TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP

KHOA ĐIỆN TỬ

Bộ môn: Công nghệ Thông tin.

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC

LẬP TRÌNH PYTHON

Sinh viên : Phạm Duy Tiến Minh
Lớp : K58KTP.K01
Giáo viên GIẢNG DẠY : Nguyễn Văn Huy
Link GitHub: https://github.com/o0myhomeland0o/BaoCaoPython

Thái Nguyên – 2025

TRƯỜNG ĐHKTCN **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

KHOA ĐIỆN TỬ ***Độc lập - Tự do - Hạnh phúc***

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC: LẬP TRÌNH PYTHON

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

Sinh viên *:Phạm Duy Tiến Minh*

Lớp *:K58KTP.K01*

Ngành: Kỹ Thuật Máy Tính

Giáo viên hướng dẫn : Nguyễn Văn Huy

Ngày giao đề:20/05/2025 *Ngày hoàn thành:10/06/2025*

Tên đề tài : Tạo ứng dụng GUI cho phép chọn thư mục, liệt kê file theo từng loại (.txt, .py, .jpg), và cho phép mở file.

Yêu cầu :

- Đầu vào: Nút “Chọn thư mục”.
- Đầu ra: Treeview hoặc Listbox hiển thị file, nút “Mở”.
- Sử dụng `os` để scan folder.
- Bắt lỗi không tìm thấy đường dẫn.
- GUI với Treeview (tkinter.ttk).
- Mở file bằng chương trình mặc định.

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

Thái Nguyên, ngày…..tháng…..năm 20.....

GIÁO VIÊN HƯỚNG DẪN

(Ký ghi rõ họ tên)

Lời nói đầu

Trong bối cảnh công nghệ thông tin phát triển mạnh mẽ như hiện nay, các ứng dụng quản lý dữ liệu đóng vai trò thiết yếu trong việc tối ưu hóa công việc và nâng cao hiệu suất cá nhân cũng như tổ chức. Việc quản lý thư mục một cách hiệu quả là một nhu cầu không thể thiếu đối với mọi cá nhân và doanh nghiệp.

Bài sau trình bày chi tiết quá trình xây dựng ứng dụng quản lý thư mục GUI đơn giản, nhằm mục đích cung cấp một công cụ trực quan và dễ sử dụng để quản lý thư mục. Ứng dụng được phát triển dựa trên ngôn ngữ lập trình Python, kết hợp với các thư viện GUI để tạo ra một giao diện thân thiện với người dùng.

Thông qua báo cáo này, chúng ta sẽ cùng tìm hiểu về các bước triển khai, các công nghệ được áp dụng, cũng như những kinh nghiệm thu được trong quá trình phát triển ứng dụng. Hy vọng rằng, sản phẩm này không chỉ đáp ứng được các yêu cầu đặt ra mà còn là nền tảng để phát triển những ứng dụng quản lý dữ liệu phức tạp hơn trong tương lai.

Mục Lục

Chương 1. Giới thiệu đề bài

1.1. Giới thiệu đề bài

Trong bài số 9, sinh viên được yêu cầu xây dựng một ứng dụng quản lý danh bạ đơn giản sử dụng ngôn ngữ lập trình Python kết hợp với thư viện tkinter để thiết kế giao diện đồ họa người dùng **(**GUI**).** Mục tiêu của bài tập là tạo ra một ứng dụng cho phép người dùng có thể thực hiện các chức năng cơ bản như duyệt file, mở file, phân loại định dạng (.txt, .py, .jpg)**..**

Ứng dụng sẽ bao gồm các thành phần giao diện như: có Treeview hiển thị file, nút "Chọn thư mục", nút "Mở file" và os để quét thư mục, filedialog để chọn folder, os.startfile() để mở file.

1.2. Các tính năng chính của chương trình

- Duyệt và hiển thị thư mục Cho phép người dùng chọn thư mục từ hệ thống bằng hộp thoại.Hiển thị cấu trúc cây thư mục và file trong Treeview.
- Phân loại file theo định dạng: Tự động nhận diện loại file (.txt, .py, .jpg, ...).Hiển thị biểu tượng hoặc nhãn phân biệt (Text, Python, Image, ...).
- Mở file nhanh: Mở file bằng chương trình mặc định khi double-click hoặc nhấn nút "Mở".Hỗ trợ mở nhiều định dạng: văn bản, hình ảnh, code, PDF,...
- Điều hướng thư mục: Truy cập vào thư mục con bằng double-click.Quay lại thư mục (có thể bổ sung nút "Back" nếu phát triển thêm).
- Xử lý lỗi: Thông báo lỗi nếu đường dẫn không tồn tại hoặc không có quyền truy cập. Bỏ qua file/thư mục hệ thống bị khóa.
- Giao diện thân thiệnSử dụng tkinter/ttk để thiết kế GUI đơn giản, dễ sử dụng.Thanh cuộn (scrollbar) để xem nội dung dài.

1.3. Kiến thức được áp dụng

Để hoàn thành bài tập, sinh viên cần vận dụng kết hợp nhiều kiến thức và kỹ năng, bao gồm:

- Lập trình hướng đối tượng (OOP): Tổ chức mã nguồn thành các class.

- Thiết kế lập trình GUI với tkinter: Tạo ra cửa sổ ứng dụng thân thiện, dễ sử dụng.
- Xử lý sự kiện giao diện: Gắn các thao tác (Chọn, mở tập tin, Back) với các nút bấm.
- Tách biệt logic và giao diện: Tổ chức mã nguồn thành nhiều module (contacts.py, main.py) để dễ bảo trì.

1.4. Thách thức khi thực hiện

Trong quá trình phát triển ứng dụng, sinh viên có thể gặp một số khó khăn như:

- Xử Lý Hệ Thống File đọc nội dung thư mục có hàng nghìn file, hay đường dẫn Unicode (tiếng Việt, emoji) có thể gây lỗi.
- Khi Hiển Thị Treeview load thư mục lớn vào Treeview → ứng dụng đơ, tốn RAM. Khó cuộn mượt khi có hàng trăm file.
- Phân Loại File & Hiển Thị Icon nhận diện loại file (.txt, .py, .jpg) từ phần mở rộng. Hiển thị icon tương ứng (khó với tkinter do thiếu hỗ trợ sẵn).
- Mở File Đa Nền Tảng os.startfile() chỉ hoạt động trên Windows. Trên macOS/Linux cần dùng lệnh khác (xdg-open, open)

Chương 2. Cơ sở lý thuyết

2.1. Quản Lý Hệ Thống Tệp (File System)

- File và Directory (Tệp và Thư mục): Tệp (File): Đơn vị lưu trữ dữ liệu (ví dụ: .txt, .py, .jpg), Thư mục (Directory): Chứa các tệp hoặc thư mục con, tổ chức dữ liệu theo cấu trúc phân cấp.
- Đường dẫn (Path): Tuyệt đối (Absolute Path): Bắt đầu từ thư mục gốc (ví dụ: C:\Users\Ten\Documents), Tương đối (Relative Path): Dựa trên vị trí hiện tại (ví dụ: ./Downloads).
- Phân Loại File & Hiển Thị Icon nhận diện loại file (.txt, .py, .jpg) từ phần mở rộng. Hiển thị icon tương ứng (khó với tkinter do thiếu hỗ trợ sẵn).
- Mở File Đa Nền Tảng os.startfile() chỉ hoạt động trên Windows. Trên macOS/Linux cần dùng lệnh khác (xdg-open, open)

2.2. Giao Diện Đồ Họa Người Dùng (GUI)

Tkinter và ttk:

- Tkinter: Thư viện tiêu chuẩn của Python để tạo GUI, hỗ trợ các widget cơ bản (Button, Label, Entry).
- ttk (Themed Tkinter): Phiên bản nâng cao của Tkinter, cung cấp các widget hiện đại (Treeview, Notebook).

Treeview:

- Widget hiển thị dữ liệu dạng cây/bảng, phù hợp để hiển thị cấu trúc thư mục.
- Hỗ trợ phân cấp (thư mục cha/con) và sắp xếp theo cột.

2.3. Tương Tác Với Hệ Thống Tệp

Module os và os.path:

- os.listdir(): Liệt kê nội dung thư mục.
- os.path.join(): Kết hợp đường dẫn an toàn giữa các hệ điều hành.

- os.path.isfile()/os.path.isdir(): Kiểm tra loại đối tượng (tệp/thư mục)

Mở Tệp với Chương Trình Mặc Định:

- os.startfile() (Windows) hoặc subprocess.run() (macOS/Linux).

2.4. Xử Lý Sự Kiện (Event Handling)

Binding sự kiện:

- Gán hành động cho các sự kiện như click chuột (<Button-1>), double-click (<Double-1>).

Callback functions:

- Hàm được gọi khi sự kiện xảy ra (ví dụ: mở tệp khi double-click).

Chương 3. Thiết kế và xây dựng chương trình

3.1. Sơ đồ khối hệ thống

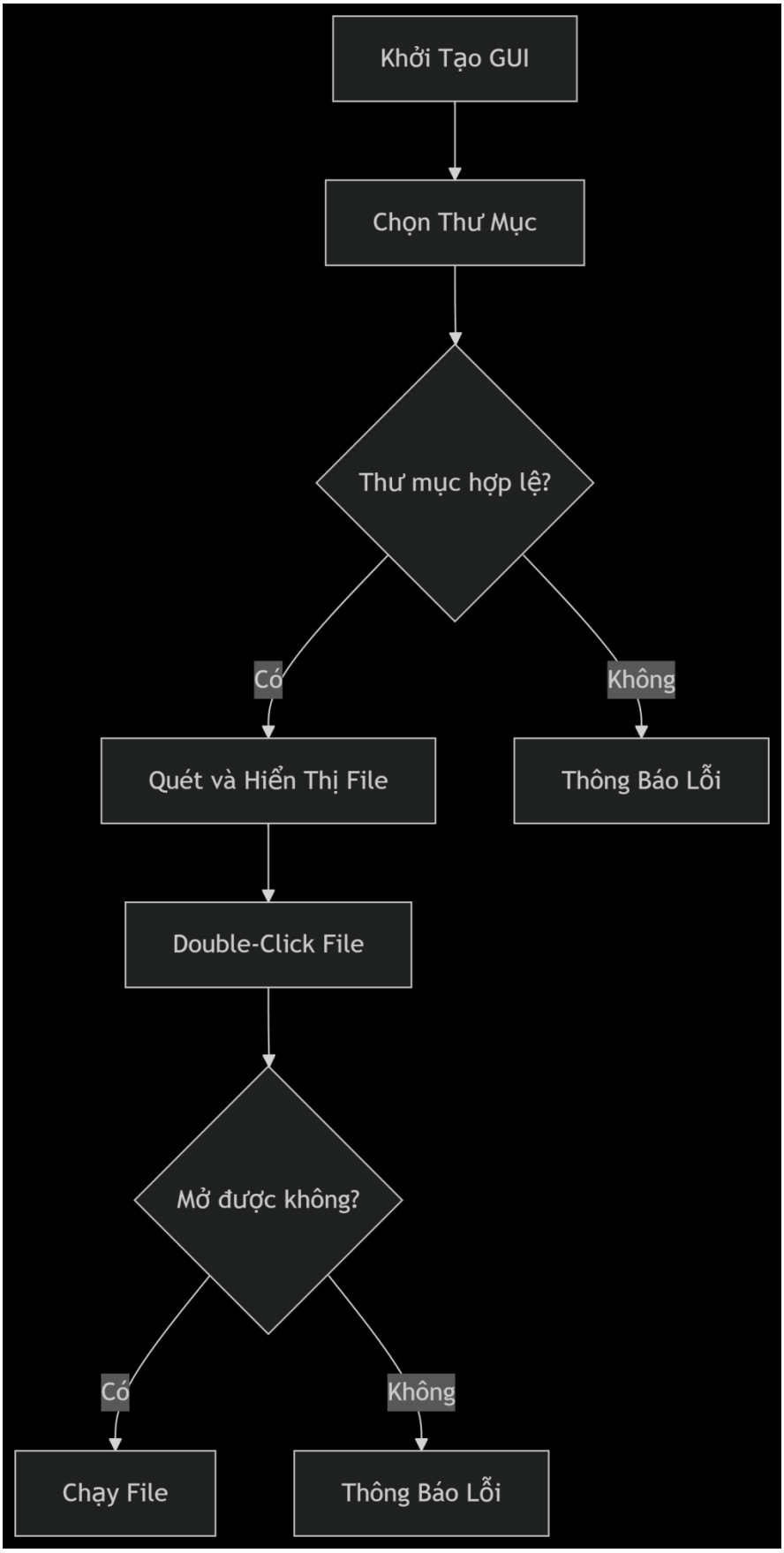
Chương trình được thiết kế gồm:

- Giao diện: xây dựng bằng tkinter và ttk (GUI), hiển thị cửa sổ chính, nút bấm, và danh sách file.
- Xử lý logic (Backend): tương tác với hệ thống file qua thư viện os, xử lý sự kiện (mở file, duyệt thư mục).
- Xử lý lỗi (Error Handling): Bắt các ngoại lệ (PermissionError, FileNotFound).

3.1.1 Các Module chính:

Tên	Chức năng
gui.py	Xây dựng giao diện người dùng bằng tkinter/ttk
file_manager.py	Tương tác với hệ thống tệp
event_handler.py	Gán hành động cho các sự kiện GUI
error_handler.py	Bắt và thông báo lỗi
main.py	Kết nối các module và khởi chạy ứng dụng

3.1.2 Biểu đồ:


Khởi Tạo GUI
Chọn Thư Mục
Thư mục hợp lệ?
Có
Không
Quét và Hiển Thị File
Thông Báo Lỗi
Double-Click File
Mở được không?
Có
Không
Chạy File
Thông Báo Lỗi

3.2. Sơ đồ khối các thuật toán chính

Các thuật toán chính được sử dụng trong chương trình và dưới đây là mô tả chức năng từng thuật toán:

3.2.1. Thuật Toán Quét Thư Mục (Directory Traversal)

Chức năng: Đọc và liệt kê nội dung thư mục bằng os.listdir() hoặc os.scandir() để lấy danh sách file/thư mục con sau đó phân loại file dựa trên phần mở rộng (.txt, .py, .jpg...) bằng os.path.splitext().

3.2.2. Thuật Toán Phân Loại File (File Classification)

Chức năng: Nhóm file theo loại để hiển thị trong Treeview.

3.2.3 Thuật Toán Sắp Xếp File (Sorting)

Chức năng: Hiển thị file theo thứ tự (tên, loại, kích thước). Sắp xếp danh sách file trước khi đưa vào Treeview. Hoặc dùng Treeview.sort() để sắp xếp động trên GUI.

3.2.4 Thuật Toán Mở File Đa Nền Tảng (Cross-Platform File Opening)

Chức năng: Mở file bằng ứng dụng mặc định trên Windows/ macOS/ Linux

3.2.5 Thuật Toán Xử Lý Sự Kiện (Event Handling)

Chức năng: Gán hành động cho nút bấm/double-click.

3.2.6 Thuật Toán Tải Chậm (Lazy Loading)

Chức năng: Tối ưu hiệu suất khi thư mục chứa nhiều file.

3.2.7 Thuật Toán Xử Lý Lỗi (Error Handling)

Chức năng: Bắt lỗi đường dẫn/quyền truy cập.

3.3. Cấu trúc dữ liệu

Các trường thông tin:
Ứng dụng quản lý các thông tin sau về mỗi file/thư mục:

- Tên file (file_name): string (vd: data.csv).

- Đường dẫn (file_path): string (vd: C:\Users\data.csv).
- Loại file (file_type): string (vd: CSV File).
- Kích thước (file_size): integer (bytes) hoặc string (vd: 2.5 MB).
- Thời gian sửa đổi (last_modified): datetime hoặc string.

3.4. Các hàm trong chương trình

Chương trình được thiết kế theo hướng chia module rõ ràng giữa xử lý dữ liệu và giao diện. Dưới đây là mô tả các hàm chính:

Tên hàm	Chức năng
__init__(self, root)	hàm khởi tạo (constructor) của lớp DirectoryManagerApp. Nó được gọi khi một đối tượng DirectoryManagerApp được tạo ra
select_directory(self)	mở hộp thoại cho phép người dùng chọn một thư mục trên hệ thống
show_files(self, directory)	Hiển thị danh sách các tệp và thư mục con trong thư mục đã chọn lên Treeview
on_double_click(self, event)	Xử lý sự kiện khi người dùng nhấp đúp chuột vào một mục trong Treeview
on_tree_select(self, event)	Xử lý sự kiện khi người dùng chọn (hoặc bỏ chọn) một mục trong Treeview
open_selected_file(self)	Mở tệp được chọn trong Treeview bằng chương trình mặc định của hệ điều hành
open_file(self)	Đây là hàm được gọi khi người dùng nhấp vào nút "Mở tập tin"

Chương 4. Thực nghiệm và kết luận

4.1. Giới thiệu về Python

Python là một ngôn ngữ lập trình bậc cao, thông dịch, được phát triển bởi Guido van Rossum và phát hành lần đầu tiên vào năm 1991. Python nổi bật nhờ cú pháp rõ ràng, dễ đọc, dễ viết và có cộng đồng phát triển rất lớn. Đây là một trong những ngôn ngữ phổ biến nhất hiện nay trong nhiều lĩnh vực như: phát triển phần mềm, trí tuệ nhân tạo, xử lý dữ liệu, phát triển web, tự động hoá và cả lập trình ứng dụng giao diện người dùng (GUI).

Một số đặc điểm chính của Python:

- Dễ học và dễ đọc: Cú pháp đơn giản, gần giống ngôn ngữ tự nhiên giúp người học nhanh chóng tiếp cận.
- Hỗ trợ lập trình hướng đối tượng: Cho phép tổ chức mã nguồn theo hướng module, class và tái sử dụng mã.
- Thư viện phong phú: Có sẵn nhiều thư viện như json, tkinter, os, re, pandas… giúp lập trình viên phát triển nhanh các ứng dụng.
- Tương thích đa nền tảng: Chạy được trên Windows, Linux, macOS mà không cần chỉnh sửa mã nguồn.

Lý do chọn Python cho bài :

- Có sẵn thư viện GUI là tkinter rất phù hợp để thiết kế ứng dụng đơn giản với các thành phần như: Entry, Button, Treeview...
- Cú pháp gọn gàng, rõ ràng giúp tập trung vào giải quyết bài toán hơn là xử lý lỗi cú pháp phức tạp.
- Tốc độ phát triển nhanh: Phù hợp với yêu cầu làm dự án cá nhân, học tập trong thời gian ngắn.

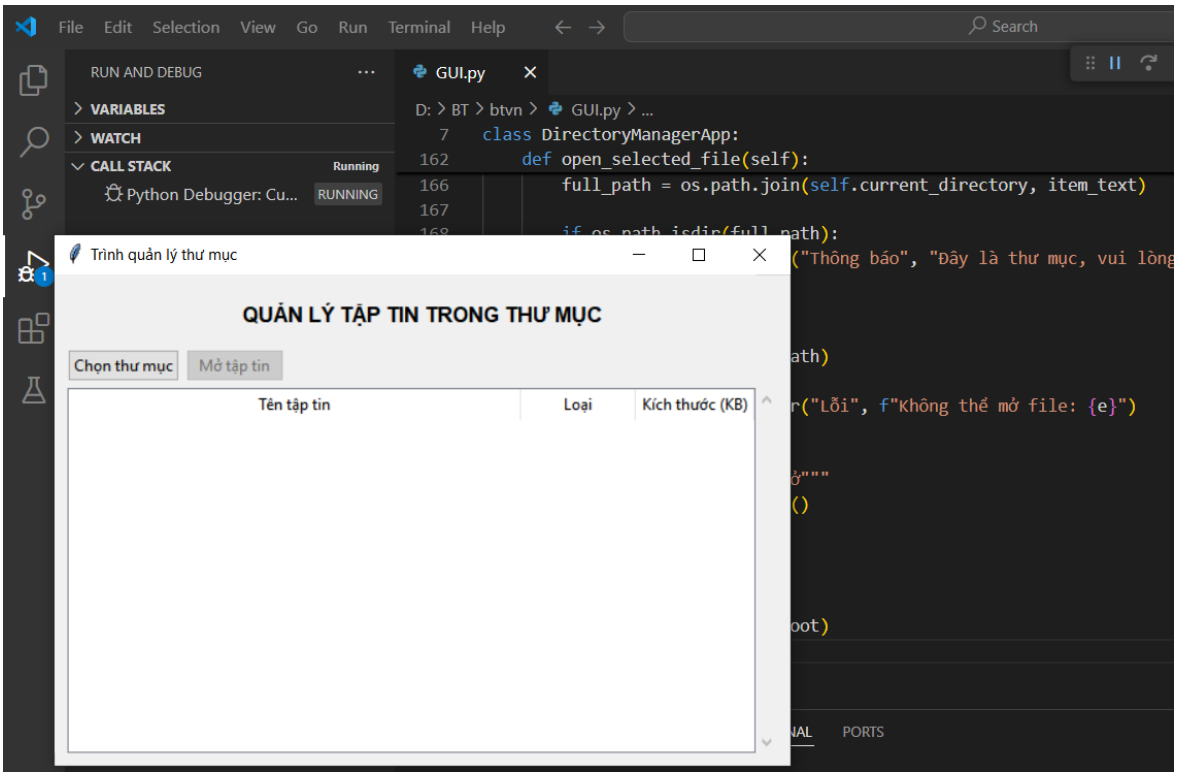
Công cụ sử dụng để lập trình

- Python phiên bản 3.x (khuyến nghị từ 3.9 trở lên).
- Môi trường phát triển: VS Code hoặc IDLE.
- Thư viện sử dụng:
    - tkinter: để thiết kế giao diện người dùng.

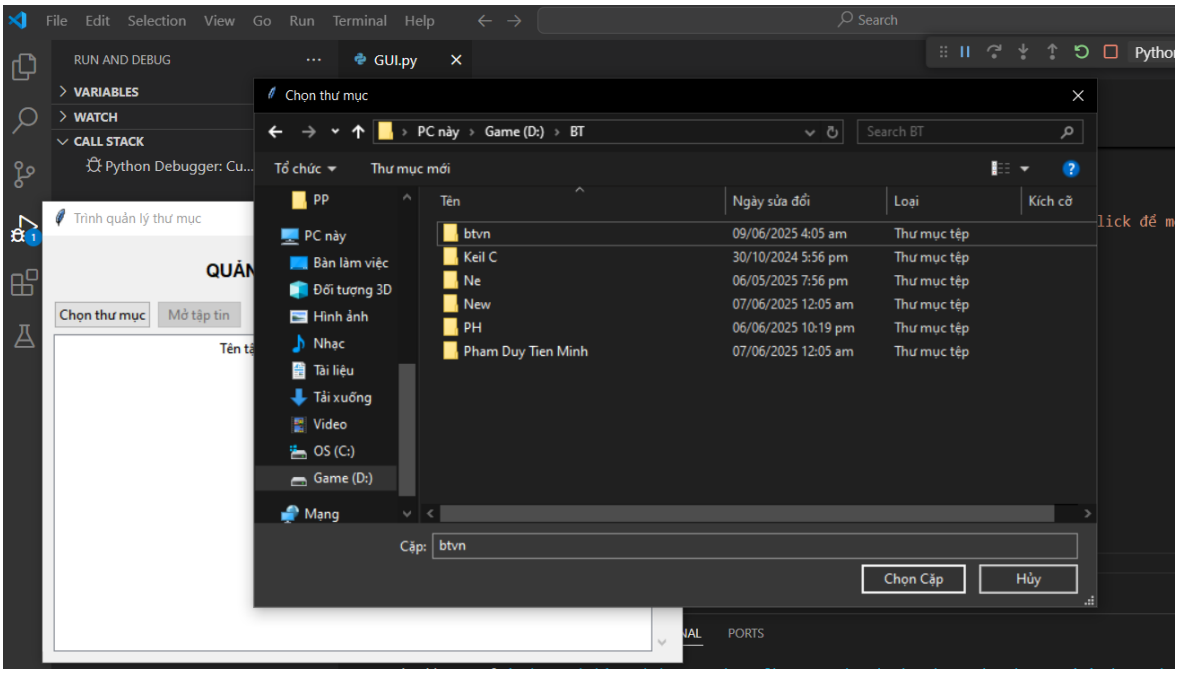
4.2. Thực nghiệm

Sau khi hoàn thiện chương trình, tiến hành chạy thử và kiểm tra lần lượt các chức năng yêu cầu ứng dụng quản lý thư mục. Các kết quả thực nghiệm được ghi lại như sau:

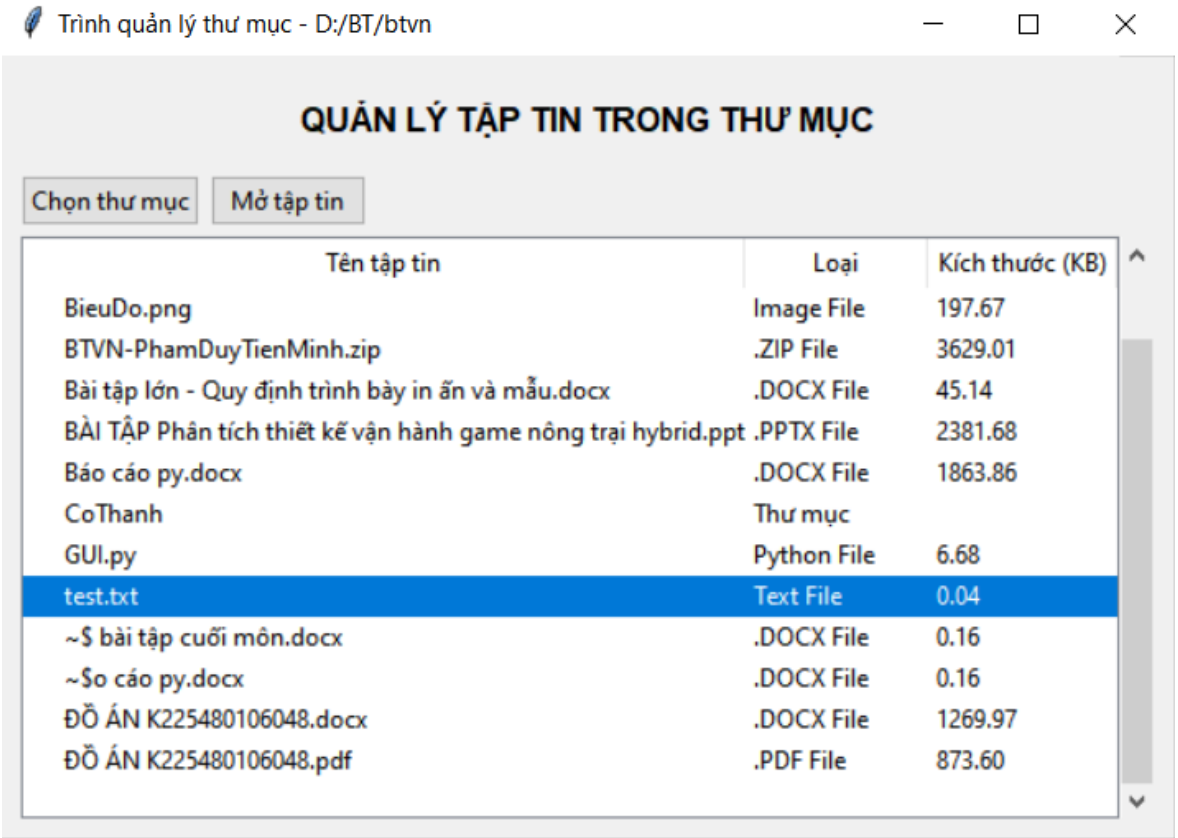
Chạy code hiển thị được giao diện



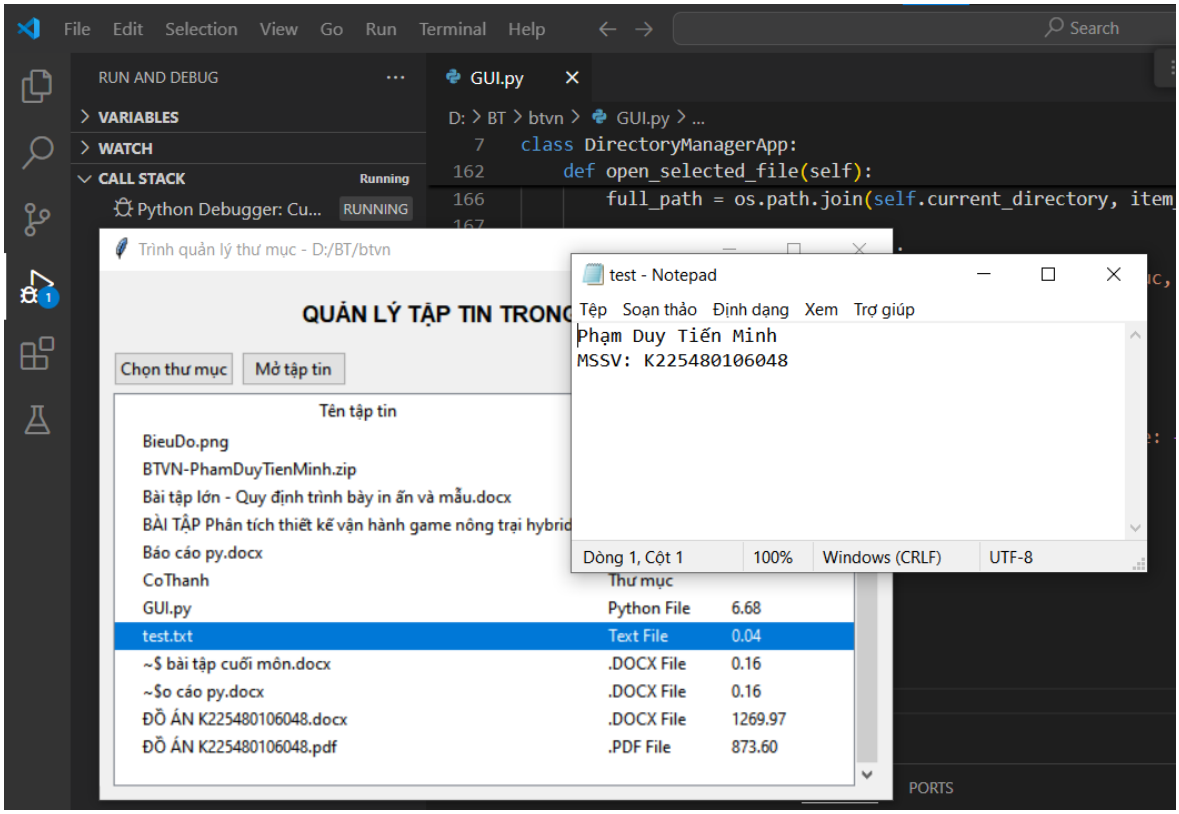
Chọn thư mục và bắt đầu tìm kiếm thư mục ta muốn và chọn



Thư mục được hiển thị theo tên loại và kích thước



Chọn file ta muốn và bấm mở tập tin ta được kết quả



Code bài :

```
import os
import tkinter as tk
from tkinter import ttk, filedialog, messagebox
from tkinter.font import Font


class DirectoryManagerApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Trình quản lý thư mục")
        self.root.geometry("800x600")

        # Tạo font đậm cho tiêu đề
        bold_font = Font(weight="bold")

        # Frame chính
```

```
        self.main_frame = ttk.Frame(root, padding="10")
        self.main_frame.pack(fill=tk.BOTH, expand=True)

        # Tiêu đề
        self.title_label = ttk.Label(
            self.main_frame,
            text="QUẢN LÝ TẬP TIN TRONG THƯ MỤC",
            font=bold_font
        )
        self.title_label.pack(pady=10)

        # Frame chứa nút chọn thư mục
        self.button_frame = ttk.Frame(self.main_frame)
        self.button_frame.pack(fill=tk.X, pady=5)

        # Nút chọn thư mục
        self.select_button = ttk.Button(
            self.button_frame,
            text="Chọn thư mục",
            command=self.select_directory
        )
        self.select_button.pack(side=tk.LEFT)

        # Nút mở file
        self.open_button = ttk.Button(
            self.button_frame,
            text="Mở tập tin",
            command=self.open_file,
            state=tk.DISABLED
        )
        self.open_button.pack(side=tk.LEFT, padx=5)

        # Thanh cuộn
        self.scrollbar = ttk.Scrollbar(self.main_frame)
        self.scrollbar.pack(side=tk.RIGHT, fill=tk.Y)

        # Treeview để hiển thị file
        self.tree = ttk.Treeview(
            self.main_frame,
            columns=("Type", "Size"),
            yscrollcommand=self.scrollbar.set,
            selectmode="browse"
        )
```

```
        self.tree.pack(fill=tk.BOTH, expand=True)

        # Cấu hình thanh cuộn
        self.scrollbar.config(command=self.tree.yview)

        # Đặt tên cho các cột
        self.tree.heading("#0", text="Tên tập tin")
        self.tree.heading("Type", text="Loại")
        self.tree.heading("Size", text="Kích thước (KB)")

        # Đặt chiều rộng cột
        self.tree.column("#0", width=400)
        self.tree.column("Type", width=150)
        self.tree.column("Size", width=150)

        # Gắn sự kiện double click
        self.tree.bind("<Double-1>", self.on_double_click)
        self.tree.bind("<<TreeviewSelect>>", self.on_tree_select)

        # Biến lưu thư mục hiện tại
        self.current_directory = ""

    def select_directory(self):
        """Mở hộp thoại chọn thư mục và hiển thị các file"""
        directory = filedialog.askdirectory(title="Chọn thư mục")
        if directory:
            try:
                self.current_directory = directory
                self.show_files(directory)
                self.root.title(f"Trình quản lý thư mục - {directory}")
            except Exception as e:
                messagebox.showerror("Lỗi", f"Không thể đọc thư mục: {e}")

    def show_files(self, directory):
        """Hiển thị các file trong thư mục lên Treeview"""
        # Xóa dữ liệu cũ
        for item in self.tree.get_children():
            self.tree.delete(item)

        try:
            # Lấy danh sách file và thư mục con
            items = os.listdir(directory)
```

```
            # Phân loại và thêm vào Treeview
            for item in items:
                full_path = os.path.join(directory, item)

                if os.path.isdir(full_path):
                    # Nếu là thư mục
                    self.tree.insert("", tk.END, text=item, values=("Thư mục",
""))
                else:
                    # Nếu là file
                    file_ext = os.path.splitext(item)[1].lower()
                    file_size = os.path.getsize(full_path) / 1024  # KB

                    # Xác định loại file
                    if file_ext == ".txt":
                        file_type = "Text File"
                    elif file_ext == ".py":
                        file_type = "Python File"
                    elif file_ext in (".jpg", ".jpeg", ".png", ".gif"):
                        file_type = "Image File"
                    else:
                        file_type = f"{file_ext.upper()} File"

                    self.tree.insert(
                        "", tk.END,
                        text=item,
                        values=(file_type, f"{file_size:.2f}"),
                        tags=("file",)
                    )

        except PermissionError:
            messagebox.showerror("Lỗi", "Không có quyền truy cập thư mục
này")
        except Exception as e:
            messagebox.showerror("Lỗi", f"Không thể đọc thư mục: {e}")

    def on_double_click(self, event):
        """Xử lý sự kiện double click để mở file hoặc thư mục"""
        item = self.tree.selection()[0]
        item_text = self.tree.item(item, "text")
        item_values = self.tree.item(item, "values")

        full_path = os.path.join(self.current_directory, item_text)
```

```
        if os.path.isdir(full_path):
            # Nếu là thư mục, mở thư mục đó
            self.current_directory = full_path
            self.show_files(full_path)
            self.root.title(f"Trình quản lý thư mục - {full_path}")
        else:
            # Nếu là file, mở file
            self.open_selected_file()

    def on_tree_select(self, event):
        """Kích hoạt nút Mở khi có item được chọn"""
        selected_items = self.tree.selection()
        if selected_items:
            self.open_button.config(state=tk.NORMAL)
        else:
            self.open_button.config(state=tk.DISABLED)

    def open_selected_file(self):
        """Mở file được chọn bằng chương trình mặc định"""
        selected_item = self.tree.selection()[0]
        item_text = self.tree.item(selected_item, "text")
        full_path = os.path.join(self.current_directory, item_text)

        if os.path.isdir(full_path):
            messagebox.showinfo("Thông báo", "Đây là thư mục, vui lòng
double click để mở")
            return

        try:
            os.startfile(full_path)
        except Exception as e:
            messagebox.showerror("Lỗi", f"Không thể mở file: {e}")

    def open_file(self):
        """Xử lý khi nhấn nút Mở"""
        self.open_selected_file()
```

```
if __name__ == "__main__":
    root = tk.Tk()
    app = DirectoryManagerApp(root)
    root.mainloop()
```

4.3. Kết luận

4.3.1. Những gì sản phẩm làm được.

Với việc hoàn thành ứng dụng quản lý thư mục, chúng ta đã thành công xây dựng một công cụ thiết thực, đáp ứng đầy đủ các yêu cầu đã đặt ra. Ứng dụng cho phép người dùng quản lý và mở các file một cách dễ dàng thông qua giao diện trực quan. Ứng dụng tải dữ liệu khi thao tác, đảm bảo tính tiện lợi khi quản lý.

4.3.2. Những gì đã học được.

Trong quá trình phát triển ứng dụng này, chúng ta đã tiếp thu được nhiều kiến thức và kỹ năng quan trọng:

- Lập trình giao diện người dùng (GUI) với Tkinter: Nắm vững cách sử dụng các widget cơ bản như Entry, Button, Label, và đặc biệt là Treeview để hiển thị dữ liệu dạng bảng.
- Quản lý dữ liệu hiệu quả: Xây dựng cấu trúc lớp để quản lý danh sách, để dễ bảo trì.
- Xử lý sự kiện: Hiểu cách liên kết các sự kiện từ giao diện người dùng (nhấn nút) với các hàm xử lý logic nghiệp vụ.

4.3.3. Các cải tiến trong tương lai

Mặc dù ứng dụng đã hoàn thành các yêu cầu đặt ra, vẫn có nhiều hướng để cải thiện và mở rộng tính năng, nâng cao trải nghiệm người dùng và tính mạnh mẽ của sản phẩm:

- Thêm chức năng tìm kiếm file theo tên/định dạng.
- Hỗ trợ thao tác file (xóa, đổi tên, sao chép).
- Giao diện người dùng nâng cao: Cải thiện thẩm mỹ giao diện bằng cách sử dụng các thư viện GUI khác mạnh mẽ hơn như PyQt hoặc Kivy, hoặc thêm các hiệu ứng chuyển động, biểu tượng trực quan.
- Hiển thị thumbnail cho file ảnh
- Hỗ trợ đa ngôn ngữ: Cho phép người dùng chuyển đổi ngôn ngữ giao diện.

Việc tiếp tục phát triển theo những hướng này sẽ biến ứng dụng quản lý thư mục thành một công cụ toàn diện và thân thiện hơn với người dùng.

Lời cảm ơn

Em xin gửi lời cảm ơn chân thành đến những cá nhân và tổ chức đã đóng góp và hỗ trợ trong quá trình hoàn thành báo cáo và phát triển ứng dụng quản lý danh bạ GUI này.

Đầu tiên, tôi xin bày tỏ lòng biết ơn sâu sắc đến Thầy Đỗ Duy Cốp đã tận tình giảng dạy, truyền đạt kiến thức và kinh nghiệm quý báu. Những hướng dẫn, góp ý và sự động viên của Thầy/Cô chính là kim chỉ nam giúp chúng tôi định hướng và hoàn thành tốt nhiệm vụ này.

Chúng tôi cũng xin cảm ơn Trường Đại Học Công Nghiệp Thái Nguyên đã tạo mọi điều kiện thuận lợi về cơ sở vật chất, tài liệu và môi trường học tập, nghiên cứu, giúp chúng tôi có thể tập trung phát triển sản phẩm một cách hiệu quả nhất.

Một lần nữa, chúng tôi xin chân thành cảm ơn sự giúp đỡ, hỗ trợ và đồng hành của tất cả mọi người. Đây là nguồn động lực to lớn giúp chúng tôi không ngừng học hỏi và phát triển.

Trân trọng.

Tài Liệu Tham Khảo

YouTube: FreeCodeCamp - Tkinter Course
Khóa học miễn phí về Tkinter (3 giờ).

YouTube: Python File Explorer
Hướng dẫn xây dựng ứng dụng quản lý file từ A-Z.