**附件：**

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：202000130138 | 姓名：宋璎航 | | 班级： 20.3 |
| 实验题目：递归练习 | | | |
| 实验学时：2 | | 实验日期：2021/9/30 | |
| 实验目的：   1. 熟悉开发工具的使用。 2. 掌握递归的实现思想。 | | | |
| 软件开发环境：  VSC | | | |
| 1. 实验内容   1、题目描述：  现有一个有n个元素的序列a=[a1,a2,⋯,an]，定义这个序列的价值为。空序列的价值为0。先给你一个长度为n的序列a，求a中所有子集价值的异或和，要求子集中元素的相对位置保持不变。  异或和: 位运算的一种。如果a、b两个值不相同，则异或结果为1；如果a、b两个值相同，异或结果为0。  输入输出格式：  输入：第一行，一个整数n  接下来一行有n个非负整数：a1,a2,⋯,an  输出：一个整数，表示所有子集价值的异或和。  2、题目描述：  现有一个有n个元素的序列a=[a1,a2,⋯,an]，定义其价值为  给出这样一个序列，求其所有排列的价值vi或  其中∣为位运算或操作，⊕为位运算异或操作。  输入输出格式：  输入：输入的第一行是一个整数n (2<=n<=10)，表示需排列的数的个数。  接下来一行是n个整数，数的范围是0到100000，每两个相邻数据间用一个空格分隔。  输出：  一个整数，代表所有排列价值的或。   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） 2. 使用递归二叉树 3. 深度优先搜索 4. 测试结果（测试输入，测试输出）   1：输入：  2  1 2  输出：6  2：输入：  3  1 2 3  输出：6   1. 分析与探讨（结果分析，若存在问题，探讨解决问题的途径）   结果均正确   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）   1.  *#include* <bits/stdc++.h>  using namespace std;  int n;  bool ifexist[100000] = {0};  int a[100000];  int value = 0;  void Subset(int x)  {  *if* (x == n)      {          int temp = 0;          int cnt = 0;          int p = 1;  *while* (cnt < n)          {  *if* (ifexist[cnt])              {                  temp += p \* a[cnt];                  ++p;              }              ++cnt;          }          value ^= temp;  *return*;      }      ifexist[x] = 0;//第x个元素不包含      Subset(x + 1);      ifexist[x] = 1;//第x个元素包含      Subset(x + 1);  }  int main()  {      cin >> n;  *for* (int i = 0; i < n; ++i)          cin >> a[i];      Subset(0);      cout << value << endl;  }  2、  *#include* <bits/stdc++.h>  using namespace std;  int a[100000];  int b[100000];  bool ifexist[100000] = {0};  int n;  int value;  void output()  {      int temp = 0;  *for* (int i = 0; i < n; ++i)          temp += (i + 1) ^ a[i];      value |= temp;  }  void dfs(int x)*//深度优先搜索，从第x位往后的全排列情况*  {  *if* (x == n)      {          output();  *return*;      }  *for* (int i = 0; i < n; ++i)      {  *if* (ifexist[i] == 0)          {              ifexist[i] = 1;//第i个数位标为存在              a[i] = b[x];              dfs(x + 1);              ifexist[i] = 0;          }      }  }  int main()  {      cin >> n;  *for* (int i = 0; i < n; ++i)          cin >> b[i];      dfs(0);      cout << value << endl;  } | | | |