廣東工業大學

操作系统课程设计

〈文件系统 GUI 设计与实现〉

学　　院　先进制造学院

专　　业　计算机科学与技术

班　　级　6 班

学　　号　3123008647

姓　　名　林峻安

指导教师　苏畅，李剑峰

目录

五、 程序运行及清单

5.1 初始化界面

5.2 点击刷新文本列表

5.3 点击显示全部文件

5.4 点击刷新文件列表（取消显示全部文件）

5.5 点击新建文件夹

5.6 查看新建的文件夹

5.7 双击打开新建的文件夹

5.8 点击新建文件

5.9 查看新建的文件

5.10 编辑文件

5.11 输入编辑信息

5.12 保存并查看文件大小变化

5.13 查看文件（只读）

5.14 删除文件

5.15 查看删除结果

一、设计概述

1.1 设计背景与目的

在操作系统课程学习中，文件系统是核心概念之一。本次课程设计旨在通过实现一个图形化文件管理器，深入理解文件系统的基本原理、目录结构组织以及用户界面设计等关键技术。通过实践，掌握文件和目录操作的编程实现方法，提升 GUI 应用开发能力。

本文件管理器模拟了一个简化的虚拟文件系统，提供了基本的文件和目录管理功能，帮助理解操作系统中文件系统的工作机制。

1.2 技术选型

- **开发语言**：Python 3.8+
- **GUI 框架**：Tkinter（Python 标准库）
- **图像处理**：Pillow（用于图标加载和处理）
- **其他库**：os, shutil, datetime, base64 等标准库

1.3 系统特点

- **虚拟磁盘机制**：使用本地文件夹模拟虚拟磁盘，支持格式化和初始化
- **完整文件操作**：支持文件和文件夹的创建、删除、编辑、查看等基本操作
- **友好用户界面**：包含路径导航、导航路径、文件列表视图等
- **日志系统**：记录所有操作的时间戳和详情
- **全部文件视图**：支持扫描并显示虚拟磁盘中的所有文件和文件夹
- **单文件应用**：使用 Base64 编码嵌入图标，无需外部资源依赖

二、功能需求分析

2.1 核心功能

文件系统管理：

- 虚拟磁盘的初始化和格式化
- 磁盘空间的模拟管理

文件与目录操作

- 文件夹操作：新建、删除、导航
- 文件操作：新建、删除、查看、编辑
- 支持文件类型识别和显示

用户界面功能

- 路径显示与导航路径
- 文件列表视图
- 操作日志显示
- 状态提示

2.2 界面交互需求

- 直观的操作面板，分类展示不同功能按钮
- 清晰的文件列表，显示名称、类型、大小等信息
- 实时更新的路径显示和导航路径
- 操作反馈机制
- 友好的对话框提示

三、系统设计

3.1 整体架构

本文件管理器采用单窗口多组件的架构设计，主要由以下几个部分组成：

- **顶部栏**：包含路径显示和面包屑导航
- **左侧操作面板**：包含各类操作按钮
- **中央文件列表区**：使用 Treeview 显示文件和文件夹
- **右侧日志区**：显示操作日志

- **底部状态栏**：显示当前操作状态

3.2 模块设计

3.2.1 用户界面模块

负责创建和管理所有 GUI 组件，包括：

- _setup_ui()：初始化主界面布局
- _create_ops_panel()：创建左侧操作面板
- _create_file_list_frame()：创建中央文件列表
- _create_log_frame()：创建右侧日志区域
- 各类按钮和对话框的创建与事件绑定

3.2.2 虚拟磁盘模块

负责虚拟磁盘的管理，包括：

- reinitialize_disk()：初始化虚拟磁盘
- format_disk()：格式化虚拟磁盘
- 磁盘路径管理和目录结构维护

3.2.3 文件与目录操作模块

实现文件和目录的基本操作：

- 文件夹操作：create_new_folder(), delete_selected()
- 文件操作：
  create_new_file(), delete_selected(), view_file(), edit_file()
- 路径导航：navigate_to_path(), on_item_double_click()

3.2.4 辅助功能模块

提供辅助功能支持：

- 日志系统：log_message(), 记录操作日志
- 状态更新：update_status(), 更新状态栏
- 路径显示：update_path_label(), update_breadcrumb(), 管理路径和导航路径
- 工具函数：load_icon(), _create_input_dialog(), 提供通用功能支持

3.3 数据结构设计

3.3.1. 类层次结构

```
class FileManagerApp(tk.Tk):
    # 继承自tkinter.Tk，是程序的主窗口类
```

3.3.2. 实例属性数据结构

路径管理

```
self.virtual_disk_root = str  # 虚拟磁盘根目录
self.current_path = str       # 当前浏览目录的
```

UI 组件对象

```
self.folder_icon = ImageTk.PhotoImage  # 文件夹图标对象
self.file_icon = ImageTk.PhotoImage    # 文件图标对象
self.tree = ttk.Treeview               # 文件列表树形控件
self.log_text = scrolledtext.ScrolledText  # 日志文本区域
self.status_bar = tk.Label             # 状态栏标签
self.path_label = tk.Label             # 路径显示标签
self.breadcrumb_frame = tk.Frame       # 面包屑导航容器
```

3.3.3. Treeview 数据结构

列定义

```
cols = ("名称", "类型", "大小", "路径")
# 每一行的数据格式：
values = (name, type, size, path)
```

数据示例

```
# 文件夹项
("documents", "文件夹", "", "/根目录")

# 文件项
("readme.txt", ".txt", "1024 B", "/根目录")

# 返回上级项
("[..]", "父目录", "", "/根目录")
```

3.3.4. 函数返回值数据结构

get_selected_item_info() 返回值

```
return (name, full_path, item_type)
# name: str - 项目名称
# full_path: str - 完整文件路径
# item_type: str - 项目类型（文件夹/文件扩展名）
```

_create_input_dialog() 返回值

```
result = {"value": None}  # 字典包装的返回值
# result["value"]: str | None - 用户输入的文本或
```

3.3.5. show_all_files() 中的数据结构

all_items 列表结构

```
all_items = [
    {
        'name': str,        # 文件/文件夹名称
        'type': str,        # 类型（文件夹/文件扩展名）
        'size': str,        # 大小字符串
        'path': str,        # 显示路径
        'full_path': str,   # 完整系统路径
        'is_dir': bool      # 是否为目录
    },
    # ... 更多项目
]
```

3.3.6. 路径导航数据结构

路径分割

```
path_parts = rel_path.split(os.sep)  # 路径分段列表
# 例如: ["folder1", "subfolder", "current"]

# 每个按钮的回调参数
partial_path = os.path.join(self.virtual_disk_root, *path_
```

3.3.7. 错误处理数据结构

异常信息封装

```
try:
    # 操作代码
except Exception as e:
    error_info = {
        'operation': str,     # 操作名称
        'error': str(e),      # 错误消息
        'timestamp': str      # 时间戳
    }
```

3.3.8. 日志消息数据结构

日志格式

```
timestamp = datetime.datetime.now().strftime("[%H:%M:%S]")
full_message = f"{timestamp} {message}\n"
```

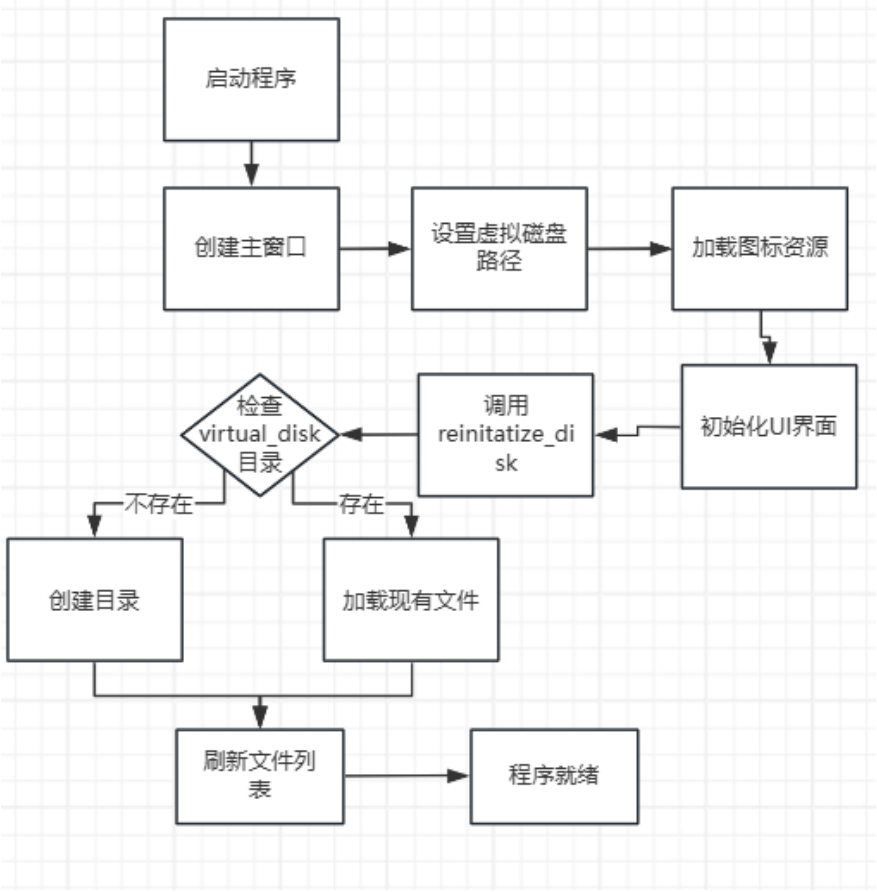
3.3.9. 文件验证数据结构

无效字符集合

```
invalid_chars = '<>:"/\\|?*'  # 字符串形式的字符集
```

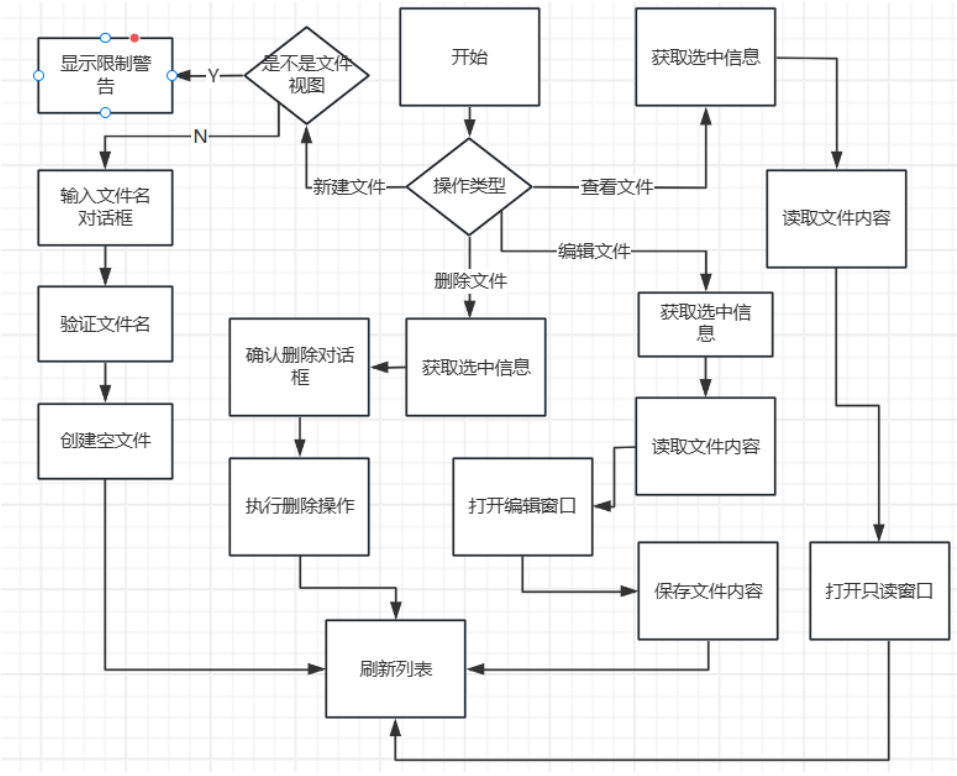
四. 各模块的算法流程图..

4.1. 程序初始化流程

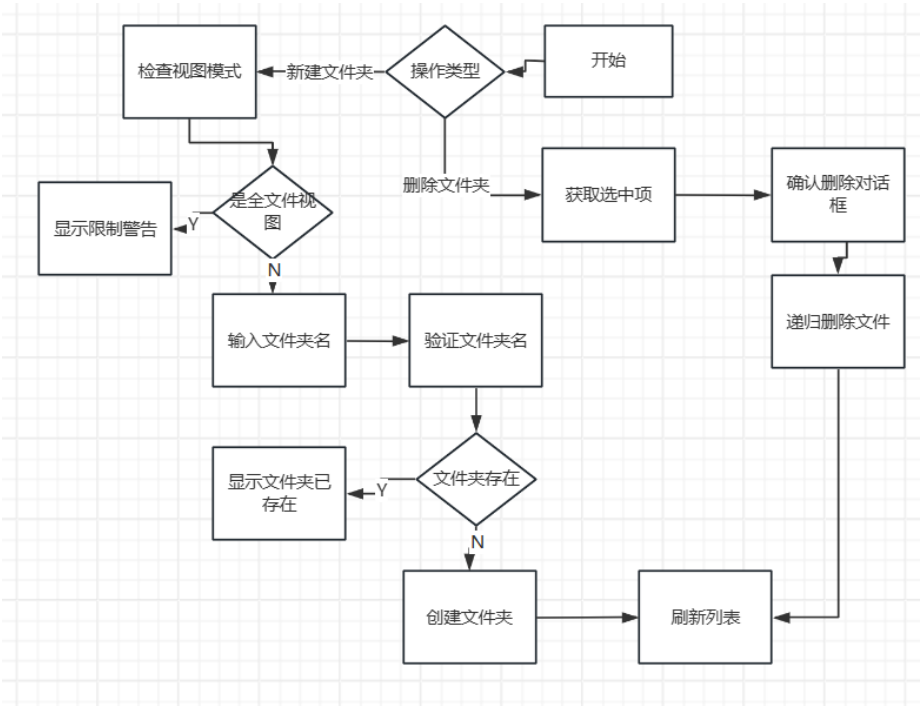

启动程序
创建主窗口
设置虚拟磁盘路径
加载图标资源
初始化UI界面
调用reinitatize_disk
检查virtual_disk目录
不存在
存在
创建目录
加载现有文件
刷新文件列表
程序就绪

4.2. 文件列表刷新流程

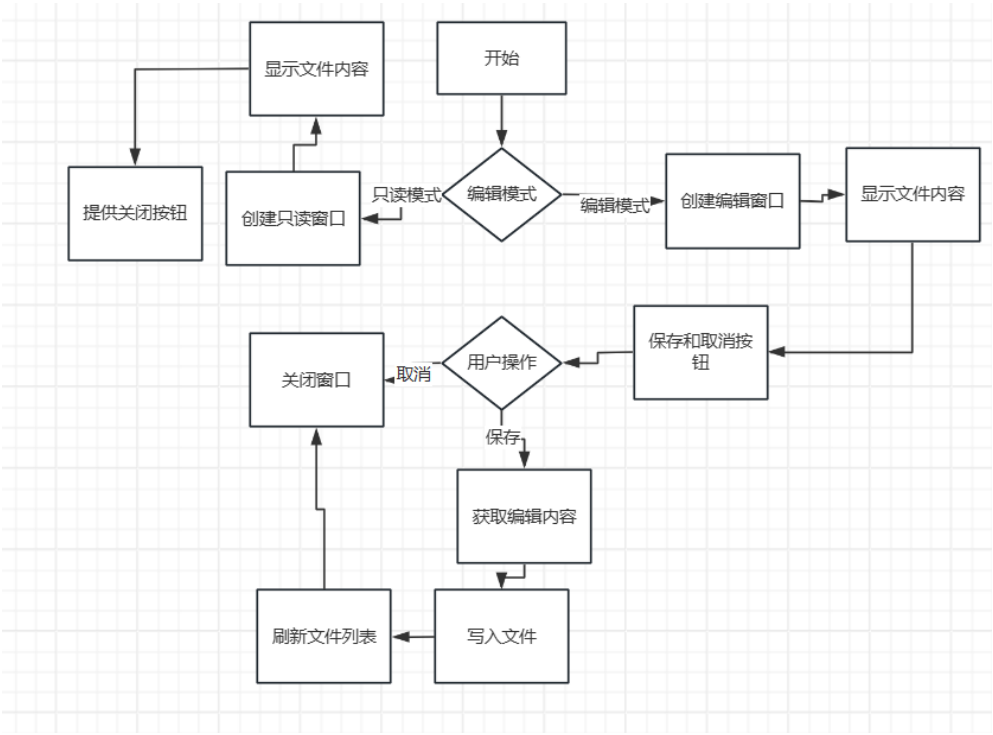
4.3. 文件操作流程



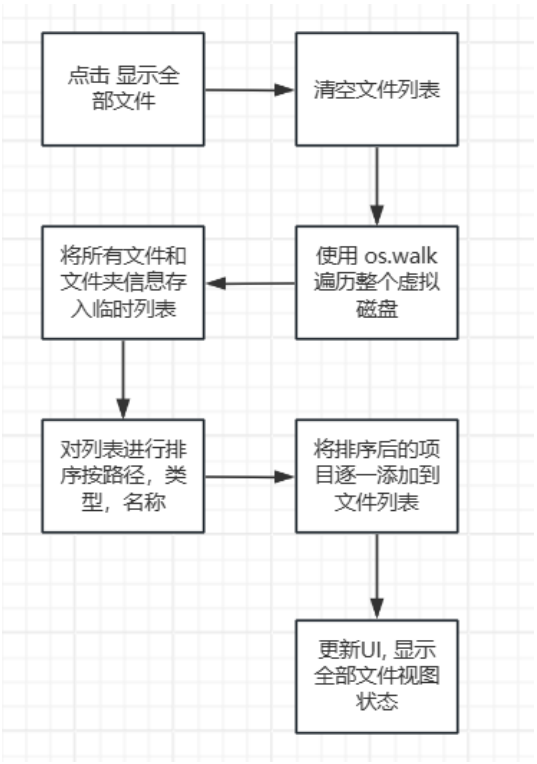
4.4. 文件夹操作流程



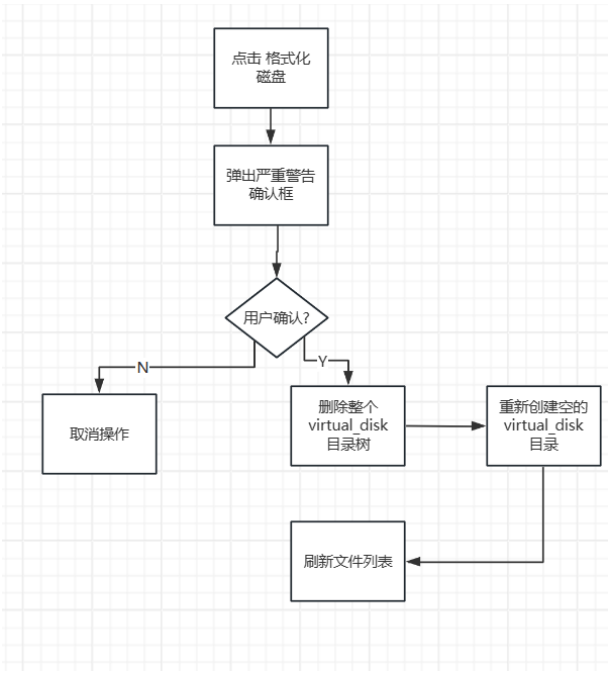
4.5. 文本编辑器流程



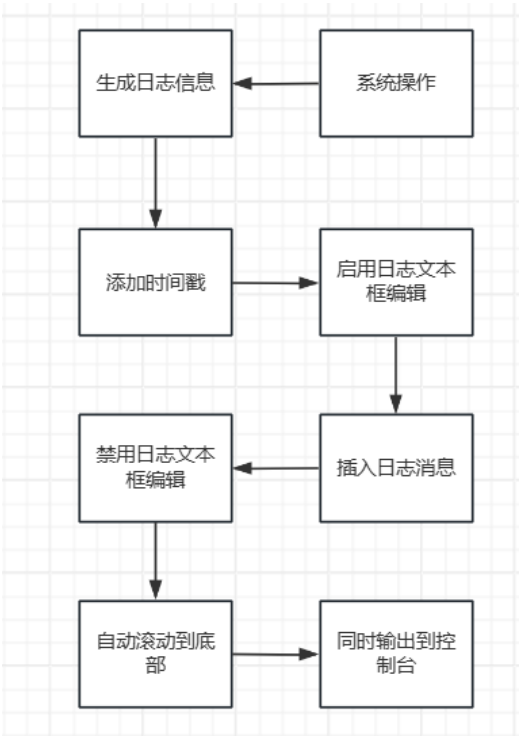
4.6 显示全部文件流程



4.7 格式化磁盘流程

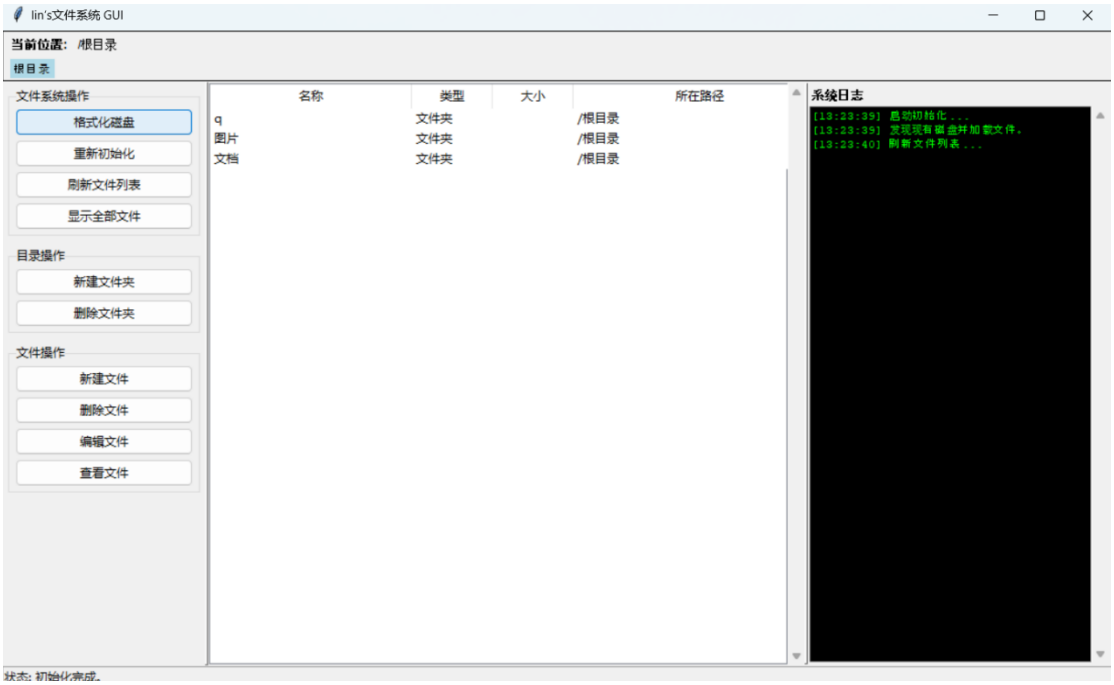


4.8. 日志系统流程

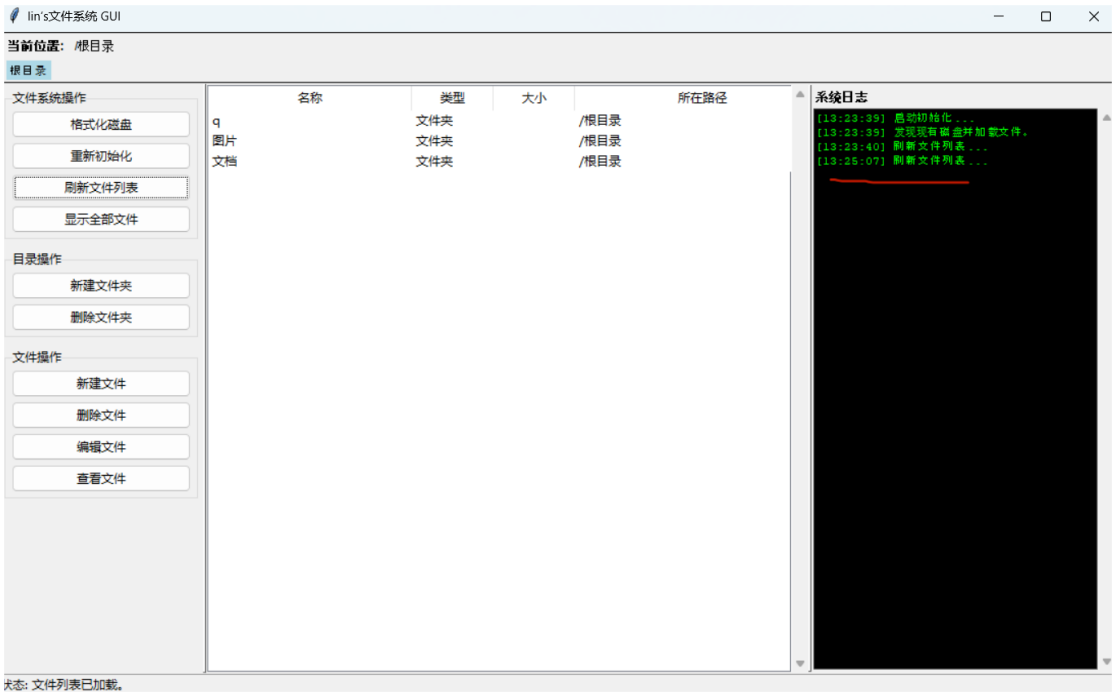


五．程序运行及清单

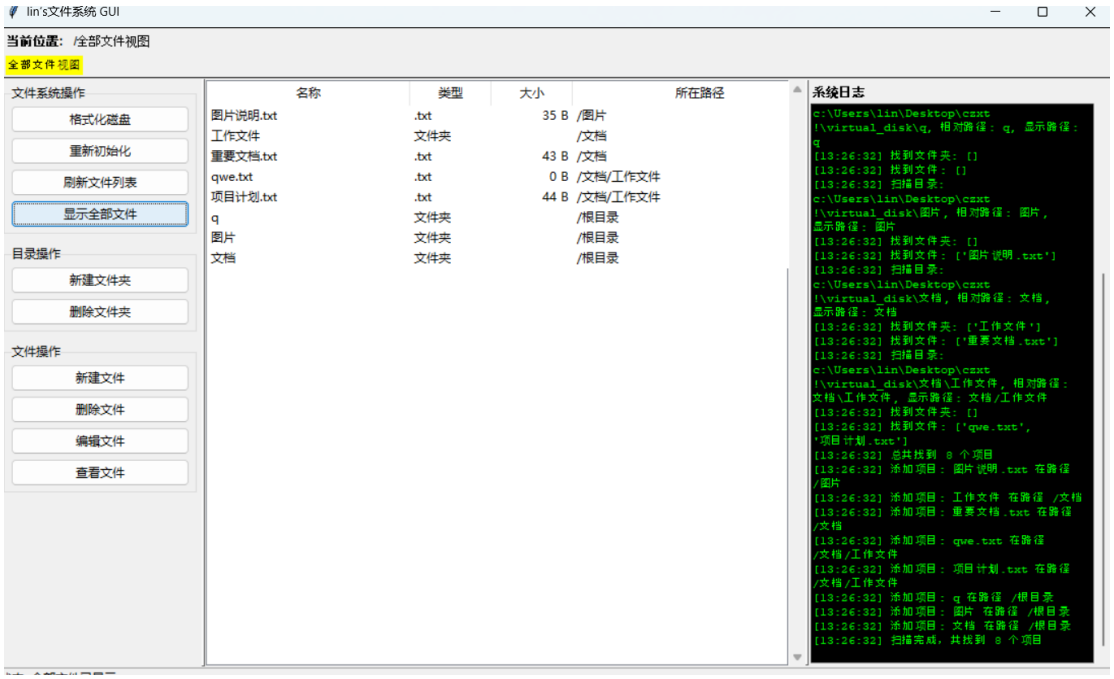
5.1 初始化界面（文件列表中是我初始化的文件）：



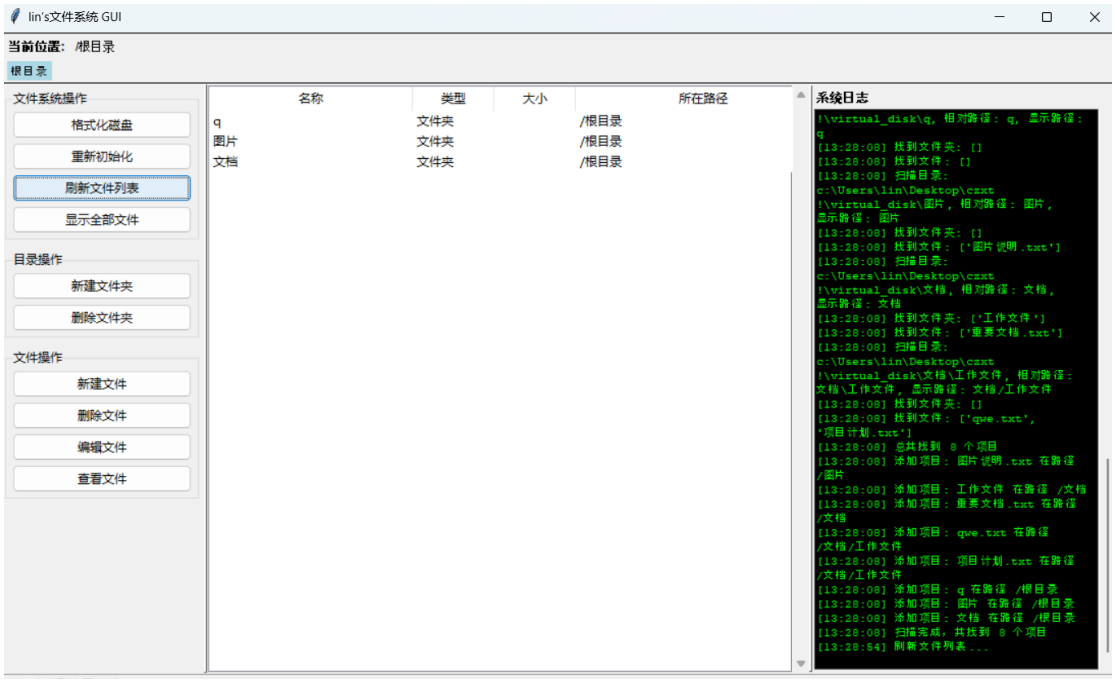
5.2 点击刷新文本列表（操作日志上会有提示）：



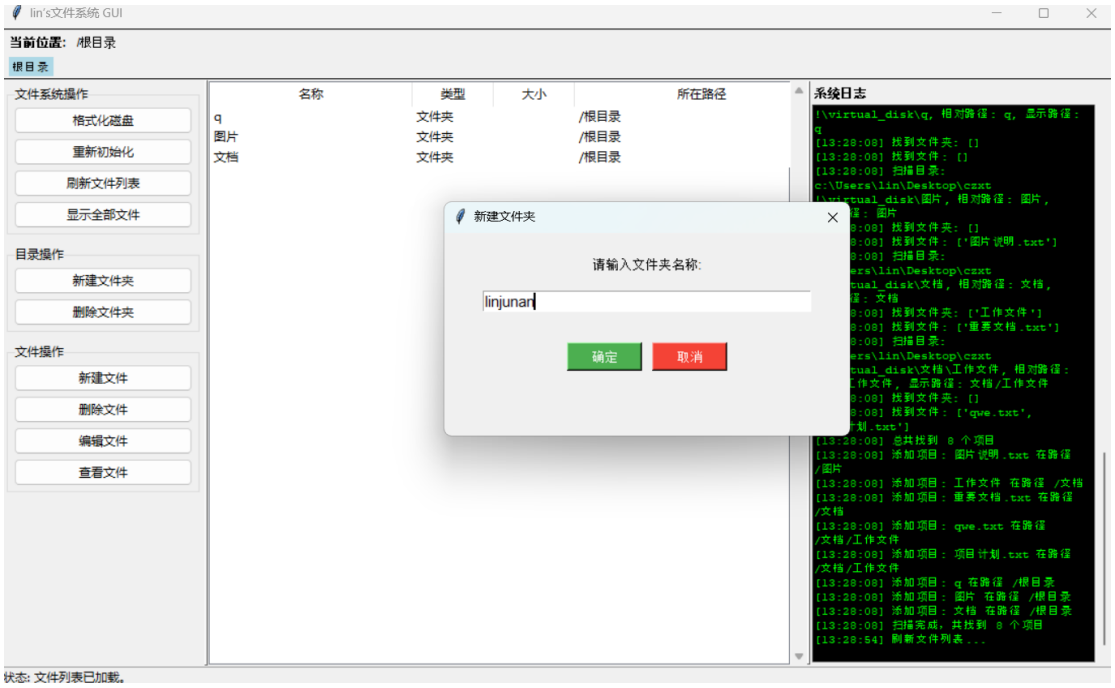
5.3 点击显示全部文件（会显示所有文件和其位置）



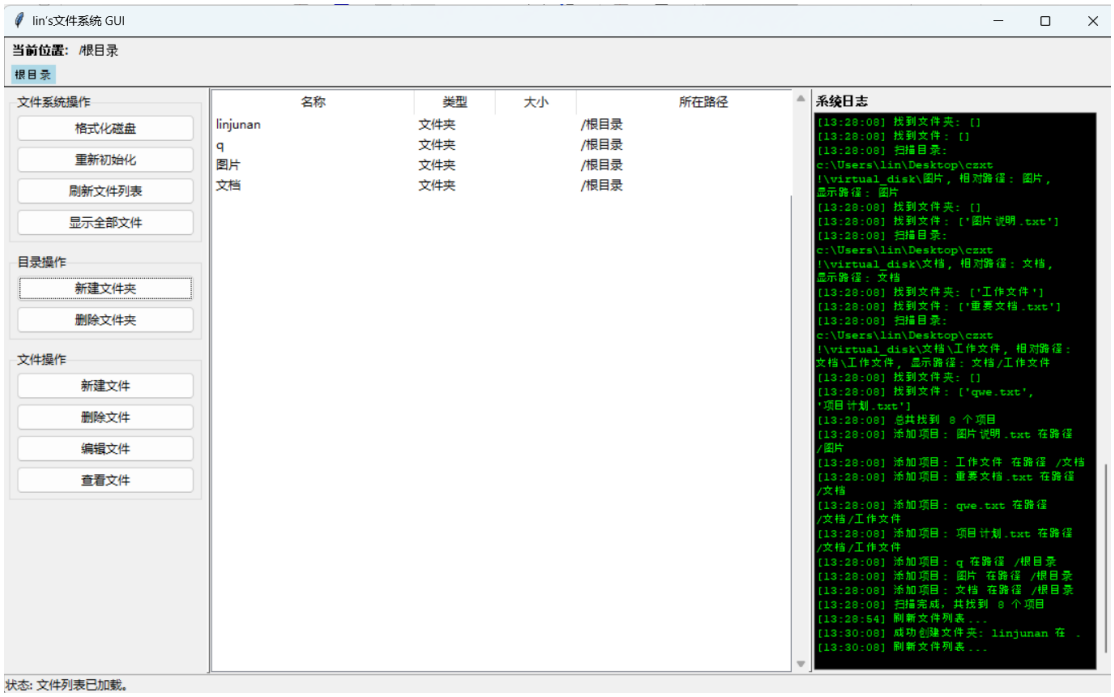
5.4 点击刷新文件按列表（可以取消显示全部文件）



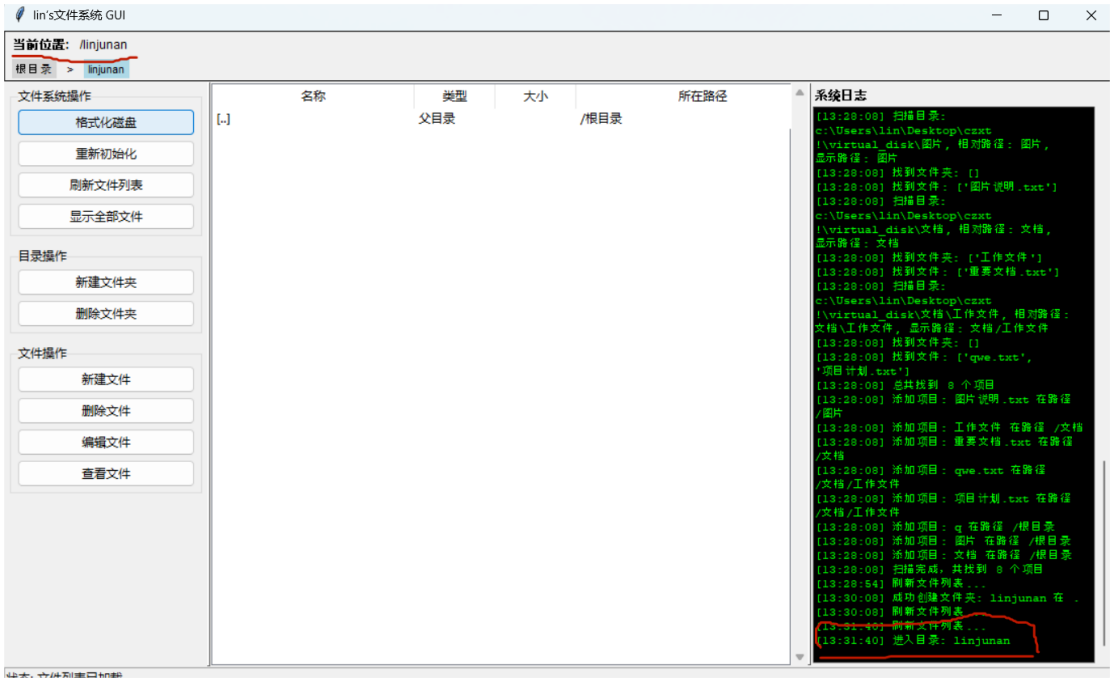
5.5 点击新建文件夹：



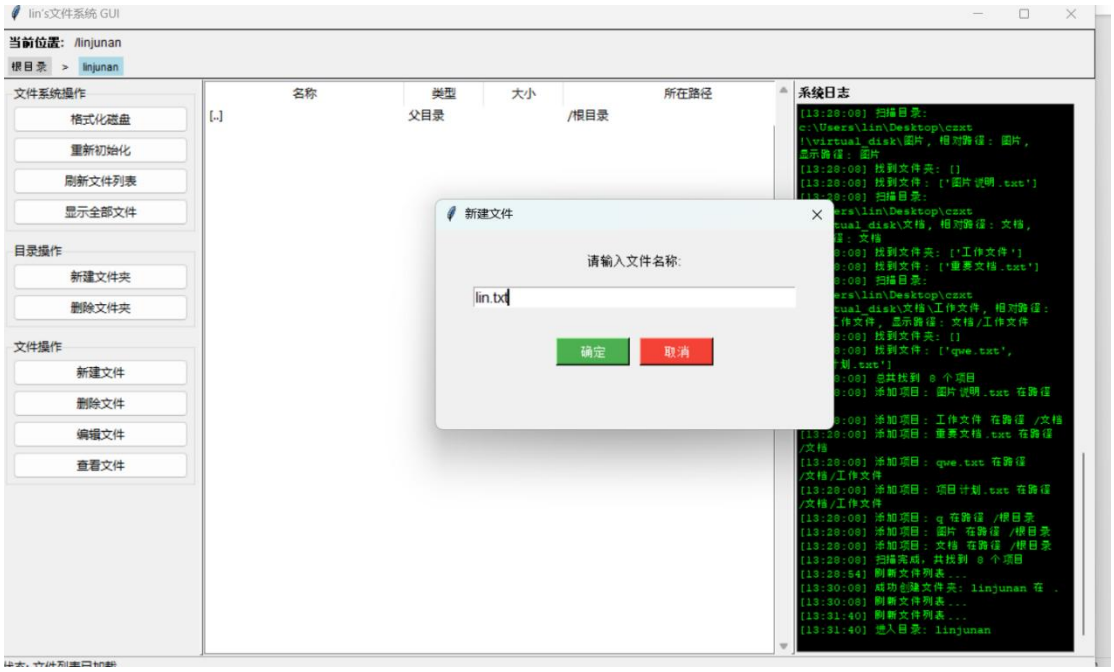
5.6 可以看到新建的文件显示在列表中：



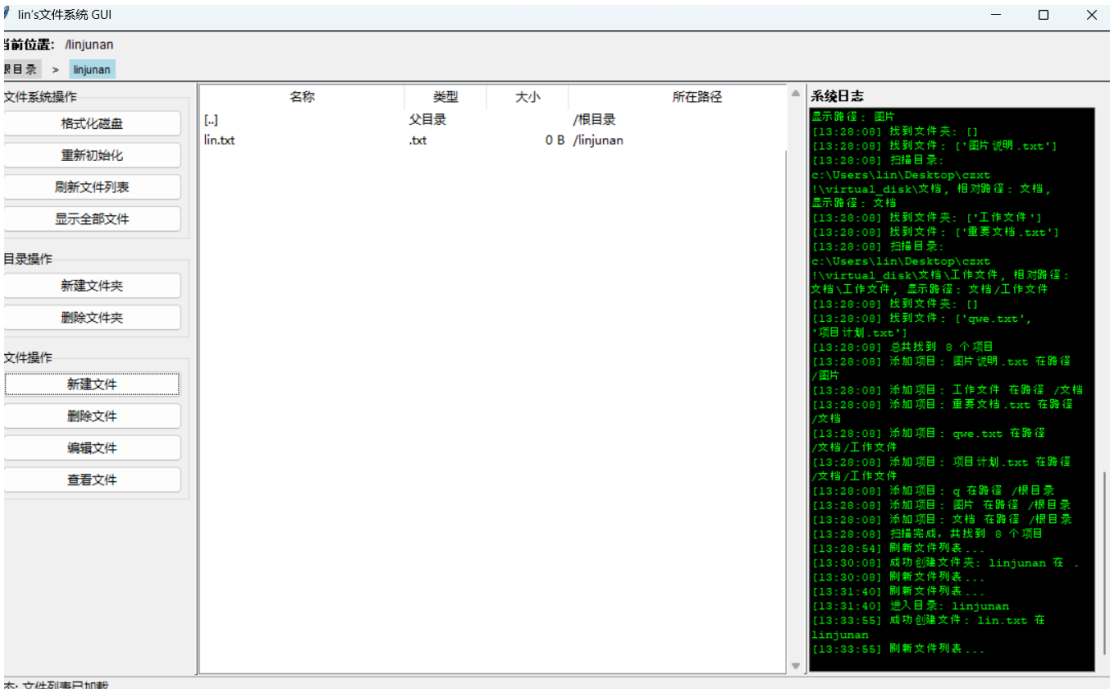
5.7 双击打开刚刚新建的文件夹（可以在左上角看到打开文件的路径，操作日志也会显示你的相关操作）：



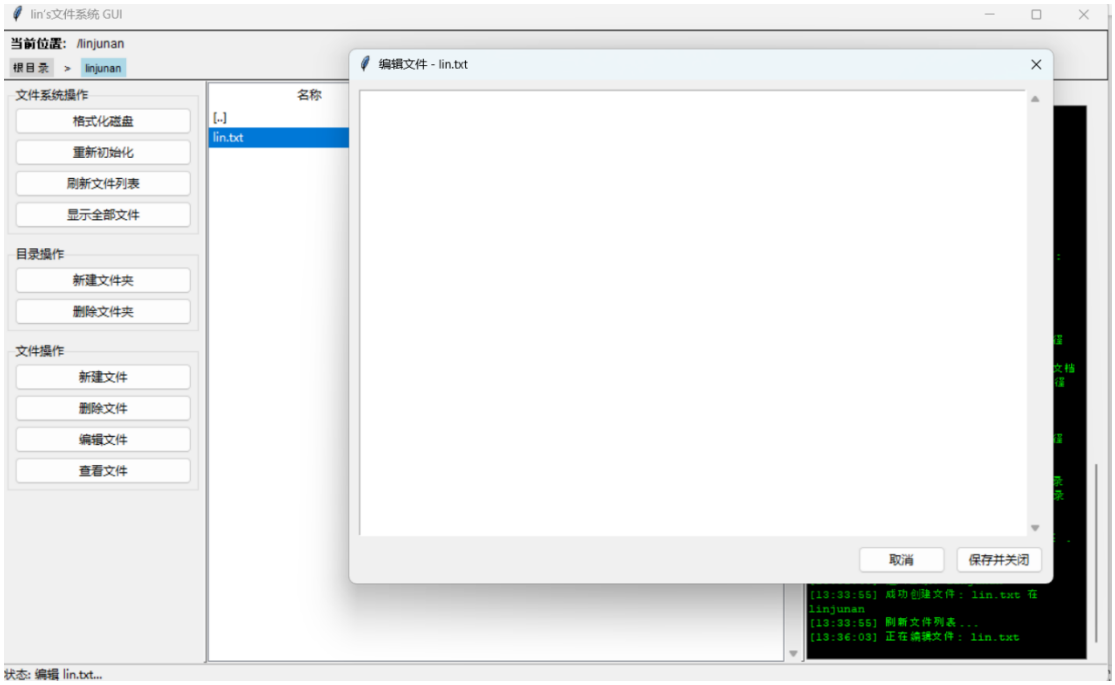
5.8 点击新建文件：



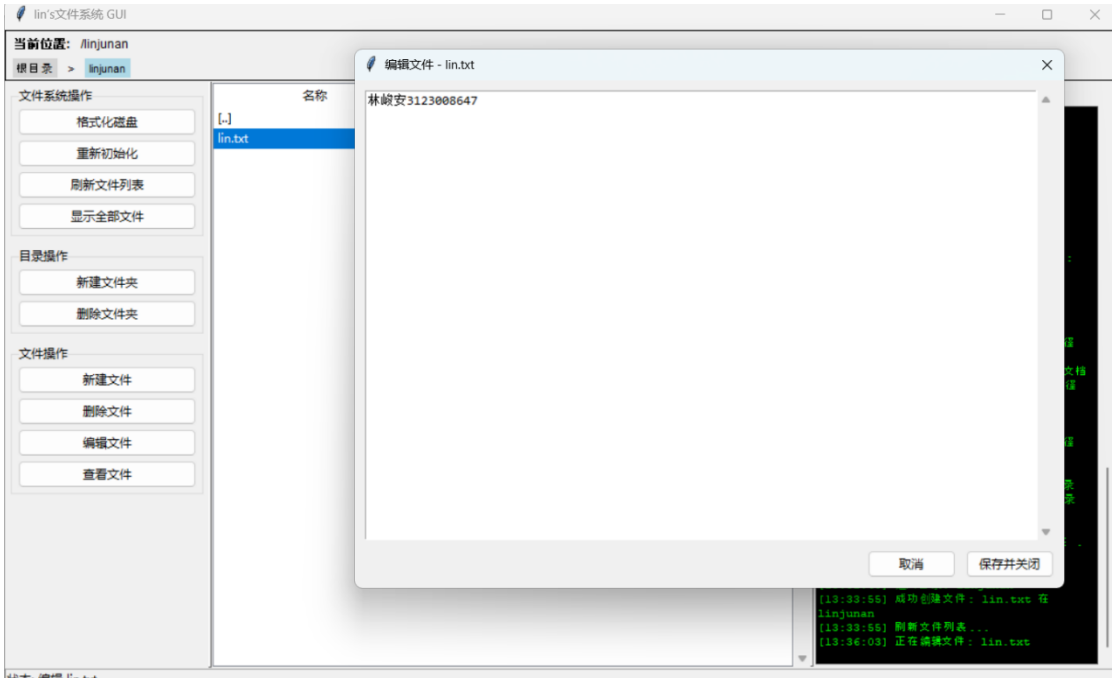
5.9 可以看到新建文件的大小和路径：



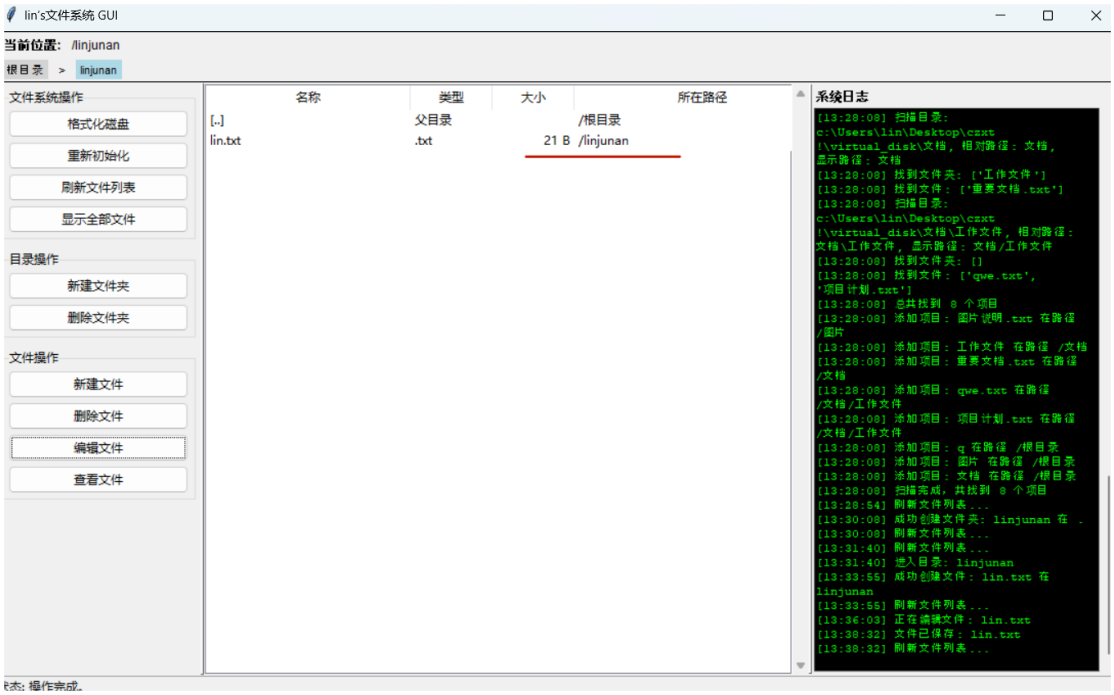
5.10 现在点击编辑刚刚新建的文本（要先点击该文本文件再点击编辑文件，同时会显示一个弹窗）：



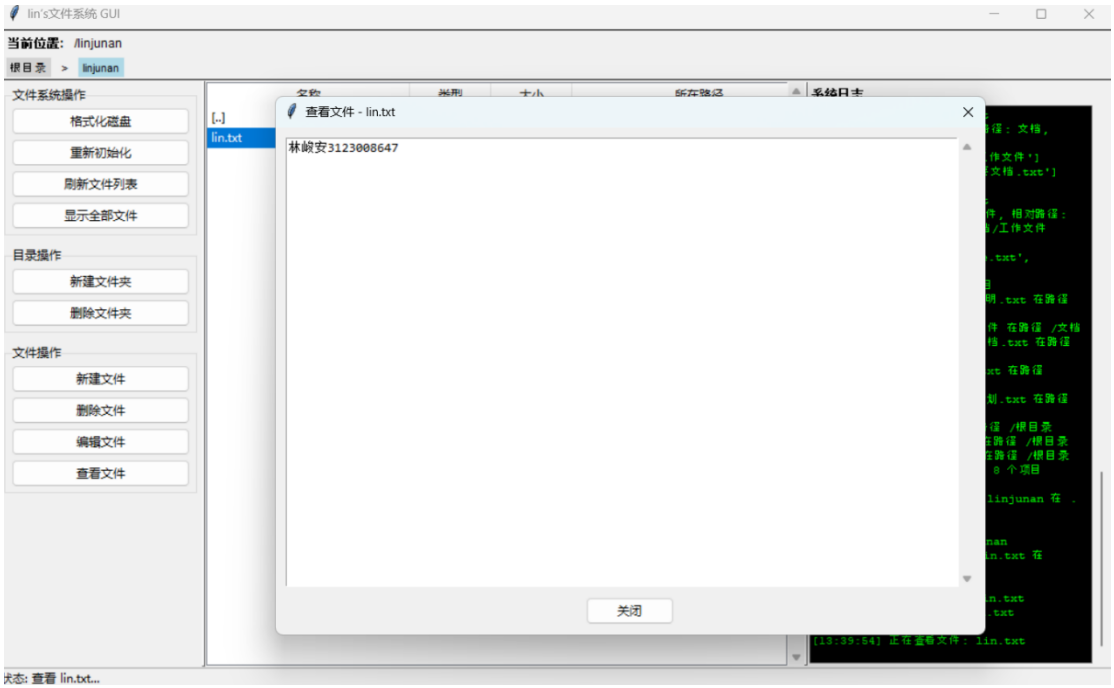
5.11 输入要编辑的信息：



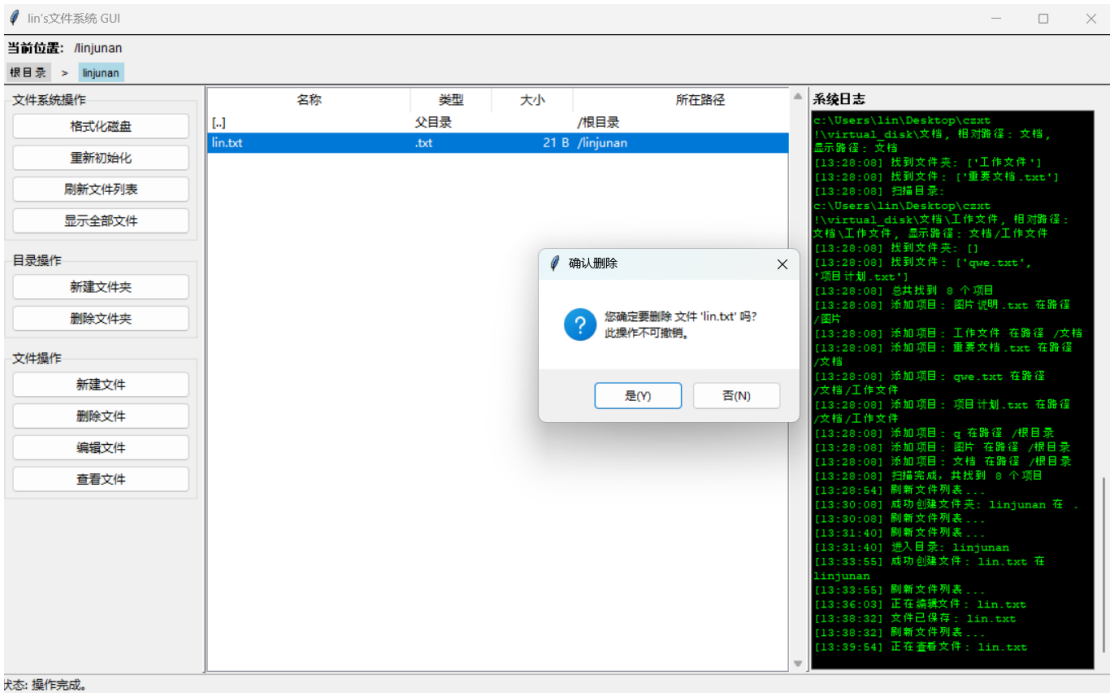
5.12 点击保存（可以看到这个文件的大小以及发生了变化）：



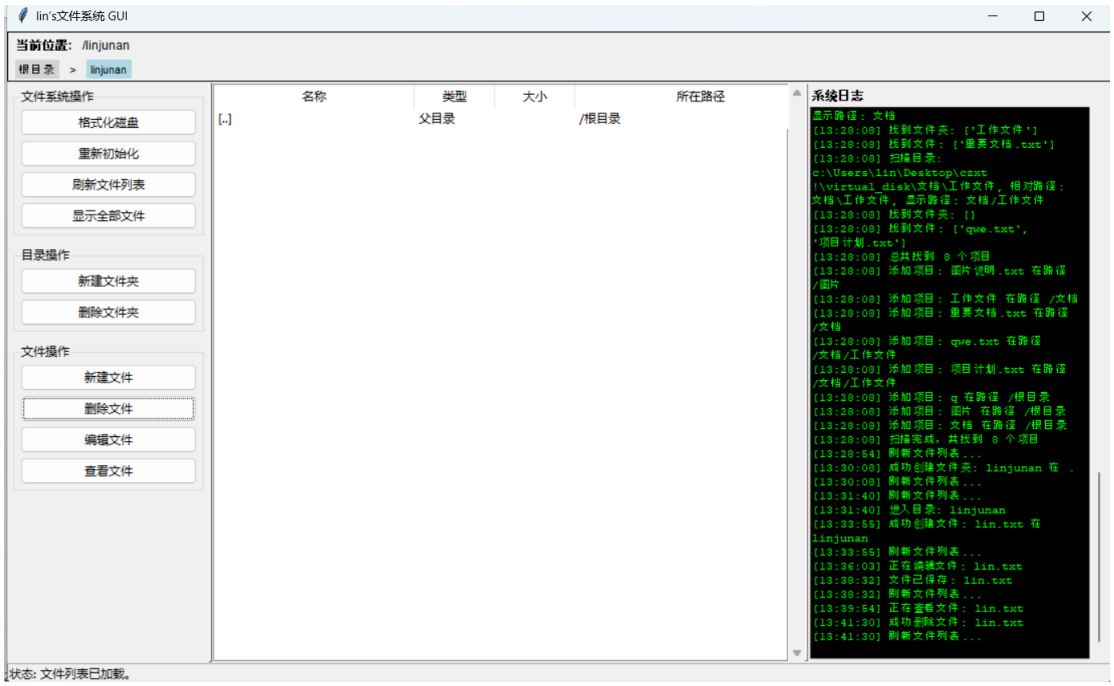
5.13 选中该文件，点击查看文件（只读）：



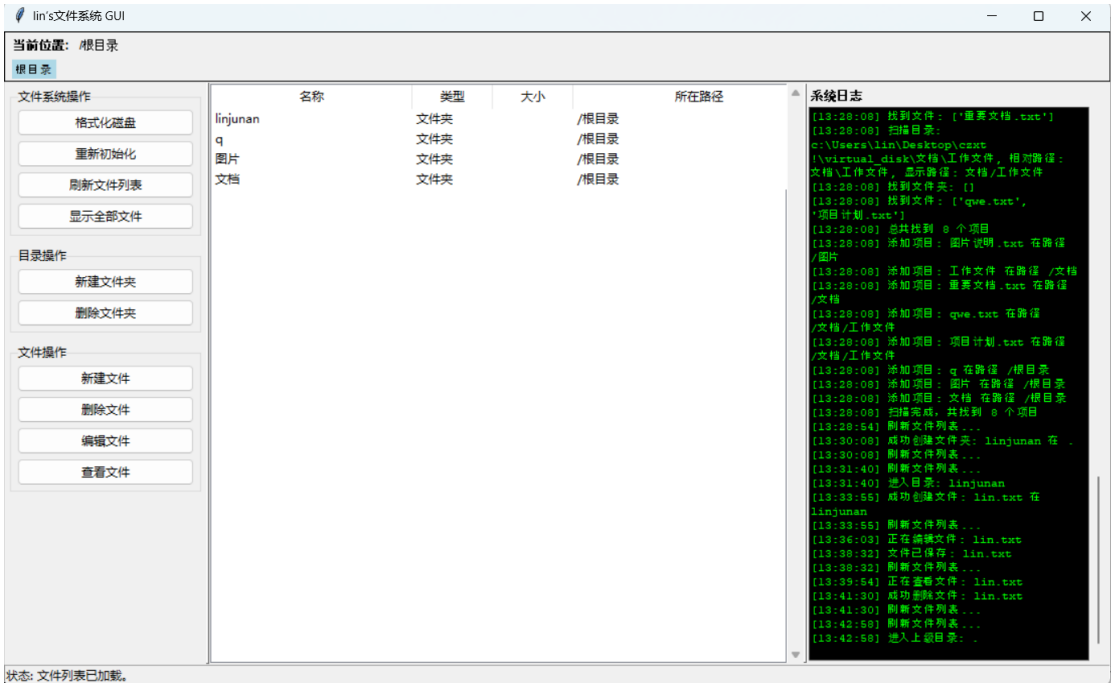
5.14 选中该文件，现在测试删除文件（会有提示信息）：



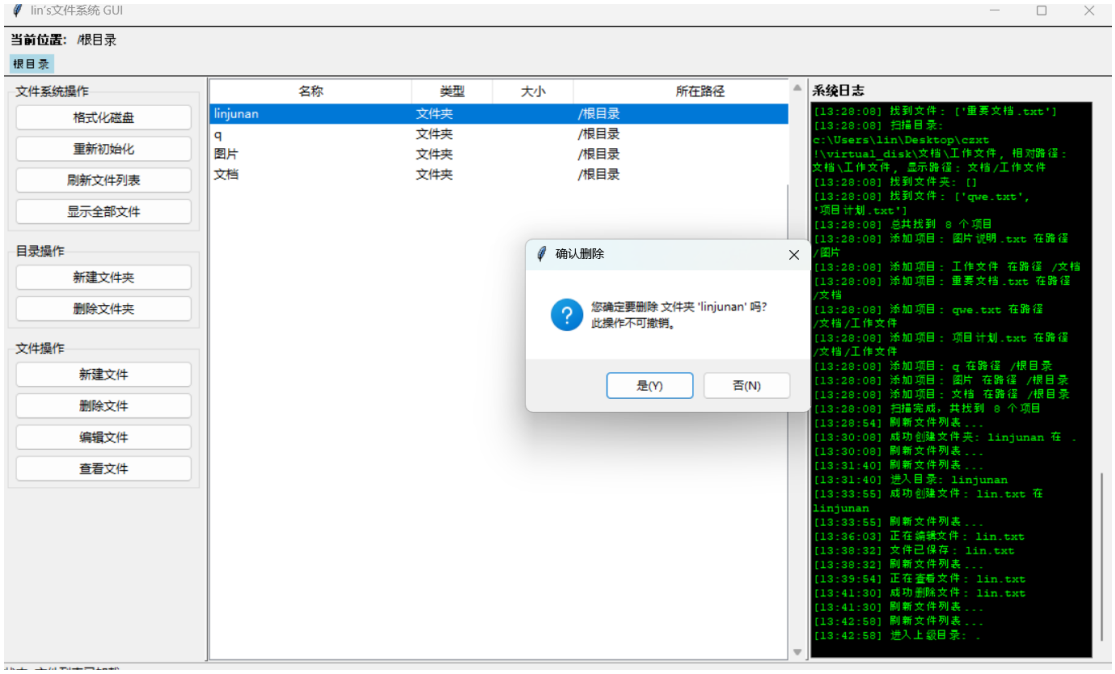
5.15 删除结果：

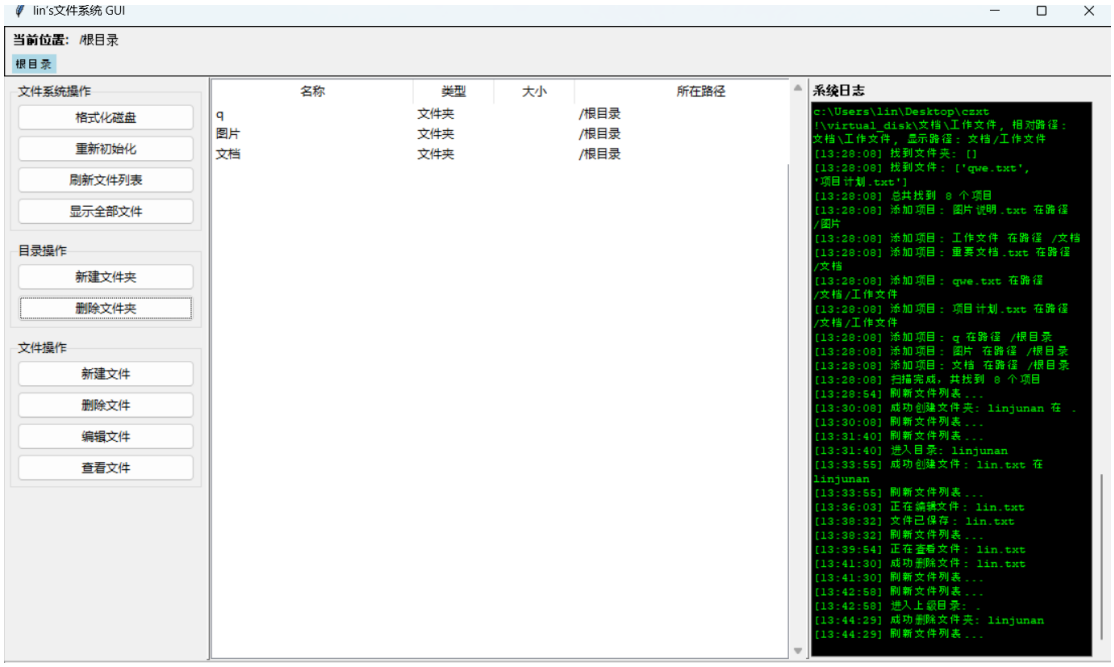


5.16 现在可以点击文件列表中最上面的【..】(可以返回到上一级文件夹)：

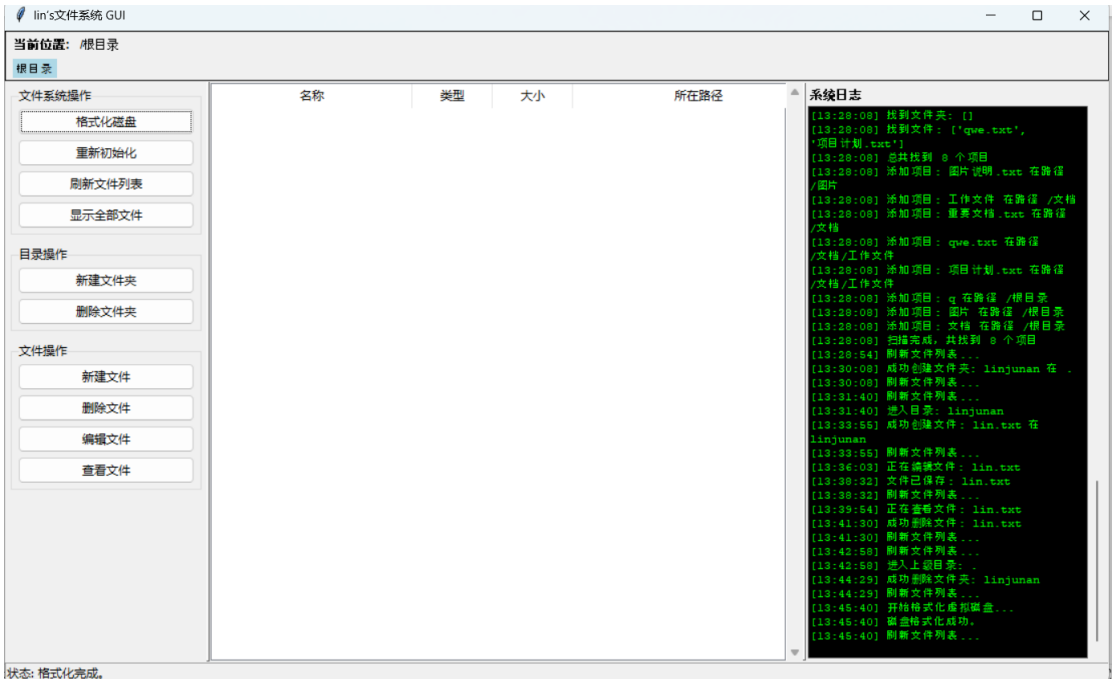


5.17 现在选中要删除的文件夹，删除 linjunan 这个文件夹：





5.18 最后点击格式化磁盘（会将里面的所有文件都删除）：



代码部分：

六. 使用说明书 .

6.1.界面说明：

- 顶部：当前路径 + 导航路径
- 左侧：操作按钮面板
- 中间：文件列表
- 右侧：系统日志
- 底部：状态栏

6.2 基本操作：

- 新建文件：点击"新建文件" → 输入文件名
- 新建文件夹：点击"新建文件夹" → 输入文件夹名
- 删除：选中文件/文件夹 → 点击"删除文件"或"删除文件夹"
- 编辑文件：选中文件 → 点击"编辑文件"
- 查看文件：选中文件 → 点击"查看文件"

6.3 导航：

- 进入文件夹：双击文件夹
- 返回上级：双击 [..]
- 面包屑导航：点击路径中任意部分快速跳转
- 查看全部文件：点击"显示全部文件"

6.4 系统操作：

- 刷新：点击"刷新文件列表"
- 格式化：点击"格式化磁盘"（清空所有文件）
- 重新初始化：点击"重新初始化"

6.5 注意事项：

- 所有文件操作在 virtual_disk 文件夹内进行

- 文件名不能包含特殊字符：<>:"/\|?*
- 删除和格式化操作不可撤销
- 在"全部文件视图"下无法创建文件/文件夹

6.6 小提示：

- 双击文件可直接查看内容
- 右侧日志实时显示操作记录
- 支持中文文件名和文件内容

七．体会，建议 ．

这次操作系统课程设计，我选择并完成了“文件系统 GUI 设计与实现”这个题目。回首整个过程，从最初拿到题目时的一知半解，到最终看着功能完善的图形化界面顺利运行时，心中充满了成就感。这不仅仅是一次简单的编程任务，更是一次将课堂上学习的抽象理论转化为实际应用的宝贵经历。

在学习操作系统课程时，文件系统、目录树、文件控制块、路径解析等概念虽然在书本上都学过，但总感觉有些遥远和抽象。通过这次课程设计，我才真正亲手“触摸”到了这些概念。例如，为了实现虚拟磁盘的功能，我用本地的一个文件夹来模拟，每一次创建文件、删除文件夹，实际上都是在调用 os 或 shutil 库对这个“虚拟磁盘”进行真实的操作。特别是路径管理和导航部分的设计，让我对绝对路径和相对路径有了更深的理解。在实现路径导航功能时 ，我需要将当前路径进行分割 ，为路径的每一部分都创建一个可以点击的按钮 ，点击后又能正确地跳转回对应的父目录。这个过程，就是对操作系统路径解析过程的一次完整模拟，远比看书本上的流程图要来得生动和深刻。

在开发过程中，自然也遇到了不少预料之中和意料之外的难题;

1.界面卡顿与响应迟钝问题：

- 问题描述：项目初期，我发现当执行一些操作，比如点击“格式化磁盘”或者在“显示全部文件”模式下刷新时，程序界面会短暂地“假死” ，按钮点了没反应。
- 分析与解决：经过查阅资料和分析，我意识到这是因为文件操作（特别是递归扫描或删除大量文件）是耗时操作，而我的程序把它放在了 Tkinter 的 UI 主线程里执行。这导致了主线程阻塞，无法处理界面的刷新和响应。虽然在这次课程设计中，因为时间关系没有引入复杂的多线程或异步 I/O 来彻底解决，但这个问题的出现让我明白了 GUI 编程中保持 UI 线程流畅的重要性，。

2.文件与文件夹的删除失败：

- 问题描述：在测试时，我发现有时删除文件或文件夹会失败 。比如，我用“查看文件”功能打开一个文本后，没有关闭查看窗口就去删除它，程序就会报错。

- 分析与解决：我很快定位到这是因为文件被我的程序自身占用了。当“查看文件”或“编辑文件”功能打开一个文件时，操作系统会将其标记为“使用中”，此时删除操作自然会被拒绝。我的解决方法是在执行删除操作前，程序能够检查并提示用户，更重要的是，在代码逻辑中确保对文件的读写操作完成后，能及时、正确地关闭文件句柄，释放资源。

3.“显示全部文件”视图下的路径显示错误：

- 问题描述：我的“显示全部文件”功能，最初的设计是扫描并列出虚拟磁盘中的所有项目。但在早期的版本里，无论文件藏在多深的子目录里，它的“所在路径”都错误地显示为根目录 。

- 分析与解决：这是一个典型的逻辑错误。问题出在我遍历目录树的函数中。我只获取了文件名，而没有正确地拼接它所在的相对路径。为了修复它，我利用 os.walk() 在递归遍历时返回的父目录路径，为每一个文件和文件夹都生成了正确的、相对于虚拟磁盘根目录的显示路径。

当然通过这次课程设计，我不仅巩固了操作系统的知识，还锻炼了 Python 和 Tkinter 的编程能力，以及软件工程中从需求分析、系统设计到编码测试的整个流程。虽然我实现的文件管理器已经具备了核心功能 ，但仍有很多可以改进的地方,例如：引入多线程来处理耗时 I/O，防止界面卡顿，增加文件的复制、移动、重命名等常用功能，甚至可以加入简单的文件搜索,或者可以为不同文件类型设置不同图标，增加右键菜单，让交互更加自然等等。总而言之，这次课程设计是一次非常有价值的实践。

八．参考文献

1.https://blog.csdn.net/liwenfei123/article/details/79485490

2.https://www.runoob.com/python/python-gui-tkinter.html

3.计算机操作系统（第四版）