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：201700130009 | 姓名： 张愈博 | | 班级： 计科3 |
| 实验题目：递归练习 | | | |
| 实验学时： | | 实验日期： 2018.09.28 | |
| 实验目的：  1.熟悉开发工具的使用。  2.掌握递归的实现思想 | | | |
| 软件环境：  Windows 10  Dev c++ 5.11  Visual studio 2017 | | | |
| 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）   第一题运用来了递归的思想，首先取出n个数中的第一个，再递归全排列剩下的n-1 个数，再第二个，递归排列余下n-1个数，以此类推。N个整数用一个Int数组存储，这里我用了更加通用的算法，可求一个线性数组list[m:n]的全排列，更加具有实用性。  第二题同样运用了递归的思想，根据n个整数生成一个n维的0、1向量数组，当数组中i号元素为0时，放弃输出第i个整数；否则输出第i个整数。根据向量数组中的第i个元素为0或为1进行递归。   1. 测试结果（测试输入，测试输出，结果分析）          1. 分析与探讨（结果分析，若存在问题，探讨解决问题的途径）   第一次运行程序，我发现题目一没有问题，但是题目二出现了奇怪的现象，在传入函数Sub\_Generation之前，list的所有元素均正常，但是在其中输出时list[0]的值被莫名改变了（见图1）。为此我修改参数再次运行了一次，发现list[0]的值应该是被temp中元素的值所覆盖了（见图2），原因未知。经过多次检查，我的代码没有逻辑问题，下面猜想可能是编译器出的问题。  换用visual studio 2017重新编译该程序，发现结果正确（见图3）。虽然结果正确了，然而我仍然无法解释为什么在dev c++ 5.11的环境下会发生这种bug。  之后，我修改了main函数里第一行变量的申明顺序，竟然发现出现了图4的状况，这次是list[4]的值发生了变化。  由于本人知识所限，无法解释上述情况，还请老师指正。  OK，经过反复查找错误，发现最终原因在于main函数第一行的temp少申请了一位空间，list同理。非常细节的一个错误，现在程序已经可以正常编译运行了。   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<iostream>  using namespace std;  #define N 5  template <class T>  void permutations(T list[],int m,int n) //生成list[m:n]的全排列  {  if(m>n) cerr<<"math error!"<<endl;  if(m==n) { //LIST[M:N]仅有一个排列，输出它  /\*copy(list,list+n+1,ostream\_iterator<T>(cout,""));\*/  for(int i=0;i<=n;i++) { cout<<list[i]; }  cout<<" ";  }    else  for(int i=m;i<=n;i++)  {  swap(list[i],list[m]); //为什么出错？ call of overloaded 'swap(int&,int&) is ambiguous? std文件里已经定义了swap函数，你不要再写了  permutations(list,m+1,n);  swap(list[m],list[i]);  }  }  template <class T>  void Sub\_Generation(T list[],int temp[],int n,int pos=0) //there is n elements in the list. And temp用于储存0、1序列  {//生成n位的0、1序列表示list的元素是否在子集中 ,pos是代表初始位置  //下面是调试用代码  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  if(list[0]==0) cout<<"error!"<<endl;  for(int i=0;i<n;i++)  { cout<<list[i]<<" ";}  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/    if(pos==n) //前面跳出循环条件时pos已经等于n  { cout<<"{";  for(int i=0;i<n;i++)  {  if(temp[i]==1)  { /\*if(list[i]==0) list[i]=1;\*/ cout<<list[i]; //神bug.....  }    } cout<<"} "; //return;  }  else  {  temp[pos]=0;  Sub\_Generation(list,temp,n,pos+1);  temp[pos]=1;  Sub\_Generation(list,temp,n,pos+1);  }  }    int main()  {  //int i,list[N-1],temp[N-1]; 错误示范  int i,list[N],temp[N];  cout<<"请输入"<<N<<"个数字"<<endl;  for(i=0;i<N;i++) { cin>>list[i]; }  cout<<"下面是它的全排列:"<<endl;  permutations(list,0,N-1);  for(i=0;i<N;i++) { cout<<endl<<list[i]; } //这里检查发现list在传入第二个函数之前是正常的  cout<<endl;  cout<<"下面是它的子集:"<<endl;  Sub\_Generation(list,temp,N,0);  return 0;  } | | | |