Студентка группы ИС-23 Васильева А. Н.

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

Тип алгоритма: циклический

Текст программы:

```
import tkinter as tk

def calculate_distance():
    v1 = float(entry_v1.get())
    v2 = float(entry_v2.get())
    s = float(entry_s.get())
    t = float(entry_t.get())

    initial_distance = abs(s)
    total_distance = t * (v1 + v2)
    final_distance = abs(initial_distance - total_distance)

    result_label.config(text=f"Distance between the cars after {t} hours:
{final_distance} km")


root = tk.Tk()
root.title("Car Distance Calculator")


label_v1 = tk.Label(root, text="Speed of Car 1 (km/h):")
label_v1.pack()
entry_v1 = tk.Entry(root)
entry_v1.pack()

label_v2 = tk.Label(root, text="Speed of Car 2 (km/h):")
label_v2.pack()
entry_v2 = tk.Entry(root)
entry_v2.pack()

label_s = tk.Label(root, text="Initial Distance (km):")
label_s.pack()
entry_s = tk.Entry(root)
entry_s.pack()

label_t = tk.Label(root, text="Time (hours):")
label_t.pack()
entry_t = tk.Entry(root)
entry_t.pack()
```

```
calculate_button = tk.Button(root, text="Calculate Distance",
command=calculate_distance)
calculate_button.pack()

result_label = tk.Label(root, text="")
result_label.pack()

root.mainloop()
```

Протокол программы:

Process finished with exit code 0

Постановка задачи №2: 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 ПЗ6, а в папку test1 - один файл из ПЗ7. Файл из ПЗ7 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:
```
import os
import sys
import subprocess

project_root = os.path.abspath('../')

paths = {
    "pz_6": os.path.join(project_root, 'ПЗ№6'),
    "pz_7": os.path.join(project_root, 'ПЗ№7', 'pz_7_1.py'),
    "pz_11": os.path.join(project_root, 'ПЗ№11'),
    "test": os.path.join(project_root, 'test'),
    "test1": os.path.join(project root, 'test', 'test1'),
    "test_file": os.path.join(project_root, 'test', 'test1', 'test.txt'),
    "reports": os.path.join(project_root, 'reports'),
    "report_pdf": 'PZ_7 (1).pdf'
}
```

```
def open_file(filename):
    if sys.platform == "win32":
        os.startfile(filename)
    else:
        opener = "open" if sys.platform == "darwin" else "xdg-open"
        subprocess.call([opener, filename])


def change_directory(path):
    if os.path.exists(path):
        os.chdir(path)
        return True
    else:
        print(f"Каталог {path} не найден")
        return False


def copy_file(source, destination):
    if os.path.exists(source):
        with open(source, 'rb') as f_src, open(destination, 'wb') as f_dst:
            f_dst.write(f_src.read())
    else:
        print(f"Файл {source} не найден")


def list_files_in_directory(path):
    if os.path.exists(path):
        os.chdir(path)
        return [f for f in os.listdir() if os.path.isfile(f)]
    else:
        print(f"Каталог {path} не найден")
        return []


def create_directory(path):
    os.makedirs(path, exist_ok=True)


def print_file_sizes(directory):
    if os.path.exists(directory):
        files = [f for f in os.listdir(directory) if
os.path.isfile(os.path.join(directory, f))]
        for file in files:
            file_path = os.path.join(directory, file)
            print(f"Размер файла {file}: {os.path.getsize(file_path)} байт")
    else:
        print(f"Каталог {directory} не найден")


os.chdir(project_root)

files_in_pz11 = list_files_in_directory(paths['pz_11'])

print("Файлы в каталоге PZ 11:", files in pz11)

create_directory(paths['test1'])

files_to_copy = ['pz_6_1.py', 'pz_6_2.py']
for file in files_to_copy:
    src = os.path.join(paths['pz_6'], file)
    dst = os.path.join(paths['test'], file)
    copy_file(src, dst)
```

```
copy_file(paths['pz_7'], paths['test_file'])

print_file_sizes(paths['test'])

if files_in_pz11:
    shortest_filename = min(files_in_pz11, key=len)
    print("Файл с самым коротким именем:",
os.path.basename(shortest_filename))


if change_directory(paths['reports']) and
os.path.exists(paths['report_pdf']):
    open_file(paths['report_pdf'])
else:
    print(f"PDF файл {paths['report_pdf']} не найден")


if os.path.exists(paths['test_file']):
    os.remove(paths['test_file'])
    print(f"Файл {paths['test_file']} успешно удален")
else:
    print(f"Файл {paths['test_file']} не найден для удаления")
```

Протокол программы:

Файлы в каталоге PZ_11: ['PZ_11.pdf', 'pz_11_1.py', 'pz_11_2.py']

Размер файла pz_6_1.py: 493 байт

Размер файла pz_6_2.py: 382 байт

Файл с самым коротким именем: PZ_11.pdf

Файл C:\Users\User\Documents\PZ_24\test\test1\test.txt успешно удален

Process finished with exit code 0

Вывод: в процессе выполнения практической работы я закрепила усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрела навыки работы с БД в IDE PyCharm Community.

Выполнены разработка кода, отладка, тестирование, оптимизация программного кода.

Готовые программные коды выложены на GitHub.