**安徽科技学院**

**软件类实验报告**

|  |  |
| --- | --- |
| 课程名称： | 操作系统 |
| 学 号： | 2701200421 |
| 姓 名： | 乔梁 |
| 班 级： | 计算机科学与技术204班 |
| 指导教师： | 李文才 |
| 开课学期： | 2022-2023-1 |
| 学 院： | 信息与网络工程学院 |

安徽科技学院教务处

|  |  |  |  |
| --- | --- | --- | --- |
| **实验名称** | **文件管理实验** | **实验学时** | 4 |
| **实验场所** | **软件实验室九** | **实验性质** | **验证** |
| **一、实验目标**：  模拟系统中的用户控制管理建立文件，删除文件，打开文件，修改文件权限，读出文件，写入文件，关闭文件，不同用户之间的权限管理，使学生理解文件系统的基本结构和文件的管理。 | | | |
| **二、实验原理：**  文件通常存放在外存(如磁盘、磁带),上， 可以作为一个独立单位存放和实施相应的操作(如打开、关闭、读、写等)。为了加快对文件的检索，往往将文件控制块集中在一-起进行管理。这种文件控制块的有序集合称为文件目录。文件控制块就是其中的目录项。 | | | |
| **三、实验要求：**  使用程序模拟实现文件和用户的管理。 | | | |
| 四、实验步骤： （包括详细实验操作步骤、流程图、代码及重要语句注释）  主函数代码可以抽调成部分函数。  package ch3;  import java.io.FileInputStream; import java.io.FileOutputStream; import java.io.IOException; import java.util.Scanner;  public class Demo {  public static void main(String[] args) throws IOException {  MFD mfd = new MFD();  mfd.setSize(4);//设定用户数量。  SumUser(mfd);//增加用户  ControlMethod(mfd);//用户随后的操作输入。  }   private static void ControlMethod(MFD mfd) throws IOException {  boolean flag1 = true;//定义逻辑控制标记  while (flag1) {  System.out.print("请输入用户名：");  Scanner scanner = new Scanner(System.in);  String name = scanner.next();  if ((mfd.Contains(name) == null)) {  System.out.println("用户名不存在");  continue;  }  User user = mfd.Contains(name);   System.out.println("当前目录：");  for (int i = 0; i < user.getFilefold().size(); i++) {  System.out.println(user.getFilefold().get(i).getFile().getName() + '\t' + "权限" + user.getFilefold().get(i).getRwx() + '\t'  + "长度" + user.getFilefold().get(i).getFile().length() + "字节");  }   boolean flag = true;//定义逻辑控制标记  TestMeun(scanner, name, user, flag);//根据输入来选择操作。  System.out.print("是否退出系统：0)退出");  int jixu = Integer.parseInt(scanner.next());  System.out.println();  if (jixu == 0) {  flag1 = false;  System.out.println("系统退出");  }  }  }   private static void SumUser(MFD mfd) {  while (mfd.getSize() < 4) {  System.out.print("新增用户：");  Scanner scanner = new Scanner(System.in);  String username = scanner.next();  User user1 = new User(username);  mfd.addUser(user1);  }  }   private static void TestMeun(Scanner scanner, String name, User user, boolean flag) throws IOException {  while (flag) {  System.out.printf("%-14s%-15s%-12s\n", "1)新建文件", "2)删除文件", "3)打开文件");  System.out.printf("%-12s%-15s%-12s\n", "4)修改文件权限", "5)读出文件", "6)写入文件");  System.out.printf("%-14s%-13s%-12s\n", "7)关闭文件", "8)保存打印并退出", "0)不保存退出");  System.out.print("请输入指令: ");  int target = Integer.parseInt(scanner.next());  if (target < 0 || target > 8) {  System.out.println("输入有误(0-8):");  continue;  }  //根据指令运行相应功能  switch (target) {  //直接退出不保存  case 0: {  System.out.print("确定退出？(Y/N):");  String exit = scanner.next();  if (exit.equals("Y") || exit.equals("y")) {  flag = false;  System.out.println("已退出");  System.out.println();  user.remove();//删除用户目录里所有文件  break;  } else {  break;  }  }  //新建文件  case 1: {  if (user.getFilefold().size() == user.getFoldsize()) {  System.out.println("文件数目达到上限！");  break;  } else {  System.out.print("请输入文件名:");  String filename = scanner.next();  System.out.print("请输入文件保护码（读写执行）1表示允许，0表示不允许：");  String rwx = scanner.next();  java.io.File filetemp = new java.io.File("E:\\IDEACode\\算法\\src\\文件管理\\文件\\" + name + "\\" + filename);  filetemp.createNewFile();//创建文件  File file = new File(filetemp, rwx);  System.out.println("文件创建成功！");  user.addFile(file);  System.out.println();  break;  }  }  //删除文件  case 2: {  if (user.getFilefold().isEmpty()) {  System.out.println("目录为空！");  break;  } else {  System.out.println("当前目录：");  for (int i = 0; i < user.getFilefold().size(); i++) {  System.out.println(user.getFilefold().get(i).getFile().getName() + '\t' + "权限" + user.getFilefold().get(i).getRwx() + '\t'  + "长度" + user.getFilefold().get(i).getFile().length() + "字节");  }  System.out.print("请输入要删除的文件名：");  String filename = scanner.next();  user.delFile(filename);  System.out.println();  break;  }  }  //打开文件  case 3: {  if (user.getFilefold().isEmpty()) {  System.out.println("目录为空！");  break;  } else {  System.out.println("当前目录：");  for (int i = 0; i < user.getFilefold().size(); i++) {  System.out.println(i + "." + user.getFilefold().get(i).getFile().getName() + '\t' + "权限" +  user.getFilefold().get(i).getRwx() + '\t' + "长度" + user.getFilefold().get(i).getFile().length() + "字节");  }  System.out.print("请选择要打开的文件：（输入index）：");  String in = scanner.next();  int index = Integer.parseInt(in);  if (user.getFilefold().get(index).CanXing()) {  System.out.println("将选中的文件放入AFD中");  System.out.println("正在打开文件.....");  //创建一个文件输入流  FileInputStream is = new FileInputStream(user.getFilefold().get(index).getFile());  byte[] buf = new byte[1024];  int length = 0;  //循环读取文件内容，输入流中将最多buf.length个字节的数据读入一个buf数组中,返回类型是读取到的字节数。  //当文件读取到结尾时返回 -1,循环结束。  while ((length = is.read(buf)) != -1) {  System.out.print(new String(buf, 0, length));  }  //最后记得，关闭流  is.close();  } else {  System.out.println("没有该文件的运行权限");  }  System.out.println();  break;  }  }  //修改文件权限  case 4: {  if (user.getFilefold().isEmpty()) {  System.out.println("目录为空！");  break;  } else {  System.out.println("当前目录：");  for (int i = 0; i < user.getFilefold().size(); i++) {  System.out.println(i + "." + user.getFilefold().get(i).getFile().getName() + '\t' + "权限"  + user.getFilefold().get(i).getRwx() + '\t' + "长度" + user.getFilefold().get(i).getFile().length() + "字节");  }  System.out.print("请选择要修改权限的文件：（输入index）：");  String in = scanner.next();  int index = Integer.parseInt(in);  System.out.print("请输入文件新的权限（RWX）：");  String rwx = scanner.next();  user.getFilefold().get(index).setRwx(rwx);  System.out.println("修改成功！");  System.out.println();  break;  }  }  //读出文件  case 5: {  if (user.getFilefold().isEmpty()) {  System.out.println("目录为空！");  break;  } else {  System.out.println("当前目录：");  for (int i = 0; i < user.getFilefold().size(); i++) {  System.out.println(i + "." + user.getFilefold().get(i).getFile().getName() + '\t' + "权限" + user.getFilefold().get(i).getRwx());  }  System.out.print("请选择要读取的文件：（输入index）：");  String in = scanner.next();  int index = Integer.parseInt(in);  if (user.getFilefold().get(index).CanRead()) {  System.out.println("将选中的文件放入AFD中");  System.out.println("正在读取文件.....");  FileInputStream is = new FileInputStream(user.getFilefold().get(index).getFile());  byte[] buf = new byte[1024];  int length = 0;  //循环读取文件内容，输入流中将最多buf.length个字节的数据读入一个buf数组中,返回类型是读取到的字节数。  //当文件读取到结尾时返回 -1,循环结束。  System.out.println("文件内容如下");  while ((length = is.read(buf)) != -1) {  System.out.print(new String(buf, 0, length));  }  //最后记得，关闭流  is.close();  } else {  System.out.println("没有该文件的读取权限");  }  System.out.println();  break;  }   }  //写入文件  case 6: {  if (user.getFilefold().isEmpty()) {  System.out.println("目录为空！");  break;  } else {  System.out.println("当前目录：");  for (int i = 0; i < user.getFilefold().size(); i++) {  System.out.println(i + "." + user.getFilefold().get(i).getFile().getName() + '\t' + "权限" +  user.getFilefold().get(i).getRwx() + '\t' + "长度" + user.getFilefold().get(i).getFile().length() + "字节");  }  System.out.print("请选择要写入的文件：（输入index）：");  String in = scanner.next();  int index = Integer.parseInt(in);  if (user.getFilefold().get(index).CanWrite()) {  System.out.print("请输入要写入的内容：");  String content = scanner.next();  FileOutputStream os = new FileOutputStream(user.getFilefold().get(index).getFile());  byte[] t = new byte[1024];  t = content.getBytes();//将字符串编码为 byte 序列，并将结果存储到一个新的 byte 数组中  os.write(t, 0, t.length);//即向文件写入一个byte数组的内容  //关闭资源  os.close();  System.out.println("将选中的文件放入AFD中,AFD的写指针指向选中文件");  System.out.println("正在写入文件...");  System.out.println("写入成功");  } else {  System.out.println("没有该文件的写入权限");  }  System.out.println();  break;  }  }  //关闭文件  case 7: {  System.out.println("文件已关闭！");  break;  }  //保存目录并打印  case 8: {  //打印目录  user.display();  flag = false;  break;  }  default:  continue;  }  }  } }  package ch3;  public class File {  private java.io.File file;  private int[] rwx = new int[3];// 安全标记    public File(java.io.File file, String rwx) {  this.file = file;  this.setRwx(rwx);  }   public String getRwx() {  return Integer.toString(rwx[0]) + Integer.toString(rwx[1]) + Integer.toString(rwx[2]);  }   public void setRwx(String rwx) {  this.rwx[0] = Integer.parseInt(String.valueOf(rwx.charAt(0)));   this.rwx[1] = Integer.parseInt(String.valueOf(rwx.charAt(1)));   this.rwx[2] = Integer.parseInt(String.valueOf(rwx.charAt(2)));  }   public java.io.File getFile() {  return file;  }   public boolean CanRead() {  if (this.rwx[0] == 1) {  return true;  }  return false;  }   public boolean CanWrite() {  if (this.rwx[1] == 1) {  return true;  }  return false;  }   public boolean CanXing() {  if (this.rwx[2] == 1) {  return true;  }  return false;  } }  package ch3;  import java.util.ArrayList;  public class User {  private String username;// 用户名  private ArrayList<File> filefold = new ArrayList();//用户目录  private int foldsize = 10;// 目录大小   public User() {  }   public User(String username) {  this.username = username;  java.io.File ff = new java.io.File("E:\\IDEACode\\code\\" + username);  ff.mkdir();  }   public String getUsername() {  return username;  }   public int getFoldsize() {  return foldsize;  }   //未选择保存则清空目录  public void remove() {  this.filefold.removeAll(filefold);  /\*java.io.File ff = new java.io.File("E:\IDEACode\code\"+username);  ff.delete();\*/  }   //添加文件  public void addFile(File file) {  filefold.add(file);  }   public ArrayList<File> getFilefold() {  return filefold;  }   //删除文件  public void delFile(String filename) {  java.io.File file = new java.io.File("E:\\IDEACode\\code\\" + username + "\\" + filename);  if (filefold.isEmpty()) return;  for (int i = 0; i < filefold.size(); i++) {  if (this.filefold.get(i).getFile().getName().equals(filename)) {  this.filefold.remove(i);  }  }  file.delete();  System.out.println("文件删除成功！");  }   public void display() {  if (filefold.isEmpty()) {  System.out.println("目录为空！");  }  for (int i = 0; i < this.getFilefold().size(); i++) {  System.out.println(this.getFilefold().get(i).getFile().getName());  }  } }  **五、实验结果：** | | | |
| **六、实验小结：**  对文件和用户的管理操作更有实感，对其的代码程序理解更具体，更高效。 | | | |