**《算法设计与分析》实验报告**

# 实验四 回溯法

|  |  |  |  |
| --- | --- | --- | --- |
| **报告书** | | | |
| 姓名 | 吴宇敖 | 指导教师 |  |
| 学号 | 176001752 | 日 期 | 2020.5.2 |
| 班级 | 17计算机软件一班 |  |  |
| **实验内容** | | | |
| **[要求先在OJ上提交通过]**  1)N皇后的排列，每行一个不冲突；N<=13。[OJ]  2)符号三角形的 第1行有n个由“+”和“-”组成的符号 ，以后每行符号比上行少1个，2个同号下面是“+”，2个异号下面是“-” 。计算有多少个不同的符号三角形，使其所含“+” 和“-” 的个数相同。[OJ] | | | |
| **实验目的** | | | |
| 1）完成两个题目  2）了解回溯算法 | | | |
| **实验过程和步骤** | | | |
| **以下仅供参考，请在报告中删除**  **【实验题目】**  **题目描述：**  **N皇后的排列，每行一个不冲突；N<=13。**  **输入要求：**  **一个数字N (6 <= N <= 13) 表示棋盘是N x N大小的。**  **输出要求：**  **前三行为前三个解，每个解的两个数字之间用一个空格隔开。第四行只有一个数字，表示解的总数。**  **解的输出顺序为从上到下从左到右，小的优先输出**  **【程序代码及注释】**  #include<iostream>  using namespace std;  int res[14];  int sum=0;  void find(int a,int n)  {  if(a==n+1)  {  sum++;//记录有几组  if(sum<=3)//输出  for(int i=1;i<=n;i++)  if(i==n)  cout<<res[i]<<endl;  else  cout<<res[i]<<" ";  }  else  {  for(int i=1;i<=n;i++)//遍历第i行的全部数  {  bool m=true;  for(int j=1;j<a;j++)//第i行前面的几行  if(res[j]==i||res[j]+a-j==i||res[j]-a+j==i)//判断两个皇后之间会不会互相相吃  {  m=false;  break;  }  if(m)  {  res[a]=i;//记录第几行应该放在那里  find(a+1,n);  }  }  }  }  int main()  {  int n;  cin>>n;  find(1,n);  cout<<sum<<endl;  //system("pause");  return 0;  }  **【实验结果及遇到的问题】**    **【算法分析】**  这个实验我是通过递归的方法，对于第一行先进行循环，然后再次调用这个函数，对第二行的每个位置进行判断，使与前面的几行放的皇后不冲突，然后记录，之后不断递归，到i行，对于i行的每一个位置与前面几行的皇后进行判断是否冲突，不冲突记录。最后当递归到第n+1行的时候，进行输出，  **【实验题目】**  **题目描述：**  **符号三角形的 第1行有n个由“+”和“-”组成的符号 ，以后每行符号比上行少1个，2个同号下面是“+”，2个异号下面是“-” 。计算有多少个不同的符号三角形，使其所含“+” 和“-” 的个数相同。**  **输入要求：**  **整数n**  **输出要求：**  **符合要求的符号三角形的个数**  **【程序代码及注释】**  #include<iostream>  using namespace std;  int num[25][25]={0};  int sum=0;  void find(int a,int n)  {  if(a==n+1)  {  for(int i=2;i<=n;i++)//从第二行开始，根据前一行的数据来进行符号的判断  for(int j=1;j<=n-i+1;j++)  if(num[i-1][j]==num[i-1][j+1])//相同为加号，异号为减号  num[i][j]=1;  else  num[i][j]=0;  int X=0,Y=0;  for(int i=1;i<=n;i++)//记录加号和减号的数量  for(int j=1;j<=n-i+1;j++)  if(num[i][j]==1)  X++;  else  Y++;  if(X==Y)//数量相同时，加一  sum++;  }  else  for(int i=0;i<=1;i++)//定义：1为加号 0为减号，把第一行的数通过递归全部赋值  {  num[1][a]=i;  find(a+1,n);  }  }  int main()  {  int n;  cin>>n;  find(1,n);  cout<<sum<<endl;  //system("pause");  return 0;  }  **【实验结果及遇到的问题】**    遇到问题：  （1）一开始，由于将数组初始化为0，所以在异或判断时，没有将不同的置为0，导致输出结果为0，最后才发现，由于递归需要多次判断，每次都需要置为0.  **【算法分析】**  这个实验，我先使用递归的方法来进行赋值，这样就可以讨论到所有的情况，之后从第二行开始，对于每一行的每个数，根据上一行来进行判断，相同为1，不同为0，之后在通过循环来记录全部的加号和减号的数量，如果相同，那么sum加一，最后输出sum | | | |