

**课程设计(综合实验)报告**

( 2019 – 2020 年度第 2 学期)

名 称： 操作系统综合实验

题 目： 文件系统

院 系： 计算机系

班 级： 软件1802

学 号： 201809000121

学生姓名： 王毅涛

指导教师： 王德文

设计周数： 分散进行

成 绩：

日期： 年 月 日

**一、 综合实验的目的与要求**

本实验模拟文件管理的工作过程，从而对各种文件操作命令的实质内容和执行过程有比 较深入的了解。模拟一个文件系统，包括目录文件，普通文件，并实现对它们的一些基本操 作

假定每个目录文件最多只能占用一个块；一个目录项包括文件名（下一级目录 名），文件类型，文件长度，指向文件内容（下一级目录）的指针内容。普通文件可以 只用目录项（FCB）代表。需要实现一个命令行操作界面，包含如下命令：CD改变目录 CREATE新建文件 LSALL 列出所有目录和文件MD新建目录 RD移除目录 DEL删除文件

**二、实验正文**

本次实验我设定了四个类：目录节点类，普通文件类，目录文件类，管理类。管理类负责接收客户端指令，目录节点类内有普通文件类实例和目录文件类实例的引用，指向的文件的类型。普通文件类含有文件名，长度。目录文件类含有一个目录节点的ArrayList，方便删改添加。具体请参阅实验代码和流程图

**三、综合实验总结或结论**

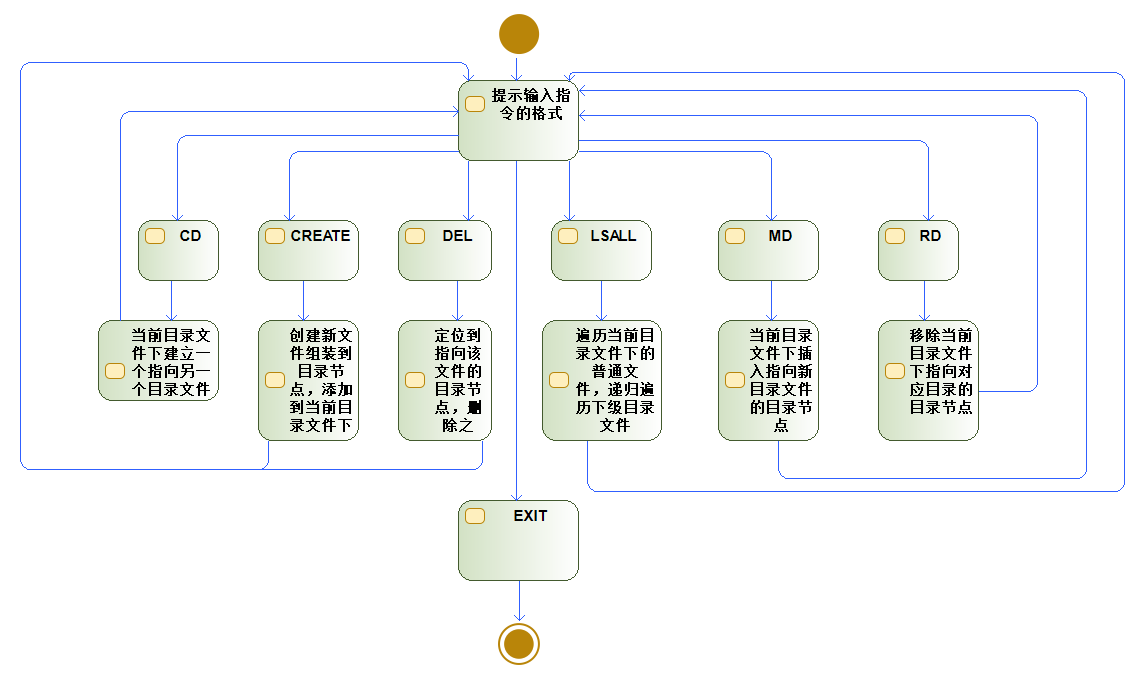
本实验的基本目的达成。

本次实验难度不高，重点在于合理编排目录节点，普通文件，目录文件的关系，我设定的是目录文件普通文件均受目录节点的控制，一个目录节点对应一个普通文件或目录文件。然后就是代码编写了。本次实验没有编写回到上一级目录的功能，这是一个遗憾。

**四、参考文献**

[1] 汤小丹, 梁红兵.《计算机操作系统》. 西安电子科技大学出版社, 第四版. 2014-05

**附录（设计流程图、程序、表格、数据等）**

流程图：

管理类：

1. **package** fiveFileSystem;
3. **import** java.util.Scanner;
5. **public** **class** Manager {
6. DirNode root;//Je suis Alpla...
7. DirFile curdf;//current DirFile
8. Scanner in =**new** Scanner(System.in);
9. **public** Manager() {
10. **super**();
11. **this**.root=**new** DirNode();
12. root.AppointDF("ROOT");
13. **this**.curdf=**this**.root.getDf();
14. }
15. **public** **void** Main() {
16. System.out.println("CD <目录名>变换目录\nCREATE <文件名> <文件长>创建文件\n"
17. + "DEL <文件名>删除文件\nLSALL 全体列表\n"
18. + "MD <目录名称>创建目录\n"
19. + "RD <目录名>移除目录\n"+
20. "HELP 提示操作信息");
21. **while** (**true**) {
22. System.out.println("\n请输入您的指令，回车结束。");
23. String op=in.next();
24. **if** (op.equalsIgnoreCase("CD")) {
25. String name=in.next();
26. **this**.ChangeDir(name);
27. }
28. **else** **if** (op.equalsIgnoreCase("CREATE")) {
29. String name=in.next();
30. **int** size=in.nextInt();
31. NormFile nf=**new** NormFile(name,size);
32. **this**.CreateNF(nf);
33. }
34. **else** **if** (op.equalsIgnoreCase("DEL")) {
35. String name=in.next();
36. **this**.DeleteFile(name);
37. }
38. **else** **if** (op.equalsIgnoreCase("LSALL")) {
39. **this**.displayDir(root);
40. }
41. **else** **if** (op.equalsIgnoreCase("MD")) {
42. String name=in.next();
43. **this**.MakeDir(name);
44. }
45. **else** **if** (op.equalsIgnoreCase("RD")) {
46. String name=in.next();
47. **this**.removeDir(name);
48. }
49. **else** **if** (op.equalsIgnoreCase("HELP")) {
50. System.out.println("CD <目录名>变换目录\nCREATE <文件名> <文件长>创建文件\n"
51. + "DEL <文件名>删除文件\nLSALL 全体列表\n"
52. + "MD <目录名称>创建目录\nRD <目录名>移除目录");
53. }
54. **else** {
55. System.out.println("Wrong Operant please check again");
56. }
57. }
58. //      String op=in.next();
59. //      if (op.equalsIgnoreCase("CD")) {
60. //          String name=in.next();
61. //          this.ChangeDir(name);
62. //      }
63. //      else if (op.equalsIgnoreCase("CREATE")) {
64. //          String name=in.next();
65. //          int size=in.nextInt();
66. //          NormFile nf=new NormFile(name,size);
67. //          this.CreateNF(nf);
68. //      }
69. //      else if (op.equalsIgnoreCase("DEL")) {
70. //          String name=in.next();
71. //          this.DeleteFile(name);
72. //      }
73. //      else if (op.equalsIgnoreCase("LSALL")) {
74. //          this.displayDir(root);
75. //      }
76. //      else if (op.equalsIgnoreCase("MD")) {
77. //          String name=in.next();
78. //          this.MakeDir(name);
79. //      }
80. //      else if (op.equalsIgnoreCase("RD")) {
81. //          String name=in.next();
82. //          this.removeDir(name);
83. //      }
84. //      else {
85. //          System.out.println("Wrong Operant please check again");
86. //      }
87. }//主要控制中枢
88. **public** **void** ChangeDir(String dirName) {
89. DirNode tmpdn=curdf.FindDF(dirName);
90. **if** (tmpdn==**null**) {
91. System.out.println("\n!No such Directory!\n");
92. **return**;
93. }**else** {
94. System.out.println("Directory "+tmpdn.getName()+"has been found.JUMPING...");
95. **this**.curdf=tmpdn.getDf();
96. }
97. }//CD
98. **public** **void** CreateNF(NormFile tmpnf) {
99. DirNode node=**new** DirNode();
100. node.AppointNF(tmpnf);
101. **this**.curdf.AddDirNode(node);
102. }//CREATE
103. **public** **void** DeleteFile(String fileName) {
104. DirNode tmpdn=**this**.curdf.FindNF(fileName);
105. **if** (tmpdn==**null**) {
106. System.out.println("No such File.");
107. }
108. **else** {
109. **this**.curdf.deleteFile(tmpdn.name);
110. System.out.println("File founded and deleted.");
111. }
112. }//DEL
113. **public** **void** displayDir(DirNode startNode) {
114. DirFile tmpdf=startNode.getDf();
115. System.out.println("-----------Dir name "+tmpdf.dirName+" displaying-----------");
116. **int** limit=tmpdf.nodeArr.size();
117. **for** (**int** i=0;i<limit;i++) {
118. **if** (tmpdf.nodeArr.get(i)!=**null**) {
119. **if** (tmpdf.nodeArr.get(i).getType()==1) {
120. System.out.println("File,name as "+tmpdf.nodeArr.get(i).getName());
121. }
122. **else** **if** (tmpdf.nodeArr.get(i).getType()==0) {
123. displayDir(tmpdf.nodeArr.get(i));//递归
124. }**else** {
125. ;
126. }
127. }**else** {
128. ;
129. }
130. }
131. System.out.println("-----------Dir name "+tmpdf.dirName+" display over-----------");
132. }//LSALL
133. **public** **void** MakeDir(String name) {
134. DirNode node=**new** DirNode();
135. node.AppointDF(name);
136. **this**.curdf.AddDirNode(node);
137. System.out.println("MD finished.");
138. }//MD
139. **public**  **void** removeDir(String dirName) {
140. DirNode toDelete=**this**.curdf.FindDF(dirName);
141. **if** (toDelete==**null**) {
142. System.out.println("No such Dir.");
143. }
144. **else** {
145. //  this.curdf.deleteFile(tmpdn.name);
146. **this**.curdf.deleteDir(toDelete);
147. System.out.println("Dir founded and removed.");
148. }
149. }//RD
151. }

目录节点类：

1. **package** fiveFileSystem;
3. **public** **class** DirNode {
4. **int** label;//不用管
5. String name;//指向目录文件，下级目录名称；指向普通文件，普通文件名称
6. **int** type;//指向的是0目录文件1普通文件
7. NormFile nf;//类型：1
8. DirFile df;//类型：0
9. //有就指向，没有空着
10. **public** DirNode() {
11. **super**();
12. **this**.type=810;//初始值
13. // TODO Auto-generated constructor stub
14. }
15. **public** **void** AppointNF(NormFile tmpnf) {
16. **this**.nf=tmpnf;
17. **this**.name=tmpnf.name;
18. **this**.setType(1);
19. }
20. **public** **void** AppointDF(String name) {
21. **this**.name=name;
22. **this**.setType(0);
23. **this**.df=**new** DirFile(**this**.name);
24. }
25. **public** **void** Clear() {
26. **this**.df=**null**;
27. **this**.nf=**null**;
28. **this**.type=810;
29. **this**.name="";
30. }
32. **public** String getName() {
33. **return** name;
34. }
35. **public** **int** getType() {
36. **return** type;
37. }
38. **public** NormFile getNf() {
39. **return** nf;
40. }
41. **public** DirFile getDf() {
42. **return** df;
43. }
44. **public** **void** setName(String name) {
45. **this**.name = name;
46. }
47. **public** **void** setType(**int** type) {
48. **this**.type = type;
49. }
50. **public** **void** setNf(NormFile nf) {
51. **this**.nf = nf;
52. }
53. **public** **void** setDf(DirFile df) {
54. **this**.df = df;
55. }
57. }

普通文件类：

1. **package** fiveFileSystem;
3. **public** **class** NormFile {
5. **public** String name;
6. **public** **int** size;
7. **public** NormFile(String name, **int** size) {
8. **super**();
9. **this**.name = name;
10. **this**.size = size;
11. }
13. }

目录文件类：

1. **package** fiveFileSystem;
3. **import** java.util.ArrayList;
4. **import** java.util.Iterator;
6. **public** **class** DirFile {
7. ArrayList <DirNode> nodeArr;
8. String dirName;
9. **public** DirFile() {
10. **super**();
11. **this**.nodeArr=**new** ArrayList<DirNode>();
12. // TODO Auto-generated constructor stub
13. }
14. **public** DirFile(String dirName) {
15. **super**();
16. **this**.dirName = dirName;
17. **this**.nodeArr=**new** ArrayList<DirNode>();
18. }
20. **public** **void** AddDirNode(DirNode tmpdn) {
21. **this**.nodeArr.add(tmpdn);
22. }
24. **public** DirNode FindDF(String name) {
25. Iterator <DirNode> it = nodeArr.iterator(); //迭代器
26. **while** (it.hasNext()) {
27. DirNode tmp=it.next();
28. **if** (name.equals(tmp.getName())&&tmp.getType()==0) {
29. **return** tmp;
30. }
31. **else** {;}
33. }
34. //System.out.println("\n!No such Directory!\n");
35. **return** **null**;
36. }//这个找下级目录文件,但是返回指向它的目录节点
38. **public** DirNode FindNF(String name) {
39. Iterator <DirNode> it = nodeArr.iterator(); //迭代器
40. **while** (it.hasNext()) {
41. DirNode tmp=it.next();
42. **if** (name.equals(tmp.getName())&&tmp.getType()==1) {
43. **return** tmp;
44. }
45. **else** {;}
47. }
48. //System.out.println("\n!No such Directory!\n");
49. **return** **null**;
50. }//这个是找普通文件,但是返回指向它的目录节点
51. **public** **void** deleteFile(String name) {
52. **if** (**this**.nodeArr!=**null**) {
53. **int** limit=**this**.nodeArr.size();
54. **for** (**int** i=0;i<limit;i++) {
55. **if** (**this**.nodeArr.get(i)==**null**) {
56. **continue**;
57. }
58. **else** {
59. **if** (name.equals(**this**.nodeArr.get(i).getName())&&**this**.nodeArr.get(i).getType()==1)
60. {
61. **this**.nodeArr.remove(i);
62. **return**;
63. }
64. **else** {;}
65. }
66. }
67. }
68. **else** {
69. System.out.println("Null Array!");
70. }
71. }//清除要删除的文件所在的DirNode
72. **public** **void** deleteDir(DirNode toDelete) {
73. **if** (**this**.nodeArr!=**null**) {
74. **int** limit=**this**.nodeArr.size();
75. **for** (**int** i=0;i<limit;i++) {
76. **if** (**this**.nodeArr.get(i)==**null**) {
77. **continue**;
78. }
79. **else** {
80. **if** (toDelete.getName().equals(**this**.nodeArr.get(i).getName())&&**this**.nodeArr.get(i).getType()==0)
81. {
82. **this**.nodeArr.remove(i);
83. **return**;
84. }
85. **else** {;}
86. }
87. }
88. }
89. **else** {
90. System.out.println("Null Array!");
91. }
92. }//连接目录文件的目录节点删除
93. }