Студент группы ИС-27 Клипань Аким

Практическое задание №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: 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 (см. таблицу 1).
2. Разработать программу с применением пакета tk, взяв в качестве условия одну любую задачу из ПЗ №№ 2 – 9.
3. Задание предполагает, что у студента есть проект с практическими работами (№№ 2-13), оформленный согласно требованиям. 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.

Тип алгоритма: циклический

Текст программы:
1)

```
import tkinter as tk
from tkinter import ttk

def submit():
    # Логика для обработки данных при нажатии кнопки Submit
    pass

def cancel():
    # Логика для обработки данных при нажатии кнопки Cancel
    window.quit()

window = tk.Tk()
window.title("Sign Up")
window.resizable(False, False)  # Запрет на изменение размера окна

# Настройка стилей
```

```
style = ttk.Style()
style.configure('TButton', padding=6, relief="flat",
background="#f0ad4e")
style.configure('TLabel', background='#2c3e50', foreground='white',
font=('Helvetica', 10))
style.configure('TEntry', padding=6, relief="flat")

# Фрейм для размещения всех виджетов
frame = tk.Frame(window, bg='#2c3e50')
frame.pack(padx=10, pady=(10, 0), fill='x')

# Заголовок
title = tk.Label(frame, text="Sign Up", bg='#f0ad4e',
font=('Helvetica', 16))
title.grid(row=0, column=0, columnspan=2, pady=10, sticky='ew')

# Поля для ввода данных
labels = ['First Name', 'Last Name', 'Screen Name', 'Date of Birth',
'Gender', 'Country', 'E-mail', 'Phone', 'Password', 'Confirm
Password']
entries = {}

for i, label_text in enumerate(labels):
   label = ttk.Label(frame, text=label_text)
   label.grid(row=i+1, column=0, padx=5, pady=5, sticky='e')
   if label_text == 'Date of Birth':
       dob_frame = tk.Frame(frame, bg='#2c3e50')
       dob_frame.grid(row=i+1, column=1, pady=5, sticky='w')
       month = ttk.Combobox(dob_frame, values=["January",
"February", "March", "April", "May", "June", "July", "August",
"September", "October", "November", "December"], width=8)
       month.set("May")
       month.pack(side='left')
       day = ttk.Combobox(dob_frame, values=list(range(1, 32)),
width=3)
       day.set(5)
       day.pack(side='left')
       year = ttk.Combobox(dob_frame, values=list(range(1900,
2024)), width=5)
       year.set(1985)
       year.pack(side='left')
       entries[label_text] = (month, day, year)
   elif label_text == 'Gender':
       gender_frame = tk.Frame(frame, bg='#2c3e50')
       gender_frame.grid(row=i+1, column=1, pady=5, sticky='w')
       gender = tk.StringVar()
```

```
        male_rb = tk.Radiobutton(gender_frame, text="Male",
variable=gender, value="Male", bg='#2c3e50', foreground='white',
selectcolor='#2c3e50')
        male_rb.pack(side='left')
        female_rb = tk.Radiobutton(gender_frame, text="Female",
variable=gender, value="Female", bg='#2c3e50', foreground='white',
selectcolor='#2c3e50')
        female_rb.pack(side='left')
        gender.set("Male")
        entries[label_text] = gender
    elif label_text == 'Country':
        country = ttk.Combobox(frame, values=["USA", "Canada", "UK",
"Australia", "Other"], width=17)
        country.set("USA")
        country.grid(row=i+1, column=1, pady=5, sticky='w')
        entries[label_text] = country
    else:
        entry = ttk.Entry(frame, show='*' if 'Password' in label_text
else '')
        entry.grid(row=i+1, column=1, padx=5, pady=5, sticky='ew')
        entries[label_text] = entry

# Checkbox
terms = tk.Checkbutton(frame, text="I agree to the Terms of Use",
bg='#2c3e50', foreground='white', selectcolor='#2c3e50')
terms.grid(row=len(labels)+1, column=0, columnspan=2, pady=5,
sticky='w')

# Кнопки
button_frame = tk.Frame(frame, bg='#2c3e50')
button_frame.grid(row=len(labels)+2, column=0, columnspan=2,
pady=10)
submit_button = ttk.Button(button_frame, text="Submit",
command=submit)
submit_button.pack(side='left', padx=5)
cancel_button = ttk.Button(button_frame, text="Cancel",
command=cancel)
cancel_button.pack(side='left', padx=5)

window.mainloop()
```

2)

```
import tkinter as tk
from tkinter import messagebox


def check_values():
```

```
    try:
        a = int(entry_a.get())
        b = int(entry_b.get())
        c = int(entry_c.get())

        if a == -b or a == -c or b == -c:
            messagebox.showinfo("Результат", "True")
        else:
            messagebox.showinfo("Результат", "False")
    except ValueError:
        messagebox.showerror("Ошибка", "Введите валидные числа")


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

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

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

tk.Label(window, text="Введите третье число:").grid(row=2, column=0,
padx=10, pady=5)
entry_c = tk.Entry(window)
entry_c.grid(row=2, 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()
```

3)

```
import os

# Получаем домашний каталог текущего пользователя
home_directory = os.path.expanduser('~')

# Путь к целевому каталогу
target_directory =
os.path.join('/home/student/Документы/uc27klipan/pz', '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.")
```

3.2)

```
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)

home_directory = os.path.expanduser('~')

# Определение путей к каталогам ПЗ6 и ПЗ7
pz6_directory =
os.path.join('/home/student/Документы/uc27klipan/pz', 'pz 6')
pz7_directory =
os.path.join('/home/student/Документы/uc27klipan/pz', '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))

# 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)

```
import os

# Путь к корневой папке, где находятся pz1 до pz17
home_directory = os.path.expanduser('~')

# Путь к папке PZ11
pz11_directory =
os.path.join('/home/student/Документы/uc27klipan/pz', '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.")
```

3.4)

```
import os
import random
import subprocess
import sys

# Путь к корневой папке, где находятся pz1 до pz17
home_directory = os.path.expanduser('~')

# Путь к папке pz на рабочем столе в OneDrive
home_directory =
os.path.join('/home/student/Документы/uc27klipan/pz')

# 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:
        if os.name == 'posix':
            # macOS
            if sys.platform == 'darwin':
                subprocess.call(('open', random_pdf_file))
            # Linux
            else:
                subprocess.call(('xdg-open', random_pdf_file))
        elif os.name == 'nt':
            os.startfile(random_pdf_file)
    except Exception 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)

```
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.")
```

Протокол работы программы:
1) /usr/bin/python3.9 /home/student/Документы/uc27klipan/pz/pz 17/17.py

Process finished with exit code 0

2) /usr/bin/python3.9 /home/student/Документы/uc27klipan/pz/pz 17/17.2.py

Process finished with exit code 0

3)/usr/bin/python3.9 /home/student/Документы/uc27klipan/pz/pz 17/17.3.py
11.1.py
Text.txt
LongestLine.txt
data1.txt
data2.txt
11.2.py

Process finished with exit code 0

3.2)/usr/bin/python3.9 /home/student/Документы/uc27klipan/pz/pz 17/17.3.2.py
File sizes in the 'test' folder:
Студент группы ИС-27 Клипань Аким (1).pdf: 96831 bytes
pz 6.2.py: 769 bytes

Process finished with exit code 0

3.3) /usr/bin/python3.9 /home/student/Документы/uc27klipan/pz/pz 17/17.3.3.py
Shortest filename in PZ11: 11.1.py

Process finished with exit code 0

3.4) /usr/bin/python3.9 /home/student/Документы/uc27klipan/pz/pz 17/17.3.4.py

Process finished with exit code 0

3.5) /usr/bin/python3.9 /home/student/Документы/uc27klipan/pz/pz 17/17.3.5.py
File 'test/test1/test.txt' successfully deleted.

Process finished with exit code 0

Вывод:

Мы закрепили усвоенные знания, понятия, алгоритмы, основные принципы составления программ, мы приобрели навыки составление программ с использованием GUI Tkinter в IDE PyCharm Community, изучили возможности модуля OS.