_entry.grid(row=2, column=4, columnspan=6, rowspan=1, padx=10, pady=0,
sticky="e")

    result_title_label = tk.Label(root, text="Результат:", **label_config)
    result_title_label.grid(row=3, column=0, columnspan=4, rowspan=1, padx=6, pady=0,
sticky="w")

    result_value_label = tk.Label(root, **label_config)
    result_value_label.grid(row=3, column=4, columnspan=6, rowspan=1, padx=6,
pady=0, sticky="we")

    submit_button = tk.Button(root, text='Выполнить', **label_config, command=calculate)
    submit_button.grid(row=4, column=3, columnspan=5, rowspan=1, padx=6, pady=0,
sticky="we")



if __name__ == '__main__':
    window = tk.Tk()
    design(root=window)
    window.mainloop()
```

Протокол работы программы:

Студент группы ИС-25 Дудков Д.Ю.



Process finished with exit code 0

Студент группы ИС-25 Дудков Д.Ю.

Постановка задачи.
Все задания выполняются с использованием модуля OS:

- перейдите в каталог PZ11. Выведите список всех файлов в этом каталоге. Имена вложенных подкаталогов выводить не нужно.
- перейти в корень проекта, создать папку с именем test. В ней создать еще одну папку test1. В папку test переместить два файла из П36, а в папку test1 - один файл из П37. Файл из П37 переименовать в test.txt. Вывести в консоль информацию о размере файлов в папке test.
- перейти в папку с PZ11, найти там файл с самым коротким именем, имя вывести в консоль. Использовать функцию basename () (os.path.basename()).
- перейти в любую папку где есть отчет в формате .pdf и «запустите» файл в привязанной к нему программе. Использовать функцию os.startfile().
- удалить файл test.txt.

Текст программы:

```
import os
from time import sleep



def files_list(directory: str) -> None:
    for file in os.listdir(directory):
        print(file)
    print()



def first():
    print("Current path:", os.getcwd())
    os.chdir("/home/danil/Documents/RKSI/Base_of_programming/Proj_1sem_Dudkov/
PZ_11_8")
    print("Path after \"cd\":", os.getcwd())

    print("List of files:\n")
    files_list(directory=".")



def second(sleep_time: int):
    os.chdir("/home/danil/Documents/RKSI/Base_of_programming/Proj_1sem_Dudkov")
    print("Current path:", os.getcwd())

    test = "./test"
```

```
    test1 = "./test/test1"

    if not os.path.isdir(test):
        os.mkdir(test)
    if not os.path.isdir(test1):
        os.mkdir(test1)

    os.replace("./PZ_6_8/PZ_6_8_1.py", f"{test}/PZ_6_8_1.py")
    os.replace("./PZ_6_8/PZ_6_8_2.py", f"{test}/PZ_6_8_2.py")
    os.replace("./PZ_7_8/PZ_7_8_1.py", f"{test1}/test.txt")

    print(f"List of files in {test}:\n")
    files_list(directory=test)

    print(f"List of files in {test1}:\n")
    files_list(directory=test1)

    print(f"Info about files in {test}:\n")
    for file in os.listdir(test):
        # проверка на то, файл или папка
        if os.path.isfile(f'{test}/{file}'):
            print(f"{test}/{file}: {os.stat(f'{test}/{file}').st_size} bytes")

    sleep(sleep_time)
    os.replace(f"{test}/PZ_6_8_1.py", "./PZ_6_8/PZ_6_8_1.py")
    os.replace(f"{test}/PZ_6_8_2.py", "./PZ_6_8/PZ_6_8_2.py")
    os.replace(f"{test1}/test.txt", "./PZ_7_8/PZ_7_8_1.py")



def third():
    os.chdir("/home/danil/Documents/RKSI/Base_of_programming/Proj_1sem_Dudkov/
PZ_11_8")
    print(f"Current path: {os.getcwd()}\n")

    print(f"List of files in .:\n")
    files_list(".")

    min_filename = min(os.listdir("."), key=lambda filename:
len(os.path.basename(filename)))
    print(f"Filename with min length: {min_filename}")



def fourth():
    os.chdir("/home/danil/Documents/RKSI/Base_of_programming/Proj_1sem_Dudkov/
Reports")
    print(f"Current path: {os.getcwd()}\n")

    os.system("evince ./PZ_7_8.pdf")



def fifth():
    os.chdir("/home/danil/Documents/RKSI/Base_of_programming/Proj_1sem_Dudkov/test")
```

```
    print(f"Current path: {os.getcwd()}\n")

    os.system("touch ./test.txt")

    print(f"List of files in .:\n")
    files_list(".")

    os.remove("./test.txt")

    print(f"List of files after removing:\n")
    files_list(".")


if __name__ == "__main__":
    first()
    # second(sleep_time=2)
    # third()
    # fourth()
    # fifth()
```

Протокол работы программы:

Current path:
/home/danil/Documents/RKSI/Base_of_programming/Proj_1sem_Dudkov/PZ_17_8
Path after "cd":
/home/danil/Documents/RKSI/Base_of_programming/Proj_1sem_Dudkov/PZ_11_8
List of files:

report.txt
text18-8.txt
__init__.py
nums.txt
PZ_11_8_2.py
PZ_11_8_1.py
edited_poetry.txt

Process finished with exit code 0

Студент группы ИС-25 Дудков Д.Ю.

Вывод: в процессе выполнения практического занятия закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрёл навыки составления программ с использованием GUI Tkinter в IDE PyCharm Community, изучитл возможности модуля OS. Были использованы языковые конструкции def, if, for, try.
Выполнены разработка кода, отладка, тестирование, оптимизация программного кода.
Готовые программные коды выложены на GitHub.