广东金融学院实验报告

课程名称：算法分析与设计

装订线

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验编号  及实验名称 | 算法分析与设计实验2 | | | 系 别 | 计算机科学与技术 |
| 姓 名 | 林旋华 | 学 号 | 181543306 | 班 级 | 1815433 |
| 实验地点 | 电教503 | 实验日期 | 2020.9.17 | 实验时数 | 2 |
| 指导教师 | 郭艺辉 | 同组其他成员 | 无 | 成 绩 |  |
| 1. 实验目的及要求 2. 掌握递归与分治法的基本思想及基本原理。 3. 掌握使用分治法求解问题的一般特征及步骤。 4. 掌握分治法的设计方法及复杂性分析方法。 5. 掌握基于分治策略的合并排序算法的问题描述、算法设计思想、算法设计过程及程序编码实现。 | | | | | |
| 1. 实验环境及相关情况（包含使用软件、实验设备、主要仪器及材料等）   1) 操作系统：Windows操作系统  2) 开发工具：Eclipse、JDK  3) 开发语言：Java | | | | | |
| 1. 实验内容及步骤（包含简要的实验步骤流程） 2. 采用递归算法产生的全排列。写出算法实现代码并截屏程序的运行结果。 3. 给定一个包含*n*个元素的一维线性序列，对这*n*个元素按照非递减顺序排序。设，采用基于分治策略的合并排序算法解决该问题。    1. 简述合并排序算法基本思想以及步骤。    2. 写出算法实现代码并截屏程序的运行结果。    3. 写出该算法所需计算时间的递归方程，并写出求解结果。 | | | | | |
| 1. 实验结果（包括程序或图表、结论陈述、数据记录及分析等，可附页）   **代码:**  public void perm(int[] li,int k,int m)  {  //产生list[k:m]的所以排列  if(k==m) {  //只剩下一个元素  for(int i=0;i<=m;i++) {  System.out.print(li[i]);  }  System.out.println();  }  else  for(int i=k;i<=m;i++) {  swap(li,k,i);  perm(li,k+1,m);  swap(li,k,i);  }  }  private static void swap(int[] list, int m, int n) {  int temp = (int) list[m];  list[m] = list[n];  list[n] = temp;  }  public static void main(String[] args) {  int[] li = {1,2,3};  perm p = new perm(); //如果全部方法都是static,可以不用new对象  p.perm(li, 0, li.length-1);  }  **截图：**    a)**基本思想：**  首先将n个待排序元素分成两个规模大致相同的子数组。如果子数组规模依然较大,那么继续分割子数组,当子数组只含单元素时,认为单元素数组本身已经排好序,这时将相邻的两个有序子数组两两合所要求的有序序列,算法终止。  **步骤：**   1. 初始化时, left= 0, right =7,合并排序算法首先将待排序元素分成两个规模大致相同的子数组, i=└(left+ right)/2┘=3 ,分割的位置为3,两个子数组分别为a[0:3]和a[4:7] 2. 对于左子数组a[0:3], left =0, right=3。合并排序算法继续将a[0:3]分成两个规模大致相同的子数组, i=└(left+ right)/2 ┘=1,分割的位置为1,两个子数组分别 3. 为a[0:1]和a[2:3]。 4. 对于左子数组a[0:1], left = 0, right=1,合并排序算法继续将a[0:1]分成两个规模大致相同的子数组, i=└(left+ right)/2 ┘=0,分割的位置为0,两个子数组分别为a[0:0]和a[1:1]。 5. 左子数组a[0:0], let= right,合并排序算法递归调用结束,左子数组a[0:0]不再分割。右子数组a[1:1], left=right,并排序算法递归调用结束,右子数组a[1:1]也不再分割。算法执行Merge(a, b, 0, 0,1)。设置一个辅助数组b[ ],设置三个游标i,j, k分别指向左子数组a[0:0]、右子数组a[1:1]和辅助数组b[ ]。将左子数组a[0:0]与右子数组a[1:1]中的元素按由小到大的顺序依次放入辅助数组b[ ],然后将数组b[ ]复制回数组a[ ]。 6. 继续分割子数组a[0:3]的右子数组a[2:3], left = 2, right -3。合并排序算法继续将a[2:3]分成两个规模大致相同的子数组, i=└(left+ right)/2 ┘=2,分割的位置为2,两个子数组分别为a12:2]和a[3:3]。 7. 左子数组a[2:2], left = right,合并排序算法递归调用结束,左子数组a[2:2]不再分割。右子数组a[3:3], left = right,合并排序算法递归调用结束,右子数组a[3:3]也不再分割。算法执行Merge(a b,2,2, 3),设置一个辅助数组b[ ],设置三个游标i.j,k分别指向左子数组a[2:2]、右子数组a[3:3]和辅助数组b[ ]。将左子数组a12:2]与右子数组a[3:3]中的元素按由小到大的顺序依次放入辅助数组b[ ],然后将数组b[ ]复制回数组a[ ]。 8. 算法继续执行Merge(a, b,0,1,3)。设置一个辅助数组b[ ],设置三个游标i,j,k分别指向左子数组a[0:1]、右子数组a[2:3]和辅助数组b[ ]。将左子数组a[0:1]与右,子数组a[2:3]中的元素按由小到大的顺序依次放入辅助数组b[ ],然后将数组b[ ]复制回数组a[ ]。 9. 对a[0:7]的右子数组a[4:7]执行相同的算法,可以得到排好序的右子数组a[4:7]={17,18,25,30}。 10. 算法继续执行Mergea(a, b, 0,3,7),设置一个辅助数组b[ ],设置三个游标i,j,k,分别指向左子数组a[0:3]、右子数组a[4:7]和辅助数组b[ ]。将左子数组a[0:3]与右子数组a[4:7]中的元素按由小到大的顺序依次放入辅助数组b[ ],然后将数组b[ ]复制回数组a[ ]。最终,合并排序算法得到排好序的数组a[0:7]= {5,9,16,17,18,23,25,30}。   b)**代码:**  public static int[] MergeSort(int[] array) {  if (array.length < 2) return array;  int mid = array.length / 2;  int[] left = Arrays.copyOfRange(array, 0, mid);  int[] right = Arrays.copyOfRange(array, mid, array.length);  return merge(MergeSort(left), MergeSort(right));  }  /\*\*  \* 归并排序——将两段排序好的数组结合成一个排序数组  \*/  public static int[] merge(int[] left, int[] right) {  int[] result = new int[left.length + right.length];  for (int index = 0, i = 0, j = 0; index < result.length; index++) {  if (i >= left.length)  result[index] = right[j++];  else if (j >= right.length)  result[index] = left[i++];  else if (left[i] > right[j])  result[index] = right[j++];  else  result[index] = left[i++];  }  return result;  }  public static void display(int[] arr) {  for(int i=0;i<arr.length;i++) {  System.out.print(arr[i]+" ");  }  System.out.println();  }  public static void main(String[] args) {  int[] arr= {23,5,9,16,30,25,17,18};  System.out.println("原数组：");  display(arr);  mergeSort m=new mergeSort();  System.out.println("合并排序结果：");  m.display(m.MergeSort(arr));  }  **截图：**    合并排序算法将规模为n的问题分解成两个规模为n/2的子问题。  该算法时间复杂度满足以下递归方程：  当n>1时，T(n)=2T(n/2)+O(n)  当n=1时，T(1)=0(1) 其中，a=2 b=2 k=1  易得 a=bk  T(n)=O(nklogbn)=O(nlogn)  因此，合并排序算法在最坏情况下的时间复杂度为O(nlogn) | | | | | |
| 1. 实验总结（包括心得体会、问题回答及实验改进意见，可附页）   通过本次实验，我学习了递归与分治法的基本思想及基本原理、分治法求解问题的一般特征及步骤、以及分治法的设计方法及复杂性分析方法。  在实验过程中，我发现我的基础不是很牢固，在碰到问题时我一般会通过搜索网上资料来补充自己的知识。例如：以前调函数时，通常是new一个对象再调用；现在发现如果全部方法都是static,就可以不用new对象直接调用。 | | | | | |
| 六、教师评语  1、完成所有规定的实验内容，实验步骤正确，结果正确；  2、完成绝大部分规定的实验内容，实验步骤正确，结果正确；  3、完成大部分规定的实验内容，实验步骤正确，结果正确；  4、基本完成规定的实验内容，实验步骤基本正确，所完成的结果基本正确；  5、未能很好地完成规定的实验内容或实验步骤不正确或结果不正确。  6、其它：  评定等级：优秀 良好 中等 及格 不及格  教师签名：郭艺辉 | | | | | |