《数据结构》实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 马昆 | 班级 | 22级软件工程专升本1班 | 学号 | 2206831544 |
| 实验名称 | **实验七 查找技术** | | | | |
| 实验时间 | 2023年5月16日 | | 成绩 | |  |
| 1. **实验目的：**   1.掌握顺序查找、折半查找算法的基本思想；  2.掌握顺序查找、折半查找算法的实现方法；  3.掌握顺序查找、折半查找算法的时间性能。  **二、实验工具：**  软件平台：Windows7或以上版本，Visual Studio 2019  **三、实验原理：**  1、算法描述：  （1）顺序查找函数：函数参数接收一个数组，一个数组长度，以及需要查找的整型值，返回查找到的元素的下标。定义一个for循环，循环的次数为数组的长度，循环体内判断被遍历到的元素是否与查找元素相同，相同则返回当前元素的下标。for循环结束还没有找到说明没有找到元素，则返回-1，表示没有找到。  （2）折半查找函数：函数参数接收一个数组，一个数组长度，以及需要查找的整型值，返回查找到的元素的下标。定义三个整型变量，用于记录查找元素的范围，分别为最小下标，最大下标，中间下标。定义一个while循环，当最小下标小于或等于最大下标时才执行循环体，循环体内将中间下标的值设为最小下标与最大下标的中间值，如果更新后的中间下标所对应的元素比查找元素小，那么最小下标更新为当前中间下标，如果更新后的中间下标所对应的元素比查找元素大，那么最大下标更新为当前中间下标，如果都不是，说明当前中间下标所对应的元素就是要查找的元素，则返回当前中间下标。如果while循环执行完毕方法还没有执行结束，说明没有找到元素，则返回-1。  **四、实验步骤和内容：**   1. 随机产生n（n>500）个整数，完成以下操作:   （1）对给定的查找集合，顺序查找与给定值k相等的元素,并返回查找所用时间；  （2）对给定的有序查找集合，折半查找与给定值k相等的元素,并返回查找所用时间；  部分参考代码如下：  #include<ctime> //使用函数库ctime  ……  int main()  {  clock\_t start,finish; //定义查找开始时间和结束时间变量  ……  start=clock(); //记录查找算法的开始时间  查找算法  finish=clock(); //记录查找算法的结束时间  ……  }   |  | | --- | | **Order.h**  #pragma once  // 冒泡排序-降序  void bubbleOrderDesc(int arr[], int len);  // 冒泡排序=升序  void bubbleOrderAsc(int arr[], int len); | | **Order.cpp**  #pragma once  // 冒泡排序-降序  void bubbleOrderDesc(int arr[], int len) {  int i, j, temp;  for (i = 0; i < len - 1; i++) {    for (j = 0; j < len - 1 - i; j++) {  if (arr[j] < arr[j + 1]) {  temp = arr[j + 1];  arr[j + 1] = arr[j];  arr[j] = temp;  }  }  }  }  // 冒泡排序-升序  void bubbleOrderAsc(int arr[], int len) {  int i, j, temp;  for (i = 0; i < len - 1; i++) {  for (j = 0; j < len - 1 - i; j++) {  if (arr[j] > arr[j + 1]) {  temp = arr[j + 1];  arr[j + 1] = arr[j];  arr[j] = temp;  }  }  }  } | | **Find.h**  #pragma once  // 顺序查找  int seqSearch(int arr[], int len, int search);  // 折半查找  int binSearch(int arr[], int len, int search); | | **Find.cpp**  #include "Find.h"  // 顺序查找  int seqSearch(int arr[], int len, int search) {  for (int i = 0; i < len; i++) {  if (arr[i] == search) {  return i;  }  }  return -1;  }  // 折半查找  int binSearch(int arr[], int len, int search) {  int low = 0, high = len, mid;    while (low <= high) {  mid = (low + high) / 2;  if (arr[mid] < search) {  low = mid;  }  else if (arr[mid] > search) {  high = mid;  }  else {  return mid;  }  }  return -1;  } | | **Main.cpp**  #include <iostream>  #include "Order.h"  #include "Find.h"  #include <ctime>  using namespace std;  const int MAX\_SIZE = 100000;  // 数组打印函数  void printArr(int arr[], int len) {  for (int i = 0; i < len; i++) {  if (!(i % 14)) {  cout << endl;  }  else {  cout << '\t';  }  cout << arr[i];  }  cout << endl;  }  int main()  {  int arr[MAX\_SIZE];  for (int i = 0; i < MAX\_SIZE; i++) {  arr[i] = i;  }  bubbleOrderAsc(arr, MAX\_SIZE);    clock\_t startTime, endTime;  int index, search = MAX\_SIZE - 1;  startTime = clock();  printArr(arr, MAX\_SIZE);  endTime = clock();  cout << "打印：" << (endTime - startTime) << "ms" << endl;  startTime = clock();  index = binSearch(arr, MAX\_SIZE, search);  cout << "折半查找" << search << "的下标为：" << index << endl;  endTime = clock();  cout << "折半查找：" << (endTime - startTime) << "ms" << endl;  startTime = clock();  index = seqSearch(arr, MAX\_SIZE, search);  cout << "顺序查找" << search << "的下标为：" << index << endl;  endTime = clock();  cout << "顺序查找：" << (endTime - startTime) << "ms" << endl;    return 0;  } | | **运行结果** |   **五、实验总结：**  通过本次实验，了解并学习了基础的排序算法以及查找算法，能够自行编写冒泡排序算法、顺序查找算法与折半查找算法。但是在计算顺序查找算法与折半查找算法所耗费的毫秒值是有一些问题，按理论来讲这两种算法的效率差距还是蛮大的，但是在实际执行时几乎没有区别，在测试数据量为10万时，折半查找与顺序查找的相差仅为1毫秒，可能是电脑过于强劲。  **六、教师评语：** | | | | | |