**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** | **磁盘管理** | **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**：  掌握磁盘管理使用的FCFS,SSTF,SCAN,CSCAN算法，对算法本身的运行细节有更深的理解和认识。 | | | |
| **二、实验原理：**  磁盘读写原理，先来先服务原理，对磁道总数的概念理解。 | | | |
| **三、实验要求：**  使用程序模拟和实现磁盘管理使用的各项算法和操作。。 | | | |
| 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）  鉴于算法代码多次复用排序算法，可以将其换成快速排序来加快代码速度，并将其放入主函数内减少代码冗余，并将算法类代码继承于主类，使得排序代码只需要写一遍即可。  另外为了方便的使用提供的sort方法排序，将int类修为其应用类Integer。  package ch4;  import java.util.Arrays; import java.util.Comparator; import java.util.Scanner;  public class DiskManage {  **public Integer[] Sorts(Integer[] a, int n) {  Arrays.sort(a, new Comparator<Integer>() {  @Override  public int compare(Integer o1,Integer o2) {  return o2-o1;  }  });  return a;  }**  public static void main(String[] args) {  Integer numS[] = new Integer[20];  int n = 0;  int m = 0;  double sum = 0;  double temp = 0;  int position = 0;  SSTF sstf = new SSTF();  SCAN scan = new SCAN();  CSCAN cscan = new CSCAN();  System.out.println("请输入磁盘请求的个数：");  try {  Scanner reader = new Scanner(System.in);  n = reader.nextInt();  } catch (Exception e) {  System.out.println("null");  }  System.out.println("请输入磁盘请求序列：");  Scanner reader = new Scanner(System.in);  for (int i = 1; i <= n; i++) {  try {  numS[i] = reader.nextInt();  } catch (Exception e) {  }  }  System.out.print("请输入磁头的初始位置：");   // try{  Scanner re = new Scanner(System.in);  numS[0] = re.nextInt();  position = numS[0]; // }  // catch(for (int i = 1; i <= n; i++){Exception e){ }  // System.out.print(a[i] + " "); }  Post();  Scanner read = new Scanner(System.in);  try {  m = read.nextInt();  } catch (Exception e) {  }  System.out.println("m的值为" + m);  read.close();  switch (m) {  case 1:  sum = 0;  System.out.println("FCFS算法序列：");  for (int i = 1; i <= n; i++) {  System.out.print(numS[i] + " ");  temp = numS[i] - numS[i - 1];  if (temp >= 0) ;  else if (temp < 0) {  temp = -temp;  }  sum = sum + temp;  }  System.out.println();  System.out.println("sum的值为" + sum);  System.out.println("平均寻到长度为" + sum / n);  break;  case 2:  sstf.Calculate(numS, n, position);  break;  case 3:  scan.Check(numS, n, position);  break;  case 4:  cscan.Check(numS, n, position);  break;  }  }   private static void Post() {  System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.out.println("提示：磁头的序列范围1~200！磁头从小号到大号寻道！");  System.out.println("请输入相应的选择操作：：");  System.out.println("1、先来先服务算法（FCFS）");  System.out.println("2、最短寻道时间优先算法（SSTF）");  System.out.println("3、扫描算法（SCAN）");  System.out.println("4、循环扫描算法（CSCAN）");  } }  package ch4;  public class CSCAN extends DiskManage{  public int m = 0; //用来存放磁头的初始位置  public boolean Run = true;  public int sum = 0;   public void Check(Integer[] a, int n, int position) {  a = Sorts(a, n);//排序  while (Run) {  for (int i = 0; i <= n; i++) {  if (a[i] == position) {  m = i;  Run = false;  }  }  }  System.out.println("CSCAN算法序列：");  for (int i = m + 1; i <= n; i++) { //磁头向大号移动  sum = sum + a[i] - a[i - 1];  System.out.print(a[i] + " ");  }  sum = sum + 200 - a[n];  sum = sum + 199;  sum = sum + a[0] - 1;  for (int i = 0; i <= m - 1; i++) {  if (i != m - 1) {  sum = sum + a[i + 1] - a[i];  }  System.out.print(a[i] + " ");  }  System.out.println();  System.out.println("寻道长度为" + sum);  System.out.println("平均寻道长度为" + sum / n);  } }  package ch4;   public class SCAN extends DiskManage{  public int m = 0; //用来存放磁头的初始位置  public boolean Run = true;  public int sum = 0;   public void Check(Integer[] a, int n, int position) {  a = Sorts(a, n);//排序  while (Run) {  //此循环用来寻找磁头的初始位置被排到了什么位置  for (int i = 0; i <= n; i++) {  if (a[i] == position) {  m = i;  Run = false;  }  }  }  System.out.println("SCAN算法序列：");  for (int i = m + 1; i <= n; i++) {//磁头向大号移动  sum = sum + a[i] - a[i - 1];  System.out.print(a[i] + " ");  }  sum = sum + 200 - a[n];  sum = sum + 200 - a[m - 1];  for (int i = m - 1; i >= 0; i--) {  if (i != 0) {  sum = sum + a[i] - a[i - 1];  }  System.out.print(a[i] + " ");  }  System.out.println();  System.out.println("寻到长度为" + sum);  System.out.println("平均寻到长度为" + sum / n);  } }  package ch4;  public class SSTF extends DiskManage{  public int m; // 用来判断排序后 磁头所在的初始位置的下标  public int b[];  public int rflag = 0, lflag = 0;  public double SUM = 0;  public int mleft, mright;   public SSTF() {  b = new int[20];  }   public void Calculate(Integer[] a, int n, int position) {  a = Sorts(a, n);//排序  if (a[n] == position) {//如果磁头的初始位置在最大号  System.out.println("SSTF序列为：");  for (int i = 0; i <= n; i++) {  System.out.print(a[i] + " ");  }  } else if (a[0] == position) {  System.out.println("SSTF序列为：");  for (int i = n; i >= 0; i--) {  System.out.print(a[i] + " ");  }  } else {  System.out.println("SSTF序列为：");  Check(a, n, position);  }  }   public void Check(Integer[] a, int n, int position) {  boolean Run = true, Running = true;  while (Run) {//此循环用来寻找磁头的初始位置被排到了什么位置  for (int i = 0; i <= n; i++) {  if (a[i] == position) {  m = i;  System.out.println("m的值为" + m);  mleft = m - 1;  mright = m + 1;  Run = false;  }  }  }  while (Running) {  if (a[m] - a[mleft] >= a[mright] - a[m] && rflag != 1) {  SUM = SUM + (a[mright] - a[m]); //用来统计寻到长度  m = mright;  mright++;  if (mright > n) {  rflag = 1;  }  System.out.print(a[m] + " ");  } else if (a[m] - a[mleft] < a[mright] - a[m] && lflag != 1) {  SUM = SUM + (a[m] - a[mleft]);  m = mleft;  mleft--;  if (mleft < 0) {  lflag = 1;  }  System.out.print(a[m] + " ");  }  if (mleft != 0 && mright > n) {  Running = false;  SUM = SUM + a[n] - a[mleft];  for (int j = mleft; j >= 0; j--) {  if (j != 0) {  SUM = SUM + a[j] - a[j - 1];  }  System.out.print(a[j] + " ");  }  }  if (mright != n && mleft < 0) {  Running = false;  SUM = SUM + a[mright] - a[0];  for (int k = mright; k <= n; k++) {  if (k != 0) {  SUM = SUM + a[k + 1] - a[k];  }  System.out.print(a[k] + " ");  }  }  }  System.out.println();  System.out.println("sum的值为" + SUM);  System.out.println("平均寻到长度为" + SUM / n);  } }    **五、实验结果：** | | | |
| **六、实验小结：**  对JAVA各个类的使用和排序都有了很深刻的认识，对磁盘管理的各项算法有了更加细节上的了解和认知，在面对细节问题时仔细思考代码的使用和意义，就能很好的了解这些算法。 | | | |