同济大学

Tongji University

《操作系统》

项目报告III

报告名称：文件管理系统

班　　级：操作系统02班

学　　号：2253744

姓　　名：林觉凯

指导老师：张惠娟

完成日期：2024年6月10日

目录

1.项目介绍

1.1 项目目的

- 理解文件存储空间的管理；
- 掌握文件的物理结构、目录结构和文件操作；
- 实现简单文件系统管理；
- 加深文件系统实现过程的理解；

1.2 项目需求

- 基本需求

在内存中开辟一个空间作为文件存储器，在其上实现一个简单的文件系统;

退出这个文件系统时，需要该文件系统的内容保存到磁盘上，以便下次可以将其回复到内存中来。

- 具体要求

文件存储空间管理可采取链接结构(如 FAT 文件系统中的显式链接等)或者其他学过的方法；空闲空间管理可采用位图或者其他方法；文件目录采用多级目录结构，目录项目中应包含：文件名、物理地址、长度等信息。

文件系统提供的操作：

格式化、创建子目录、删除子目录、显示目录、更改当前目录、创建文件、打开文件、关闭文件、写文件、读文件、删除文件。

1.3 开发环境介绍

- 开发环境：Windows 11
- 开发软件：Visual Studio Code
- 开发语言：python 3.9.13 通过 conda 配置 python 环境，并且 pip 安装 PyQt5

1.4 项目结构介绍

- fileSystem.py——项目源代码
- fileSystem.exe——项目可执行问卷
- images——项目所需图片
- fat,disk,catalog——先前测试保存的数据

```
os_project3_fileSystem
|_disk
    |_images
    |_fileSystem.exe
    |_fat
    |_disk
    |_catalog
|_images
|_fileSystem.py
|_fat
|_disk
|_catalog
```

图 1.1 项目目录结构

2.项目运行

2.1 编译运行

若需要编译运行，需要安装 PyQt5。打开 Visual Studio Code，在终端命令行中输入 pip install PyQt5(如下图所示)。等待安装好 PyQt5 后，即可在 Visual Studio Code 中编译运行 fileSystem.py 并观察运行结果。

图 2.1 安装 PyQt5

2.2 直接运行

在 Windows 系统下打开文件夹 disk 即可运行，确保 images 文件夹和 fileSystem.exe 在同一目录下，这时直接打开 fileSystem.exe 即可运行。

3.项目设计

3.1 文件物理结构

链式结构是文件存储的一种物理结构，常用于存储不连续分配的文件。链式结构通过链表的方式将文件的各个部分链接起来，这样可以有效利用磁盘空间，但也有一些缺点和优点。

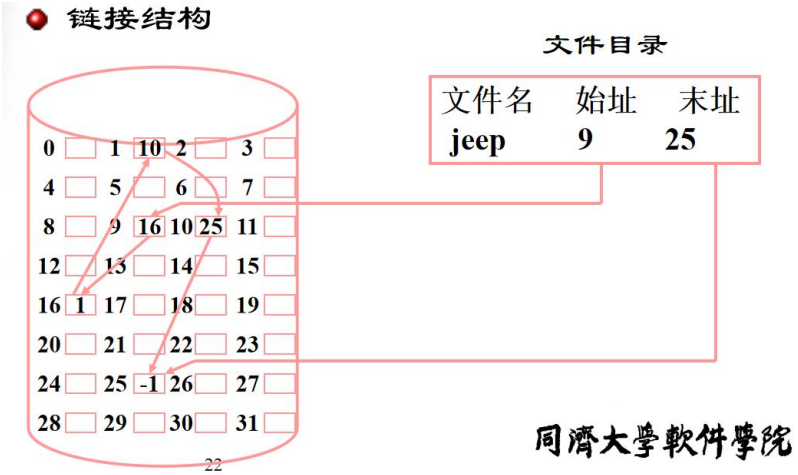

图 2.2 文件的链式结构

链式结构有以下特点：灵活的空间利用：不需要连续的存储空间，可以利用磁盘的碎片空间，从而提高磁盘空间的利用率。文件扩展方便：文件可以方便地增加新的块，不需要移动整个文件。易于实现文件扩展和收缩：链式结构使得增加或删除文件中的数据块变得容易。但是访问速度慢：由于需要通过链表指针逐个块进行访问，读取文件的速度较慢，特别是对于顺序访问的情况。额外存储开销：每个块需要存储指向下一个块的指针，这会占用额外的存储空间。可靠性较低：如果链表中的一个指针损坏，将导致文件无法继续读取，文件的部分数据可能会丢失。

3.2 文件存储空间和空闲空间

此次项目我们采用了链接结构来管理文件存储空间。每一个物理块对应一位,空闲物理块我们标志为-2，否则标志为-1 或者自然数。若标志为自然数，这种情况则表示是它链接的的是下一块的块号；若标志为-1，这种情况则表示后面没有链接的块，这一个物理块为最后一个块。

3.3 文件目录

此次项目我们采用了多级目录结构，Catalog 类代表了一个目录或者是一个文件。当 Catalog 是一个目录这种情况，它就有一个子结点列表；当 Catalog 是一个文件这种情况，它就有一个 FCB(文件控制块)来存储文件的元数据，包括文件名、创建时间和最后修改时间等等。文件控制块(File Control Block，FCB)是一种数据结构，用于存储与文件相关的信息，是操作系统管理文件的重要组成部分。FCB 类有一个指向 FAT 的引用，用于管理文件的物理块。文件分配表(File Allocation Table, FAT)是一个数组，用于记录每个簇的使用情况和链接信息。文件的存储位置通过 FAT 表来查找。

3.4 文件系统保存

在我们要退出文件系统时，我们需要决定是否将此次操作写入磁盘中，如果我们选择是，文件系统就会将此次操作所对应的 FAT 表、disk 磁盘和 catalog 目录结构序列化之后写入到本地的文件中。这样，当下次启动文件系统时，我们就可以从这些保存好的文件中读取先前的数据数据。

4.项目功能

4.1 新建文件/文件夹

在右侧空白处右键点击，选择“new”操作，便可以选择要创建的类型——文件或者文件夹，如下图所示。创建文件/文件夹后，设置文件的名称。

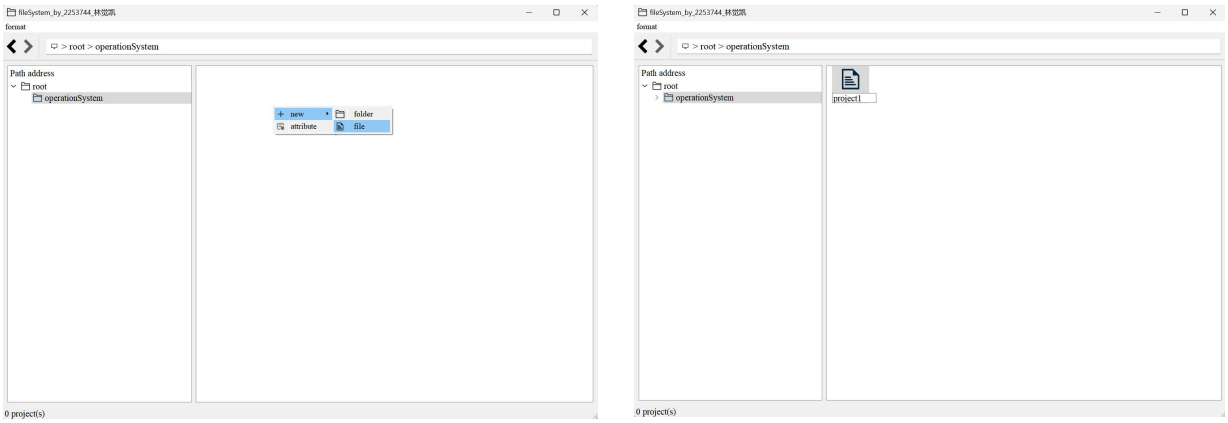

图 4.1 新建文件/文件夹

操作系统为它创建了一个新的列表项，表示新的文件，并将其添加到列表视图中。同时操作系统为新文件分配一个文件控制块(FCB)用于存储文件的元数据。这些元数据包括文件名、文件类型、创建时间、修改时间、访问权限等。然后，它创建了一个新的节点，表示新的文件，并将新的 CatalogNode 添加到目录表中。，最后更新树形地址。

4.2 打开文件/文件夹

我们可以通过两种方法来打开文件/文件夹，第一是双击要打开的文件/文件夹，第二是右击要打开的文件/文件夹，选择“open”操作即可打开。

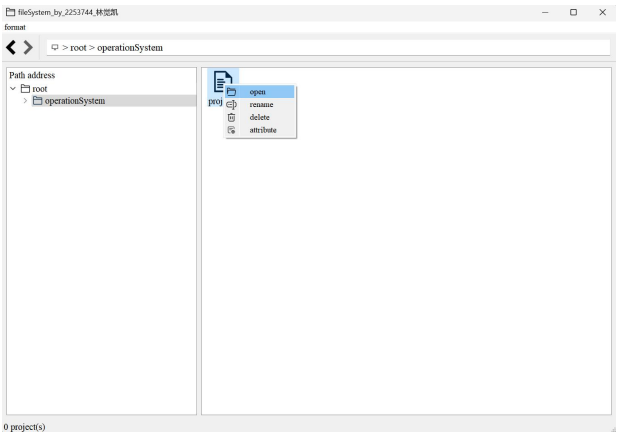

图 4.2 打开文件/文件夹

操作系统从根目录开始，逐层解析路径中的各个部分。逐个搜索每一级目录的目录项，以找到下一级目录或最终的文件。在找到文件的目录项后，最后进入相应的文件/文件夹，同时所在目录路径地址也进行更新。

4.3 重命名文件/文件夹

如下图所示，选择某一个文件/文件夹，右击鼠标，选择“rename”操作，即可对选定文件/文件夹进行重命名操作。

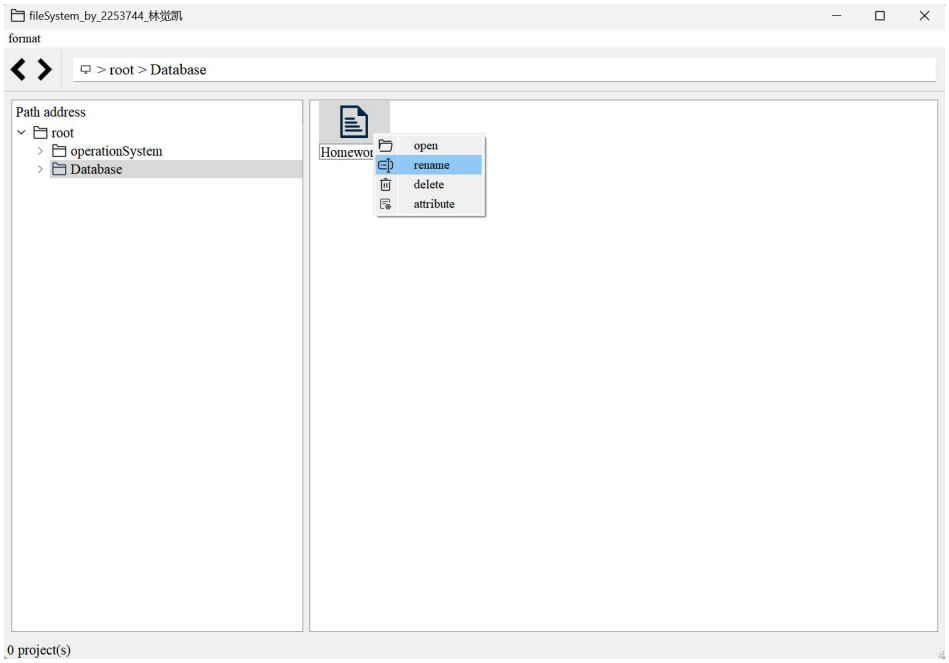

图 4.3 重命名文件/文件夹

在源目录中删除旧的目录项，同时在目标目录中添加新的目录项，并更新相关元数据。更新内存中的文件系统数据结构，确保新的名称和路径在内存中也是最新的。对于已打开的文件或文件夹，需要更新内存中的文件描述符和相关信息。

4.4 删除文件/文件夹

如下图所示，选择某一个文件/文件夹，右击鼠标，选择“delete”操作，即可对选定文件/文件夹进行删除操作。

删除操作首先会检查用户是否选择了一个文件或文件夹，如果没有选择，操作会被终止。这时，文件系统会弹出一个提示框询问用户是否确定要删除选中的文件或文件夹。如果用户选择 cancel，那么删除操作就会被终止。删除的具体流程为从列表视图中删除选中的文件项、删除文件项对象、递归地删除选中文件项

的子文件或子文件夹、从当前节点的子节点列表中删除选中的文件项，最后更新目录结构表和树形视图。

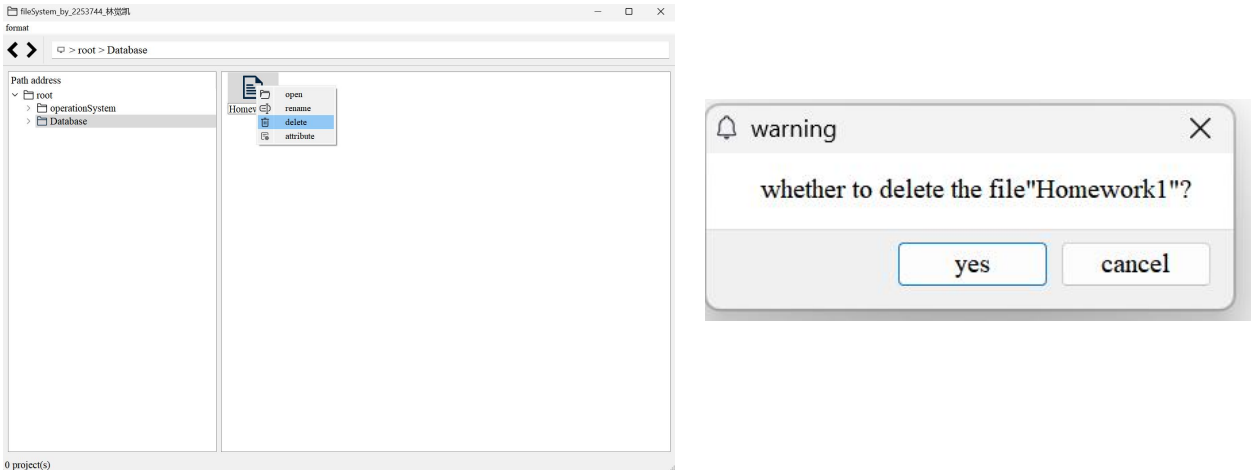

图 4.4 删除文件/文件夹

4.5 查看文件/文件夹属性

如下图所示，选择某一个文件/文件夹，右击鼠标，选择“attribute”操作，即可对选定文件/文件夹进行查看属性操作。查看属性的内容有类型、文件/文件夹名称、创建时间和更新时间等等，如果是文件夹的话还会显示该文件夹下有多少个项目。

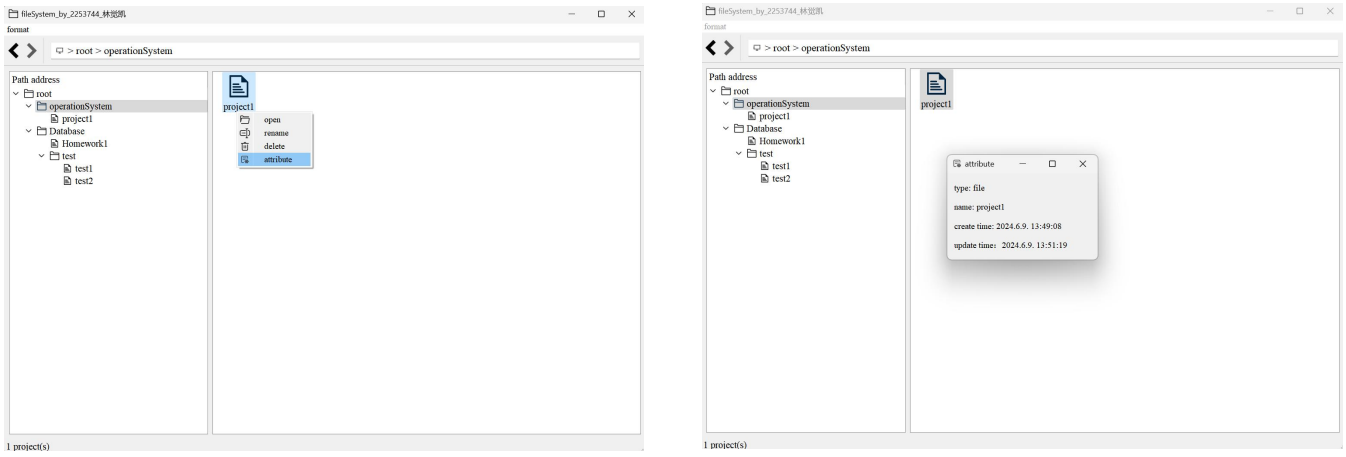

图 4.5 查看文件/文件夹属性

在初始化每一个文件/文件夹时，会保存一些关于该文件/文件夹的参数，例如文件/文件夹的名称、是否为文件、创建时间、更新时间和子节点数量等等。并且将这些信息保存到文件/文件夹属性的这个类中。每当我们调用查看属性的函数时，就会显示出相应的窗口进行展示。

4.6 输入文件内容

一个文件可以用来保存文本信息。如下图所示，选择某一个文件，右击鼠标，选择“open”操作，或者双击选中的文件，便可以打开该文件，并且在文件中输入内文本内容。

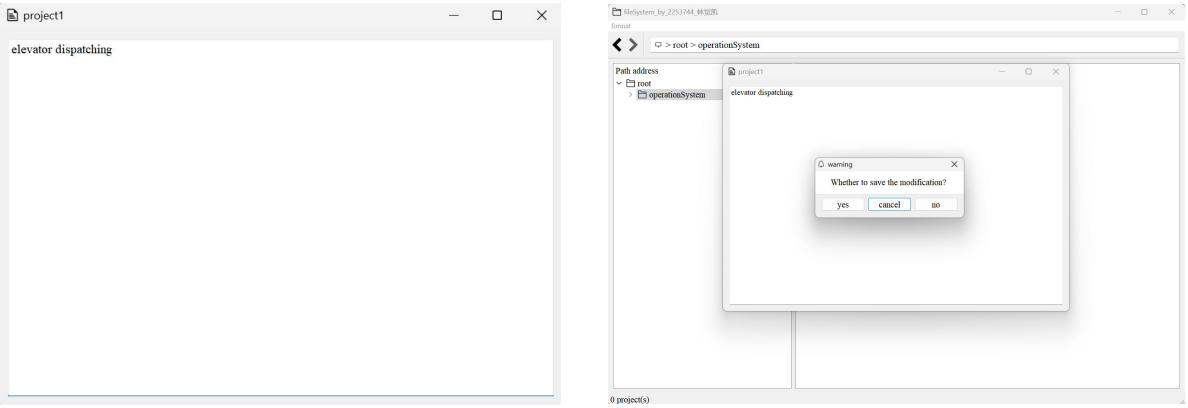

图 4.6 输入文本内容操作

当输入完文本内容要返回时，会跳出来一个询问框，是否要保存该文本文件。如果用户选择了保存，那么文件内容 data 就会保存在文件系统中，下次打开文件系统时先前的数据就会显示出来。

4.7 格式化磁盘

在顶部有一个”格式化”选项，点击之后，给出提示确定是否格式化，如果选择要格式化，将会进行格式化操作。

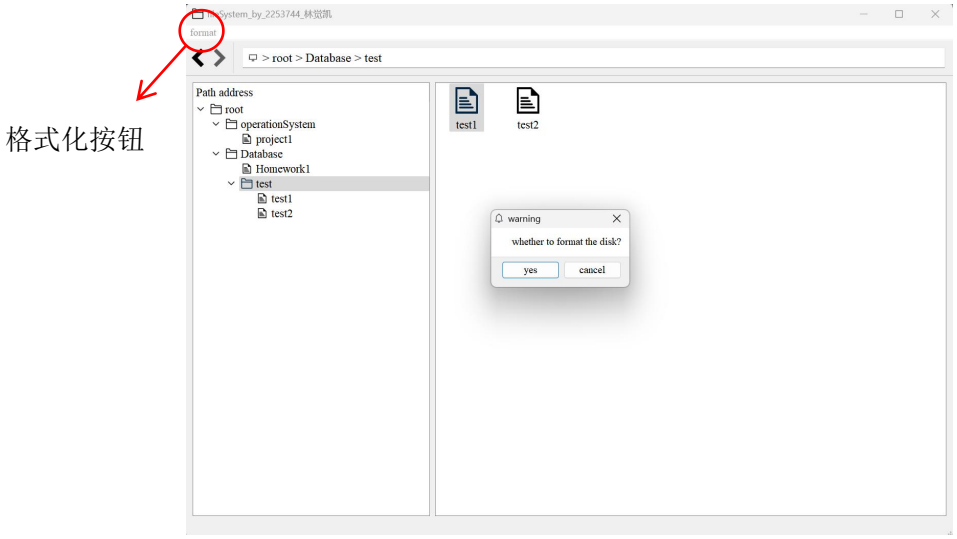

图 4.7 格式化磁盘

格式化会结束任何正在进行的编辑操作，同时文件系统创建一个新的 FAT、一个新的 disk 和一个新的 catalog，然后将它们分别写入到对应的文件中，完成文件系统的格式化。最后，系统会隐藏当前窗口，并且创建一个窗口对象并显示。

4.8 查看目录

用户可以在右侧的 path address 和上方的窗口顶部看到当前的目录和路径。

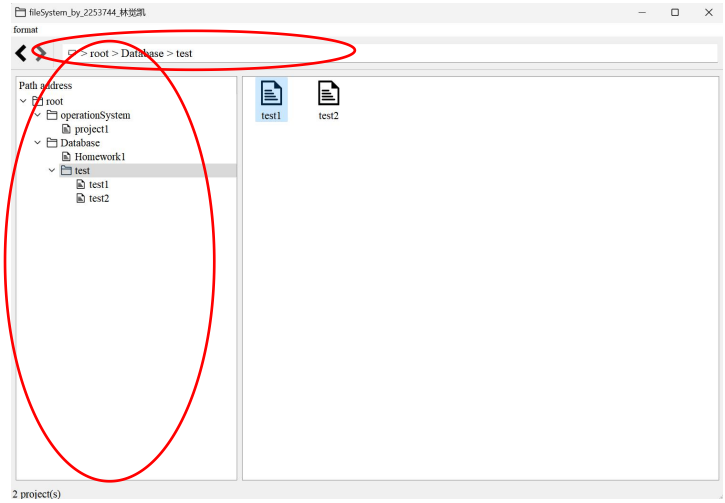

图 4.8 查看目录

在左侧的导航栏是一个地址树部件，它展示了文件系统的目录结构。用户可以通过点击树中的项来导航到不同的目录。目录可以展开、收起。同时上方的地址栏给用户以直观的显示当前所在的目录。

4.9 写入磁盘操作

如下图所示，在用户退出文件系统的时候，可以决定是否将本次操作写入磁盘中，如果写入，本次操作的数据就会保存在相应的文件中，并且在下一次打开的时候显示先前的数据内容。

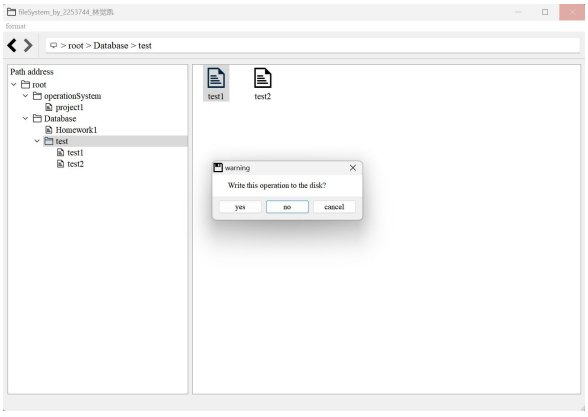

图 4.9 写入磁盘操作

在我们要退出文件系统时，我们需要决定是否将此次操作写入磁盘中，如果我们选择是，文件系统就会将此次操作所对应的 FAT 表、disk 磁盘和 catalog 目录结构序列化之后写入到本地的文件中。这样，当下次启动文件系统时，我们就可以从这些保存好的文件中读取先前的数据数据。

4.10 前进/返回操作

在文件系统的左上角分别有有个前进键和返回键,点击这两个按键既可以实现前进/返回操作下一级/上一级目录的操作。(可以点击这两个按键的时候按键的颜色会比原来变深一点)

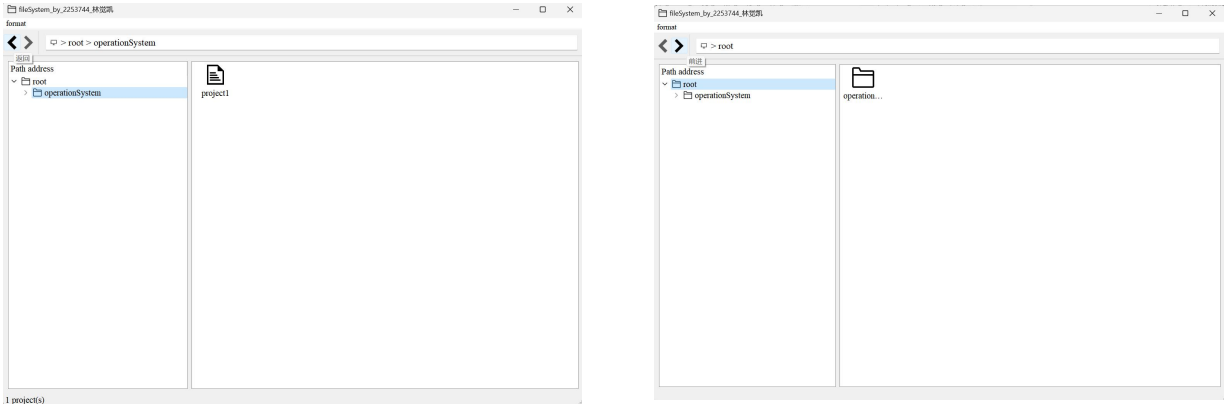

图 4.6 前进/返回操作

注意到如果当前节点是根节点，那么返回的按键将会被禁止使用;如果目前所在的节点之前有访问到它的子节点,即有记录它先前子节点的位置,这个时候前进按键将可以被使用.

5.项目总结

5.1 项目问题和改进

此次项目中主要遇到的问题是文件链式存储的表示代码编写问题。最后在学习借鉴资料和参考之后,最后成功完成了代码的编写。本次项目由于临近期末,时间比较少,项目实现的文件系统的功能较少,只实现了一些文件系统基本的功能,对比 windows 的文件管理系统还有查看、排序、撤销、分类、分组等等更多的功能选择,该文件管理系统实现的功能还是较少。未来项目的改进方向可以是更多的功能选择、更好看的系统界面等等。

5.2 项目心得与收获

本次项目有了第一次项目使用 PyQt5 编写代码的基础,这次使用 PyQt5 就显得更加熟练。我认为此次项目我获得了以下收获,第一,对 python 这门语言掌握地更加熟练。这次项目使用 python 编写,调用的 python 中的很多函数库,同时前端 PyQt5 的多次使用应用,这让我对语言的编写更加熟练;第二就是对书上内容的再次巩固。文件系统是一个比较有趣的项目,我们实际上自己制作了有个文件系统,可以在自己的文件系统中进行一系列操作,做完之后非常有成就感和体验感。同时,对于书上操作系统文件管理这部分的知识内容有了更加清晰的了解,在建立很好的理解之后,我们才能顺利地完成这部分代码的编写。临近期末,这也让我复习了文件管理系统部分的内容知识。希望在之后有空的时候可以再次完善我的文件管理系统,添加更多的功能,做出一个更好的系统界面。