山东大学 计算机科学与技术 学院

数据结构与算法 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201805130155 | 姓名： 赵雨晗 | | 班级： 18计科3班 |
| 实验题目：递归练习 | | | |
| 实验学时：2 | | 实验日期： 2019.9.11 | |
| 实验目的：  1、 掌握各种简单排序算法。 | | | |
| 软件开发工具：  Windows : Vs Code + MingW | | | |
| 1. 实验内容   1、 创建排序类：数据含有n个整数，使用动态数组存储；提供操作：按名次排序、及时终止的选择排序、及时终止的冒泡排序、插入排序；  2、 键盘输入n, 随机生成n个0~1000之间的整数建立排序实例；输出各种排序算法的排序过程。  3、 统计每一种排序所耗费的时间（即比较次数和移动次数）。     1. 数据结构与算法描述 （整体思路描述，所需要的数据结构与算法）   1、选择排序第一次从待排序的数据元素中选出最小（或最大）的一个元素，存放在序列的起始位置，然后再从剩余的未排序元素中寻找到最小（大）元素，然后放到已排序的序列的末尾。以此类推，直到全部待排序的数据元素的个数为零。  2、冒泡排序重复地走访过要排序的元素列，依次比较两个相邻的元素，如果他们的顺序（如从大到小、首字母从A到Z）错误就把他们交换过来。走访元素的工作是重复地进行直到没有相邻元素需要交换，也就是说该元素列已经排序完成。  3、插入排序如果有一个已经有序的数据序列，要求在这个已经排好的数据序列中插入一个数，但要求插入后此数据序列仍然有序，这个时候就要用到一种新的排序方法——插入排序法,插入排序的基本操作就是将一个数据插入到已经排好序的有序数据中，从而得到一个新的、个数加一的有序数据，算法适用于少量数据的排序，   1. 测试结果（测试输入，测试输出） 2. 分析与探讨（结果分析，若存在问题，探讨解决问题的途径）   并无问题   1. 附录：实现源代码（本实验的全部源程序代码，程序风格清晰易理解，有充分的注释）   #include<cstdio>  #include<iostream>  #include<cstring>  #include<string>  #include<algorithm>  using namespace std;  template<class T> int task1(T \*a, T b, int n){ //模板类函数 0为边界返回  if (!n) return a[n] == b;  else return a[n] == b | task1(a, b, n - 1); //位运算取或  }  inline void task2(int n){ //一种位运算枚举子集的方法，可用于枚举任意一个二进制集合的子集  int tmp = (1 << n) - 1;//因为题目需要全枚举到 所以定义最大集合为 2^n - 1，注意位运算优先级最低  for (int i = tmp; i; i = (i - 1) & tmp) //不断减1， 并用位运算保证一定是子集  {  for (int j = 0; j < n; ++j){//输出方案  if (i & (1 << j)) printf("1"); // 判断当前位是否为1  else printf("0");  }  printf("\n");  }  }  inline void task2\_1(int i, int n){//把上一种方法改为递归版本  for (int j = 0; j < n; ++j){//输出方案  if (i & (1 << j)) printf("1"); // 判断当前位是否为1  else printf("0");  }  printf("\n");  if (i) task2\_1(i - 1, n);  }  int a[10];  int main()  {  cout<<task1(a, 0, 9)<<endl;  // task2(5);  task2\_1((1 << 5) - 1, 5);  system("pause");  } | | | |