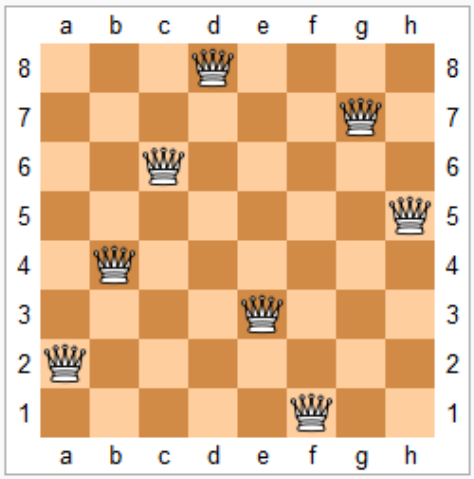
LeetCode：51. N-Queens（N 皇后问题）- Python

原创 GrowthDiary007 2019-02-02 12:59:40 1360 收藏 版权
分类专栏： 算法 Python LeetCode C++

51. N皇后

问题描述：

`n` 皇后问题研究的是如何将 `n` 个皇后放置在 `n×n` 的棋盘上，并且使皇后彼此之间不能相互攻击。



One solution to the eight queens puzzle

上图为 8 皇后问题的一种解法。

给定一个整数`n`，返回所有不同的 `n` 皇后问题的解决方案。

每一种解法包含一个明确的`n`皇后问题的棋子放置方案，该方案中 `Q` 和 `.` 分别代表了皇后和空位。

示例:

```
输入: 4
输出: 2
解释: 4 皇后问题存在如下两个不同的解法。
[
[".Q…", // 解法 1
“…Q”,
“Q…”,
“…Q.”],
```

```
["…Q.", // 解法 2
“Q…”,
```

```
“...Q”,
“.Q...”]
]
```

问题分析：

该有多无聊，才去重新写一个n年前的算法题目？这个第一次见是上数据结构课，递归的那一章，哈哈。总结一下：

（1）方法是回溯法，具体就是深度优先搜索。

（2）我们一行一行的放下去，在放的过程中，判断这个位置是否可以放？可以就放，不可以就进行下一个位置，到头了，就回溯到上一行。

（3）放完n个皇后，就输出一组结果。直到结束。

Python3实现：

方法一，递归的方法：

```python
# @Time   :2019/02/02
# @Author :LiuYinxing
# 回溯法，递归


class Solution:
    def solveNQueens(self, n):
        """
        :type n: int
        :rtype: int
        """
        res, q = [], [-1] * n   # cnt 用计数，q用于已经放的位置，例如q[2]=3 表示第3行
        
        def dfs(k, n):
            if k == n:
                tmp = []
                for i in range(n):  # 输出一个结果
                    s = ''
                    for j in range(n):
                        s += 'Q' if q[i] == j else '.'
                    tmp.append(s)
                res.append(tmp)
            else:
                for j in range(n):  # 一行一行的进行深度搜索
                    if self.place(k, j, q):
                        q[k] = j
                        dfs(k+1, n)
        dfs(0, n)
        return res
```

```
30
31      def place(self, k, j, q):  # 判断该位置是否可以放一个棋子
32          for i in range(k):
33              if q[i] == j or abs(q[i]-j) == abs(i-k):  # 不同列，不同斜线
34                  return 0
35          return 1
36
37
38  if __name__ == '__main__':
39      solu = Solution()
40      print(solu.solveNQueens(4))
41
```

方法二，栈的方法（这段 **C++** 代码是很久之前学数据结构时候的，这里只提供思路）：

```
1   #include <iostream>
2   #include <cmath>
3   using namespace std;
4   #define MaxSize 100
5   int count=1;                            //记录解个数
6   struct StType                           //定义顺序栈类型
7   {
8           int data[MaxSize];                      //data[i]存放第i个皇后的列号
9           int top;                                        //栈顶指针
10  };
11  bool place(StType st,int i,int j)       //测试(i,j)是否与1～i-1皇后有冲突
12  {
13          int k=1;
14          if (i==1)
15                  return true;                    //放第一个皇后时没有冲突
16          while (k<=i-1)                              //j=1到k-1是已放置了皇后的列
17          {
18                  if ((st.data[k]==j) || (abs(j-st.data[k])==abs(k-i)))
19                          return false;
20                  k++;
21          }
22          return true;
23  }
24  void queen(int n)                                               //求解n皇后问题
25  {
26          int i,j,k;
27          bool find;
28          StType st;                                                      //定义栈st
29          st.top=0;                                                       //初始化栈顶指针
30          st.top++;                                                       //将(1,1)进栈
31          st.data[st.top]=1;
32          while (st.top>0)                                        //栈不空时循环
33          {
```

```
34                  i=st.top;                                           //当前皇后为第i个
35                  if (st.top==n)                                  //所有皇后均放好,输出一个解
36                  {
37                          cout<<n<<"皇后问题第"<<count++<<"个解为:";
38                          for (k=1;k<=st.top;k++)
39                                  cout<<"("<<k<<" "<<st.data[k]<<")"<<" ";
40              cout<<endl;
41                  }
42                  find=false;
43                  for (j=1;j<=n;j++)
44                          if (place(st,i+1,j))    //在i+1行找到一个放皇后的位置(i+1,j
45                          {
46                                  st.top++;
47                                  st.data[st.top]=j;  //在i+1行找到一个放皇后的位置(i
48                                  find=true;          //如果为真，则返回继续，再此查找
49                                  break;
50                          }
51                  if (find==false)                            //在i+1行找不到放皇后的位置
52                  {
53                          while (st.top>0)
54                          {
55                                  if (st.data[st.top]==n)             //退栈
56                                          st.top--;
57                                  for (j=st.data[st.top]+1;j<=n;j++)      //在i行i
58                                          if (place(st,st.top,j))
59                                          {
60                                                  st.data[st.top]=j;
61                                                  break;
62                                          }
63                                  if (j>n)                                        //当前皇
64                                          st.top--;                           //退栈
65                                  else                                        //本行找
66                                          break;
67                          }
68                  }
69          }
70  }
71
72  int main()
73  {
74          int n;                                              //n存放实际皇后个数
75          cout<<"温馨提示输入的数尽量不要大于15否则CPU会累死的。"<<endl;
76          cout<<"请输入N皇后问题的N(N<20)值:"<<endl;
77          cin>>n;
78          if (n>20)
79                  cout<<"N值太大,本程序无法为你求解。"<<endl;
80          else
81          {       cout<<n<<"皇后问题求解如下:"<<endl;
82                  queen(n);
83          cout<<endl;
```

```
84          }
85          return 0;
86  }
```

声明： 总结学习，有问题或不当之处，可以批评指正哦，谢谢。

题目链接:https://leetcode-cn.com/problems/n-queens/