**河南开封科技传媒学院实验报告**

••••••••••••••••••••••••••••••••• 密 ••••••••••••••••••••••••••••••••• 封 ••••••••••••••••••••••••••••••••• 线 •••••••••••••••••••••••••••••••••

**20 -20 学年第 学期**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 姓名： | 刘东 | | 学号： | 2136101045 | 专业： | 软件工程 | |
| 所在学院： | | 信息工程学院 | | | 指导老师： | | 袁帅 |
| 实验时间： | | 2023/4/19 | | | 实验地点： | | 实验室 |
| 课程名称： | | 数据结构 | | | | | |
| 实验题目： | | 第八次实验 | | | | | |
| 实验目的： | | | | | | | |
| 实验内容：   1. 排序问题   给一个数组，如A[]={1，3，5，46，11，22，37，45，44，78，67，98}，用除了冒泡排序外的任意一种方法把此数组按降序排列。  思考题：超级水王问题  题目来源：  Tango是微软亚洲研究院的一个试验项目。研究院的员工和实习生们都很喜欢在Tango上面交流灌水。传说，Tango有一大“水王”，他不但喜欢发贴，还会回复其他ID发的每个帖子。坊间风闻该“水王”发帖数目超过了帖子总数的一半。如果你有一个当前论坛上所有帖子（包括回帖）的列表，其中帖子作者的ID也在表中，你能快速找出这个传说中的Tango水王吗？  题目描述：  给你一个数组，需要你确定其中元素的水王，水王的条件为出现的频率超过总数的一半。  如B[]= { 1, 1, 2, 3, 1, 1 };其中水王为“1”  C[]= { 1, 1, 1, 2, 4, 2, 2, 3, 3, 3, 5 } 无水王返回“-1”  示例：求D[]= { 1, 1, 1, 2, 4, 2, 2, 3, 3, 3, 5,2,3,2,2,2,2,2,2 } | | | | | | | |
| 实验代码：  #include<iostream>  #include<algorithm>  using namespace std;  void print(int a[], int n){  for(int j= 0; j<n; j++){  cout<<a[j] <<" ";  }  cout<<endl;  }  //合并  void merge(int arr[], int tempArr[], int left, int mid, int right) {  //标记左半区域的第一个未排序的元素  int l\_pos = left;  //标记右半区域的第一个未排序的元素  int r\_pos = mid + 1;  //临时数组元素的下标  int pos = left;    //合并  while (l\_pos <= mid && r\_pos <= right) {  if (arr[l\_pos] < arr[r\_pos])  tempArr[pos++] = arr[l\_pos++];  else  tempArr[pos++] = arr[r\_pos++];  }    //当一个区合并完全后，另一个区应直接插入  //左区插入  while (l\_pos <= mid) {  tempArr[pos++] = arr[l\_pos++];  }  //右区插入  while (r\_pos <= right) {  tempArr[pos++] = arr[r\_pos++];  }    //把临时的数组拷贝到arr[]  while (left <= right) {  arr[left] = tempArr[left];  left++;  }    }  //归并排序  //划分  void msort(int arr[], int tempArr[], int left, int right) {  //如果只有一个元素，那么就不需要继续划分  //只有一个元素的区域本身就是有序的，只需要被归并即可  if (left < right) {  int mid = (left + right) / 2;  //划分两个区域  msort(arr, tempArr, left, mid);  msort(arr, tempArr, mid + 1, right);  //合并两个区域  merge(arr, tempArr, left, mid, right);  }  }  //归并排序的入口  void mergeSort(int arr[], int n) {  //分配一个数组  int \*tempArr = new int[n];  if (tempArr) { //分配成功  msort(arr, tempArr, 0, n - 1);  delete []tempArr;  } else {  cout << "error: failed to allocate Memory" <<endl;  }  }  bool sw(int a, int b) {  return a > b;  }  int main() {  int arr[12] = {1,3,5,46,11,22,37,45,44,78,67,98};  print(arr, 12);  mergeSort(arr, 12);  cout<<"结果：";  reverse(arr, arr+12);      // sort(arr, arr+12, sw);    print(arr, 12);  return 0;  } | | | | | | | |
| 实验结果截图及实验心得： | | | | | | | |