Студент группы ис-27 Слюсарев Е. Р.

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 приобрести навыки составления программ с использованием GUI Tkinter в IDE PyCharm Community, изучить возможности модуля OS.

Задачи

№1

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

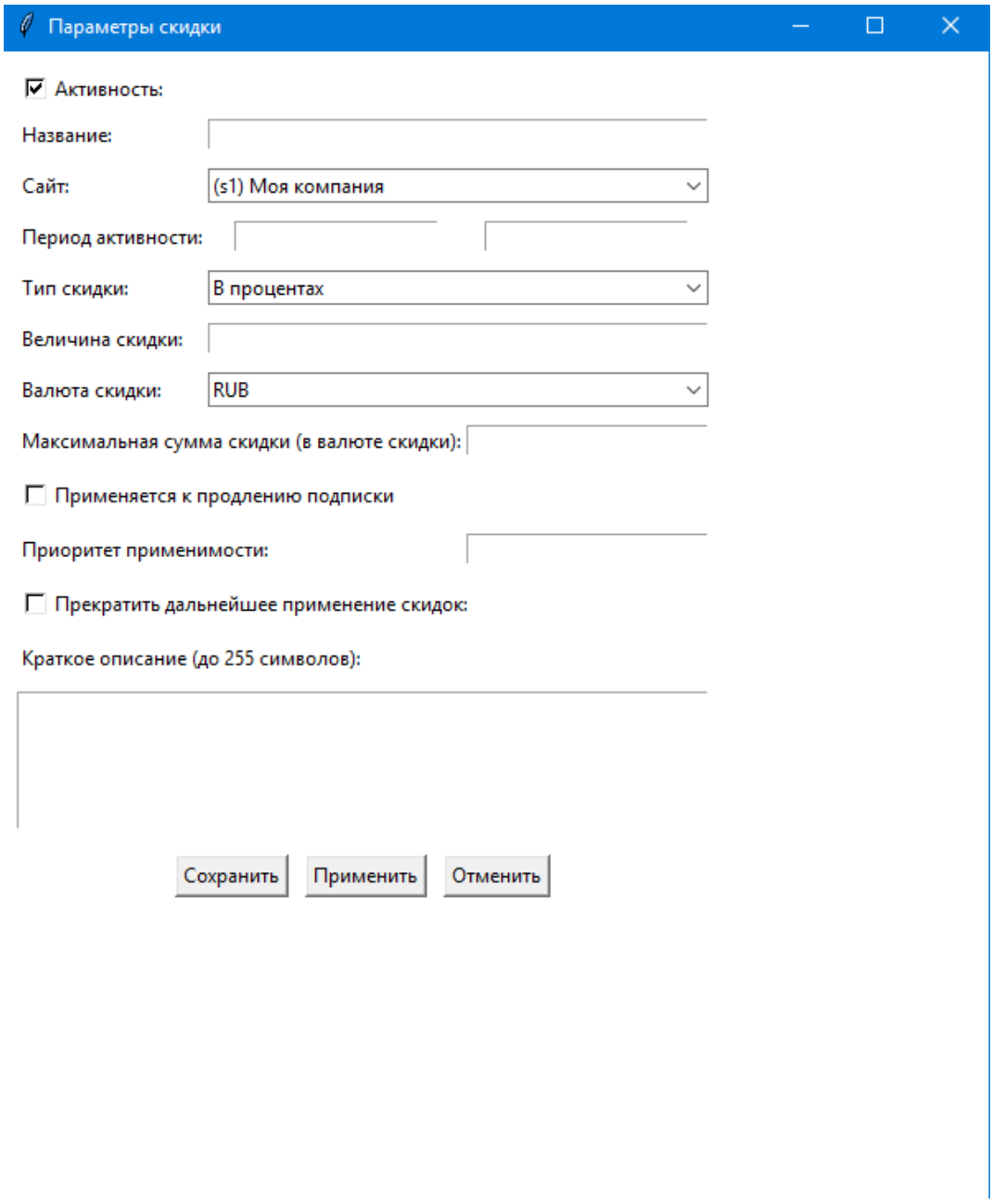
Текст программы:

```
# 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

import tkinter as tk
from tkinter import ttk

# Create the main window
root = tk.Tk()
root.title("Параметры скидки")
root.geometry("600x700")
root.configure(bg="white")

# Create the main frame
main_frame = tk.Frame(root, bg="white", padx=10, pady=10)
main_frame.grid(row=0, column=0, sticky=(tk.W, tk.E, tk.N, tk.S))

# Activate Checkbox
active_var = tk.BooleanVar(value=True)
active_check = tk.Checkbutton(main_frame, text="Активность:",
variable=active_var, bg="white")
active_check.grid(row=0, column=0, columnspan=2, sticky=tk.W)

# Name Label and Entry
name_label = tk.Label(main_frame, text="Название:", bg="white")
name_label.grid(row=1, column=0, sticky=tk.W, pady=5)
name_entry = tk.Entry(main_frame, width=50)
name_entry.grid(row=1, column=1, columnspan=2, sticky=(tk.W, tk.E), pady=5)

# Site Label and Combobox
site_label = tk.Label(main_frame, text="Сайт:", bg="white")
site_label.grid(row=2, column=0, sticky=tk.W, pady=5)
site_combobox = ttk.Combobox(main_frame, values=["(s1) Моя компания"],
state="readonly")
site_combobox.current(0)
site_combobox.grid(row=2, column=1, columnspan=2, sticky=(tk.W, tk.E),
pady=5)

# Period of Activity Label and Entries
```

```
period_label = tk.Label(main_frame, text="Период активности:", bg="white")
period_label.grid(row=3, column=0, sticky=tk.W, pady=5)
period_start_entry = tk.Entry(main_frame)
period_start_entry.grid(row=3, column=1, pady=5)
period_end_entry = tk.Entry(main_frame)
period_end_entry.grid(row=3, column=2, pady=5)

# Type of Discount Label and Combobox
discount_type_label = tk.Label(main_frame, text="Тип скидки:", bg="white")
discount_type_label.grid(row=4, column=0, sticky=tk.W, pady=5)
discount_type_combobox = ttk.Combobox(main_frame, values=["В процентах"],
state="readonly")
discount_type_combobox.current(0)
discount_type_combobox.grid(row=4, column=1, columnspan=2, sticky=(tk.W,
tk.E), pady=5)

# Amount of Discount Label and Entry
discount_amount_label = tk.Label(main_frame, text="Величина скидки:",
bg="white")
discount_amount_label.grid(row=5, column=0, sticky=tk.W, pady=5)
discount_amount_entry = tk.Entry(main_frame)
discount_amount_entry.grid(row=5, column=1, columnspan=2, sticky=(tk.W,
tk.E), pady=5)

# Currency Label and Combobox
currency_label = tk.Label(main_frame, text="Валюта скидки:", bg="white")
currency_label.grid(row=6, column=0, sticky=tk.W, pady=5)
currency_combobox = ttk.Combobox(main_frame, values=["RUB"],
state="readonly")
currency_combobox.current(0)
currency_combobox.grid(row=6, column=1, columnspan=2, sticky=(tk.W, tk.E),
pady=5)

# Maximum Discount Amount Label and Entry
max_discount_label = tk.Label(main_frame, text="Максимальная сумма скидки (в
валюте скидки):", bg="white")
max_discount_label.grid(row=7, column=0, columnspan=2, sticky=tk.W, pady=5)
max_discount_entry = tk.Entry(main_frame)
max_discount_entry.grid(row=7, column=2, sticky=(tk.W, tk.E), pady=5)

# Apply to Subscription Checkbox
apply_subscription_var = tk.BooleanVar()
apply_subscription_check = tk.Checkbutton(main_frame, text="Применяется к
продлению подписки", variable=apply_subscription_var, bg="white")
apply_subscription_check.grid(row=8, column=0, columnspan=3, sticky=tk.W,
pady=5)

# Priority Label and Entry
priority_label = tk.Label(main_frame, text="Приоритет применимости:",
bg="white")
priority_label.grid(row=9, column=0, columnspan=2, sticky=tk.W, pady=5)
priority_entry = tk.Entry(main_frame)
priority_entry.grid(row=9, column=2, sticky=(tk.W, tk.E), pady=5)

# Stop Further Discounts Checkbox
stop_discounts_var = tk.BooleanVar()
stop_discounts_check = tk.Checkbutton(main_frame, text="Прекратить дальнейшее
применение скидок:", variable=stop_discounts_var, bg="white")
stop_discounts_check.grid(row=10, column=0, columnspan=3, sticky=tk.W,
pady=5)

# Short Description Label and Entry
short_desc_label = tk.Label(main_frame, text="Краткое описание (до 255
символов):", bg="white")
```

```
short_desc_label.grid(row=11, column=0, columnspan=3, sticky=tk.W, pady=5)
short_desc_entry = tk.Text(main_frame, height=5, width=50)
short_desc_entry.grid(row=12, column=0, columnspan=3, sticky=(tk.W, tk.E),
pady=5)

# Buttons
button_frame = tk.Frame(main_frame, bg="white")
button_frame.grid(row=13, column=0, columnspan=3, pady=10)
save_button = tk.Button(button_frame, text="Сохранить")
save_button.grid(row=0, column=0, padx=5)
apply_button = tk.Button(button_frame, text="Применить")
apply_button.grid(row=0, column=1, padx=5)
cancel_button = tk.Button(button_frame, text="Отменить")
cancel_button.grid(row=0, column=2, padx=5)

# Run the application
root.mainloop()
```

Протаол работы:

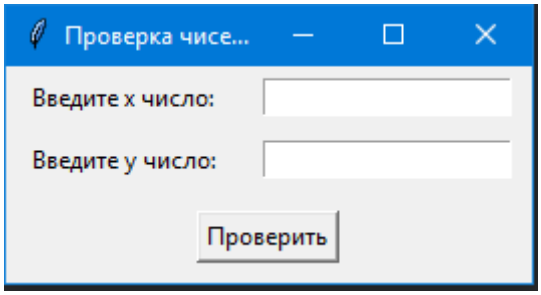


№2

Разработать программу с применением пакета tk, взяв в качестве условия задачу 3_1

```
# Разработать программу с применением пакета tk, взяв в качестве условия
# задачу 3_1

import tkinter as tk
from tkinter import messagebox


def check_values():
    x = int(entry_x.get())
    y = int(entry_y.get())

    if x > 0 and y < 0:  # проверка  в какой четверти находится координата
        messagebox.showinfo("Результат", "True")
    else:
        messagebox.showinfo("Результат", "False")


# Создание основного окна
window = tk.Tk()
window.title("Проверка чисел на противоположность")

# Метки и поля для ввода чисел
tk.Label(window, text="Введите x число:").grid(row=0, column=0, padx=10,
pady=5)
entry_x = tk.Entry(window)
entry_x.grid(row=0, column=1, padx=10, pady=5)

tk.Label(window, text="Введите y число:").grid(row=1, column=0, padx=10,
pady=5)
entry_y = tk.Entry(window)
entry_y.grid(row=1, column=1, padx=10, pady=5)


# Кнопка для проверки значений
check_button = tk.Button(window, text="Проверить", command=check_values)
check_button.grid(row=3, column=0, columnspan=2, pady=10)

window.mainloop()
```

Протакол работы:



№3_1

перейдите в каталог PZ11. Выведите список всех файлов в этом каталоге. Имена

вложенных подкаталогов выводить не нужно.

Код программы:

```
# перейдите в каталог PZ11. Выведите список всех файлов в этом каталоге.
Имена
```

```
# вложенных подкаталогов выводить не нужно.

import os

# Путь к целевому каталогу
target_directory = os.path.join('C:\Projects_1sem_Slusarev\pz 11')

# Проверяем существование каталога
if os.path.exists(target_directory) and os.path.isdir(target_directory):
    # Получаем список всех файлов в целевом каталоге
    files = [f for f in os.listdir(target_directory) if
os.path.isfile(os.path.join(target_directory, f))]

    # Выводим список файлов
    for file in files:
        print(file)
else:
    print(f"The directory '{target_directory}' does not exist.")
```

Протакол работы :

```
1.py
2.py
F1.txt
F2.txt
F_main.txt
new_file.txt
text18-26.txt
```

№3_2

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

Код программы:

```
# 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
import os
import shutil
import random

# Определение путей
project_root = os.path.abspath(os.path.dirname(__file__))
test_folder = os.path.join(project_root, 'test')
test1_folder = os.path.join(test_folder, 'test1')

# Создание папок test и test/test1
os.makedirs(test1_folder, exist_ok=True)
```

```
# Определение путей к каталогам ПЗ6 и ПЗ7
pz6_directory = os.path.join('C:\Projects_1sem_Slusarev\pz 6')
pz7_directory = os.path.join('C:\Projects_1sem_Slusarev\pz 7')

# Проверка существования каталогов ПЗ6 и ПЗ7
if not os.path.exists(pz6_directory):
    print(f"The directory '{pz6_directory}' does not exist.")
    exit(1)
if not os.path.exists(pz7_directory):
    print(f"The directory '{pz7_directory}' does not exist.")
    exit(1)

# Получение списка всех файлов в каталогах ПЗ6 и ПЗ7
pz6_files = []
for root, dirs, files in os.walk(pz6_directory):
    for f in files:
        pz6_files.append(os.path.join(root, f))

pz7_files = [os.path.join(pz7_directory, f) for f in
os.listdir(pz7_directory) if os.path.isfile(os.path.join(pz7_directory, f))]

# Проверка наличия достаточного количества файлов для перемещения
if len(pz6_files) < 2:
    print(f"Not enough files in '{pz6_directory}' to move.")
    exit(1)
if len(pz7_files) < 1:
    print(f"No files found in '{pz7_directory}' to move.")
    exit(1)

# Выбор двух случайных файлов из ПЗ6
random_pz6_files = random.sample(pz6_files, 2)

# Выбор одного случайного файла из ПЗ7
random_pz7_file = random.choice(pz7_files)

# Перемещение выбранных файлов из ПЗ6 в папку test
for file_path in random_pz6_files:
    shutil.move(file_path, test_folder)

# Перемещение и переименование выбранного файла из ПЗ7 в папку test/test1
shutil.move(random_pz7_file, os.path.join(test1_folder,'test.txt'))



# Получение списка файлов в папке test
test_files = [f for f in os.listdir(test_folder) if
os.path.isfile(os.path.join(test_folder, f))]

# Вывод информации о размере файлов в папке test
print("File sizes in the 'test' folder:")
for file_name in test_files:
    file_path = os.path.join(test_folder, file_name)
    file_size = os.path.getsize(file_path)
    print(f"{file_name}: {file_size} bytes")
```

№3_3

перейти в папку с PZ11, найти там файл с самым коротким именем, имя вывести в консоль. Использовать функцию basename () (os.path.basename()).

Код программы:

```
# перейти в папку с PZ11, найти там файл с самым коротким именем, имя вывести
в
```

```
# консоль. Использовать функцию basename () (os.path.basename()).
import os

# Путь к папке PZ11
pz11_directory = os.path.join('C:\Projects_1sem_Slusarev\pz 11')

# Проверка существования папки PZ11
if not os.path.exists(pz11_directory):
    print(f"The directory '{pz11_directory}' does not exist.")
    exit(1)

# Получение списка файлов в папке PZ11
files_in_pz11 = os.listdir(pz11_directory)

py_files = [f for f in os.listdir(pz11_directory) if f.endswith('.py')]

# Инициализация переменной для хранения имени файла с самым коротким именем
shortest_filename = None
shortest_length = None

# Поиск файла с самым коротким именем
for filename in py_files:
    file_path = os.path.join(pz11_directory, filename)
    file_length = len(filename)  # Длина имени файла
    if shortest_length is None or file_length < shortest_length:
        shortest_filename = filename
        shortest_length = file_length

# Вывод имени файла с самым коротким именем, используя os.path.basename()
if shortest_filename:
    shortest_basename = os.path.basename(shortest_filename)
    print("Shortest filename in PZ11:", shortest_basename)
else:
    print("No .py files found in PZ11 directory.")
```

Протакол работы:

```
Shortest filename in PZ11: 1.py
```

№3_4

перейти в любую папку где есть отчет в формате .pdf и «запустите» файл в привязанной к нему программе. Использовать функцию os.startfile().

Код программы:

```
# перейти в любую папку где есть отчет в формате .pdf и «запустите» файл в
# привязанной к нему программе. Использовать функцию os.startfile().

import os
import random

# Путь к папке pz на рабочем столе в OneDrive
home_directory = os.path.join('C:\Projects_1sem_Slusarev')

# Список всех возможных папок pz* от pz1 до pz17
pz_folders = [f'pz {i}' for i in range(1, 18)]

# Выбор случайной папки из списка pz*
random_pz_folder = random.choice(pz_folders)

# Путь к случайной папке pz*
random_pz_directory = os.path.join(home_directory, random_pz_folder)
```

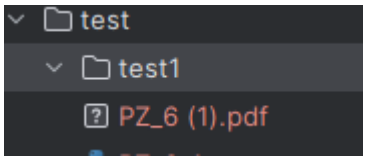
```
# Путь к папке report в случайной папке pz*
report_directory = os.path.join(random_pz_directory, 'report')

# Проверка существования папки report
if not os.path.exists(report_directory):
    print(f"The directory '{report_directory}' does not exist.")
    exit(1)

# Поиск PDF-файла в папке report
pdf_files = [os.path.join(report_directory, f) for f in
os.listdir(report_directory) if f.endswith('.pdf')]

# Выбор случайного PDF-файла из найденных
if pdf_files:
    random_pdf_file = random.choice(pdf_files)
    try:
        os.startfile(random_pdf_file)
    except OSError as e:
        print(f"Error opening file '{random_pdf_file}': {e}")
else:
    print(f"No PDF files found in the 'report' directory of
'{random_pz_folder}'.")
```

№ 3_5

удалить файл test.txt.

```
# удалить файл test.txt.

import os

# Определение пути к файлу test.txt
file_path = os.path.join('test', 'test1', 'test.txt')

# Проверка существования файла
if os.path.exists(file_path):
    try:
        os.remove(file_path)
        print(f"File '{file_path}' successfully deleted.")
    except OSError as e:
        print(f"Error deleting file '{file_path}': {e}")
else:
    print(f"File '{file_path}' does not exist.")
```

Протакол работы:



В процессе выполнения задания выработал основные принципы составления программ, приобрел навыки составления программ с использованием GUI Tkinter в IDE PyCharm Community, изучение возможностей модуля OS.

Готовые решения выложенны на Github