BIBA 模型实现

一、实验目的

1.了解 BIBA 模型的相关知识
2.通过编程实现基于 BIBA 模型的完整性访问控制机制

二、实验原理

1.BIBA 模型的完整性定义

完整性级别高的实体对完整性低的实体具有完全的支配性，反之如果一个实体对另一个实体具有完全的控制权，说明前者完整性级别更高，这里的实体既可以是主体也可以是客体。完整性级别和可信度有密切的关系，完整级别越高，意味着可信度越高。

2.BIBA 模型的规则

对于写和执行操作，有如下规则：

a.写规则控制：当且仅当主体 S 的完整性级别大于或等于客体 O 的完整性级别时，主体 S 可以写客体 O,一般称之为上写。

b.执行规则控制：当且仅当主体 S2 的完整性级别高于或等于 S1,主体 S1 可以执行主体 S2。

关于读操作，通过定义不同的规则，读规则有不同的控制策略。相应的模型分别为低水标模型、环模型和严格完整性模型。

a.低水标模型：任意主体可以读任意完整性级别的客体，但是如果主体读完整性级别比自己低的客体时，主体的完整性级别将为客体完整性级别，否则，主体的完整性级别保持不变。

b.环模型：不管完整性级别如何，任何主体都可以读任何客体。

c.严格完整性模型：这个模型对读操作是根据主客体的完整性级别严格控制的,即只有完整性级别低或相等的主体才可以读完整性级别高的客体,称为下读。

一般当我们提及 BIBA 模型，一般都是指 BIBA 严格完整性模型，总结来说是上写、下读。

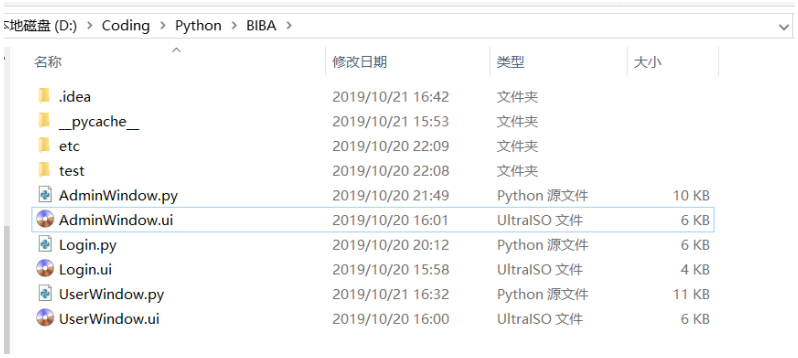
三、实验环境

实验环境初始化设置如下：

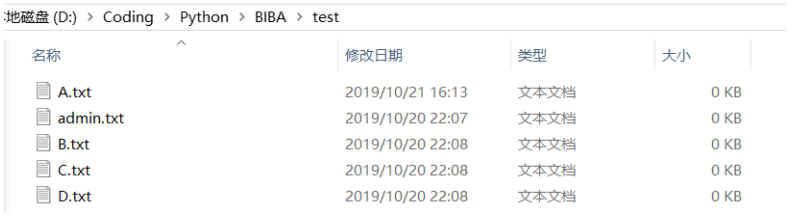
操作系统	Windows 10
运行环境	Python3.7，pyqt5
passwd.txt 文件位置	./etc/passwd.txt
预定义的四种等级的文件位置	./test

其中 passwd.txt 中包含账户名，经过 hash 的密码和用户等级。本实验设置了四个等级的文件组和用户组（A、B、C、D，其中 A 为最高等级，管理员的等级为 A）。整个项目的程序入口为 Login.py，然后分别会根据用户名进入 AdminWindow.py（管理员面板）和 UserWindow.py（用户面板）。

整个 BIBA python 项目的目录结构如下：



A、B、C、D 四个等级的文件如下



四、实验内容

1. 登录界面

从用户输入框得到账号和密码

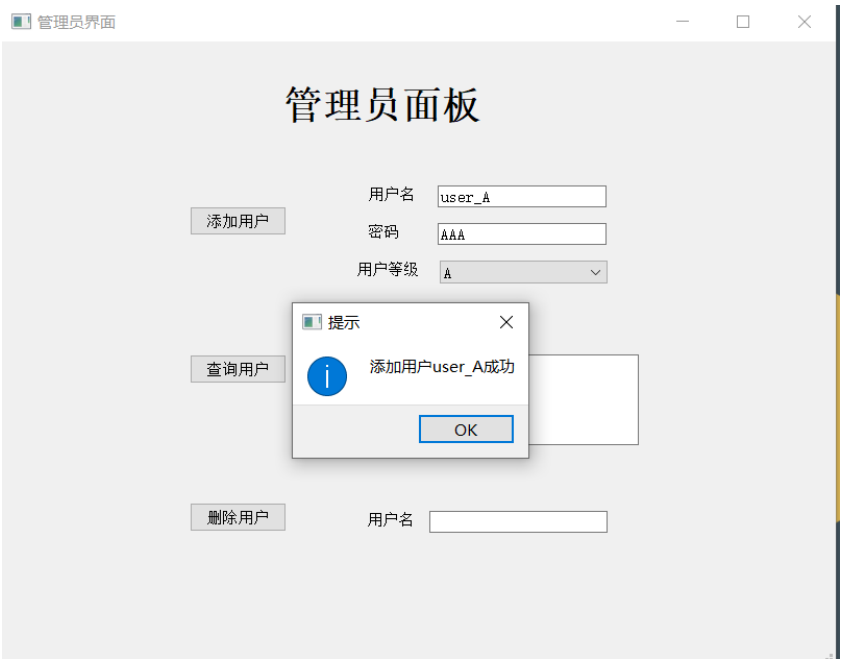
首先判断账号和密码是否合法，其次判断账号密码是否正确。若账号是“admin”，进入管理员界面。若为普通用户，进入用户界面。

登录界面的账号和密码判断函数如下

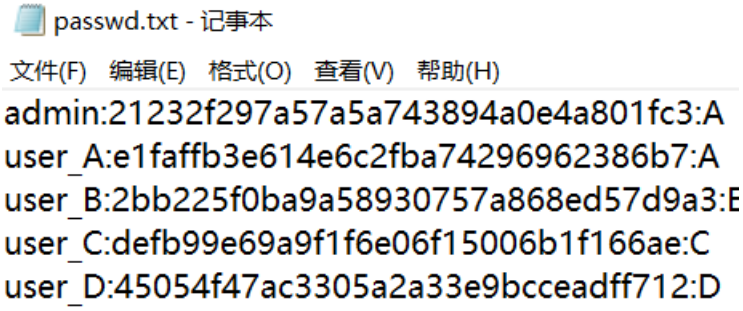
```
def check_user_and_passwd(self):
    user=self.lineEdit.text()
    passwd=self.lineEdit_2.text()
    md5=hashlib.md5()
    md5.update(passwd.encode("utf-8"))
    passwd=md5.hexdigest()
    if (user == '') or (passwd == ''):
        QMessageBox.warning(self, "警告", "账号和密码不能为空", QMessageBox.Yes)
        self.lineEdit.setFocus()
    print(user, passwd)
    fr = open('./etc/passwd.txt')
    arrayofLines = fr.readlines()
    numberofLines = len(arrayofLines)
    for line in arrayofLines:
        line = line.strip()
        listFromLine = line.split(':')
        name = listFromLine[0]
        if name == user:
            numberofLines = -1
            truepasswd = listFromLine[1]
            if truepasswd == passwd:
                group = listFromLine[2]
                print("\n登录成功!\n")
                if name == 'admin':
                    print('admin登录')
                    adminUI.show()
                    MainWindow.close()
                else:
                    urName = user
                    print("\n用户登录")
                    userUI.show()
```

2. 管理员界面

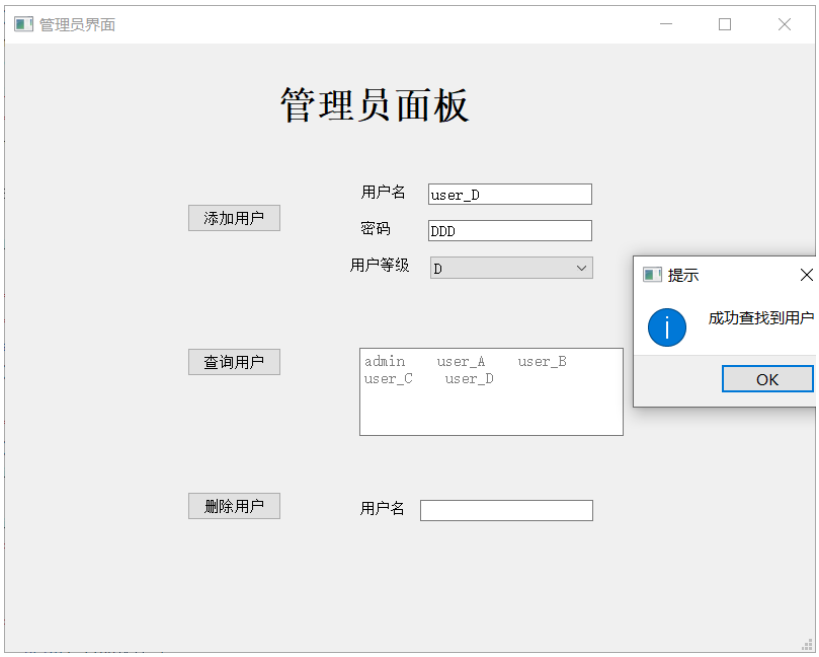
可以进行添加用户操作



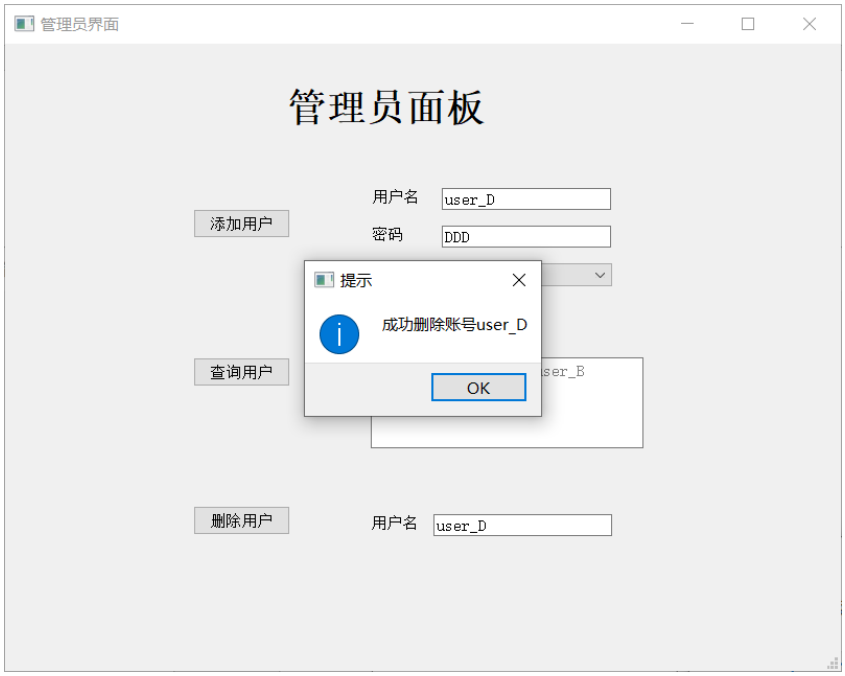
账号名，经过 hash 处理的密码，用户等级放在“./etc/passwd.txt”中



管理员可以查询系统中的所有用户



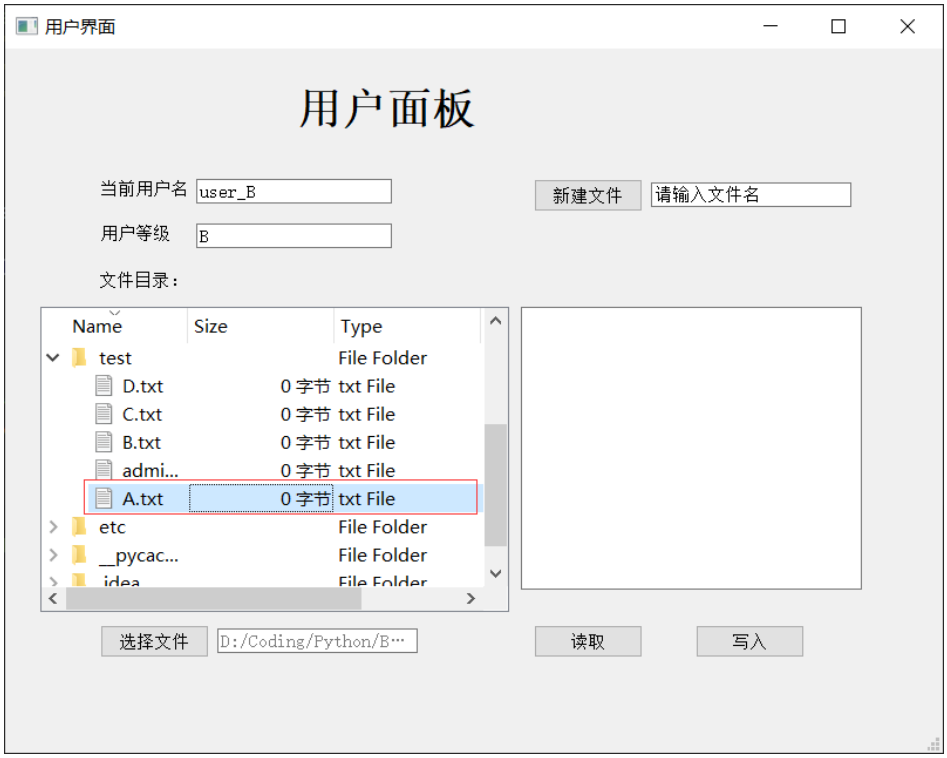
管理员可以删除系统中的用户



3.用户界面

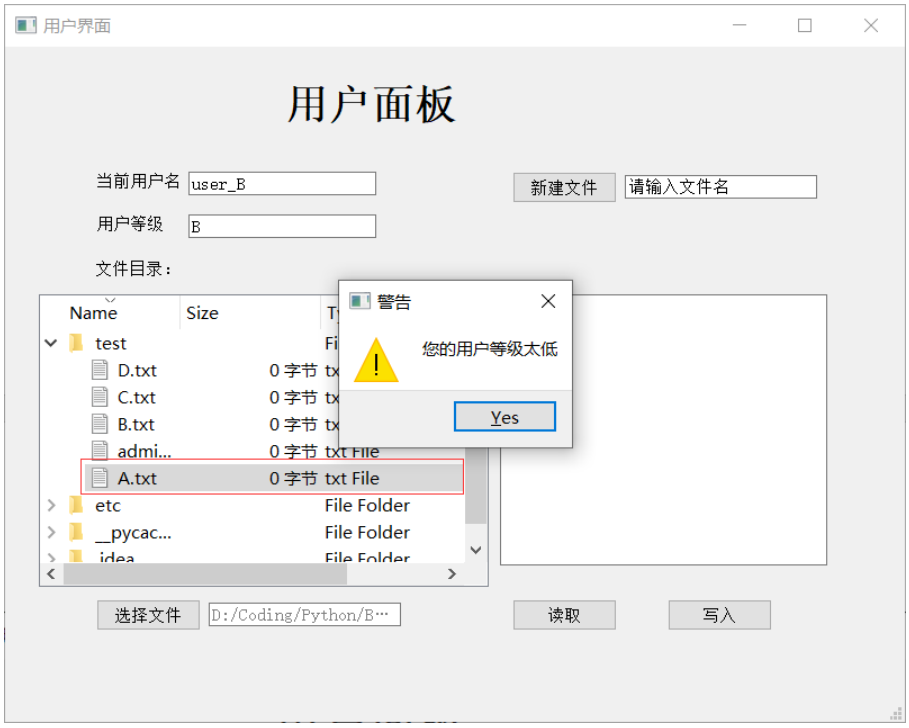
当普通用户登录时，会进入用户界面，此处以 user_B 为例。

双击“./test”文件夹下的文件，会选中该文件

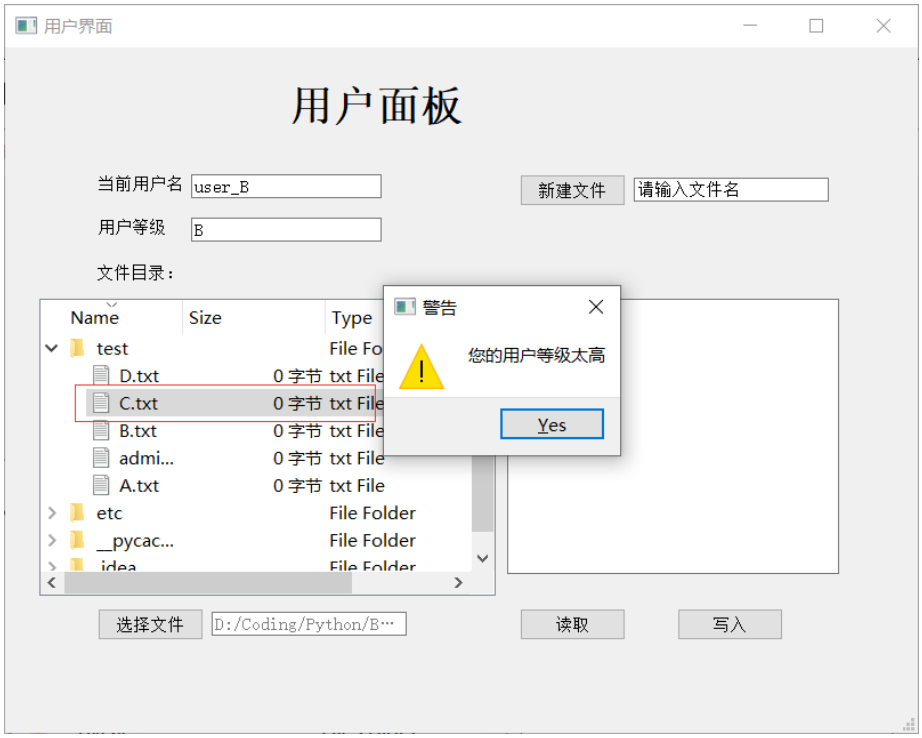


当用户等级为 B 的 user_B 读取文件等级为 A 的文件 A 时，可以读取高等级的文件。

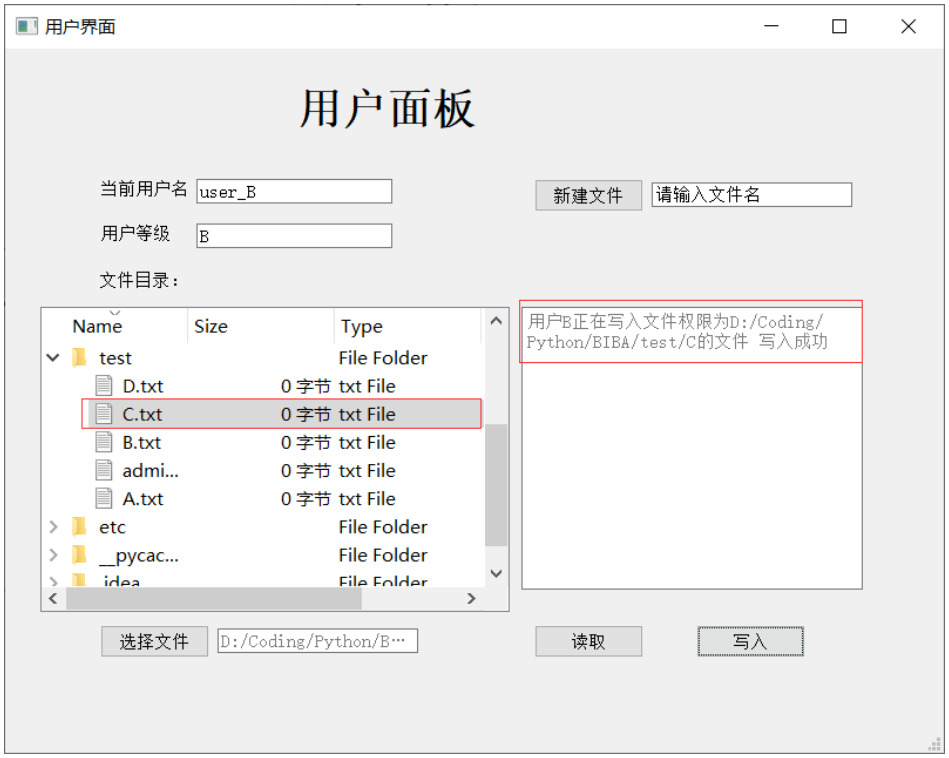
当用户等级为 B 的 user_B 写入文件等级为 A 的文件 A 时，由于文件等级过高，用户 user_B 没有写入的权限，弹出警告。



当用户等级为 B 的 user_B 读取文件等级为 C 的文件 C 时，由于用户等级过高，不能读取低等级的文件，弹出警告。



当用户等级为 B 的 user_B 写入文件等级为 C 的文件 C 时，可以写入低等级的文件。


用户界面
用户面板
当前用户名
user_B
用户等级
B
文件目录：
新建文件
请输入文件名
Name
Size
Type
test
File Folder
D.txt
0 字节 txt File
C.txt
0 字节 txt File
B.txt
0 字节 txt File
admi...
0 字节 txt File
A.txt
0 字节 txt File
etc
File Folder
__pycac...
File Folder
idea
File Folder
用户B正在写入文件权限为D:/Coding/Python/BIBA/test/C的文件 写入成功
选择文件
D:/Coding/Python/B···
读取
写入