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、 熟悉开发工具的使用。  2、 掌握递归的实现思想。 | | | |
| 软件开发工具：  Windows : Vs Code + MingW | | | |
| 1. 实验内容   1、 键盘输入n和n个互不相同的整数，输出n个整数的全排列。  2、 键盘输入n和n个互不相同的整数，输出n个整数的所有子集。   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.使用递归枚举全排列的方法，输出整体方案。  2.使用二进制枚举子集的方法，事实上可以非递归更优的解决子集或者任意集合子集的普遍性问题，但为了实验要求，改为递归做法。   1. 测试结果（测试输入，测试输出）   2 1 2  1 2 2 1    2 1  1 2 2  1 2   1. 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");  } | | | |