# 1 项目简介

八皇后问题是一个古老而著名的问题，是回溯算法的经典问题。该问题是十九世纪著名的数学家高斯在1850年提出的：在8\*8的国际象棋棋盘上，安放8个皇后，要求没有一个皇后能够“吃掉”任何其它一个皇后，即任意两个皇后不能处于同一行，同一列或者同一条对角线上，求解有多少种摆法。本实验拓展了N皇后问题，即皇后个数由用户输入。

# 2 程序说明

## 2.1 算法分析

以排列方阵的左上角为(0,0)建立坐标系，水平向右方向为x轴，竖直向下方向为y轴，建立一个坐标系以方便思考与表述：

先在(0,0)位置摆放皇后，接下来在下一行寻找符合规则的可以摆放下一个皇后的位置，即棋盘上任意两个皇后不能处于同一行，同一列或者同一条对角线上。若找到则继续下一行，否则退回到上一行调整上一个皇后的位置，那么循环结束的条件便是第0行已经全部试完，只能调整不存在的第-1行的皇后位置。

这是典型的回溯思想，但是考虑到老师在课堂所讲：递归在性能上存在不足，而且每增加一个皇后数量，摆放方式增长的数量级是巨大的，极易造成递归爆炸，因此，本程序的主体虽然采用了回溯的思想，但对递归利用循环结构进行了展开，仅仅是在判断正上方、左上方与右上方位置有没有皇后时用到了简易的递归。

说明：

①二维的方阵排列在本程序中是转换为一维string字符串来执行的；

②综合考虑程序运行时间以及每增加一个皇后，结果数量的增长量级，将皇后数量上限定在15；

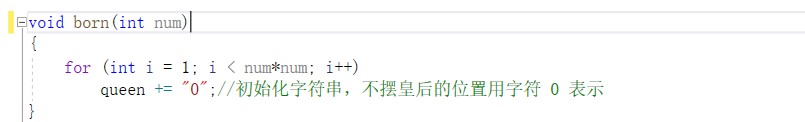
③由于本程序中是按照从上往下的顺序摆放皇后，所以在摆放一个皇后棋子时，只要考察正上方、左上方、右上方无皇后即可，无需考虑其下方位置。

④本程序中用字符 \* 表示该位置摆放的是皇后棋子，用字符 0 表示该位置不摆放皇后棋子

## 2.2 程序结构设计



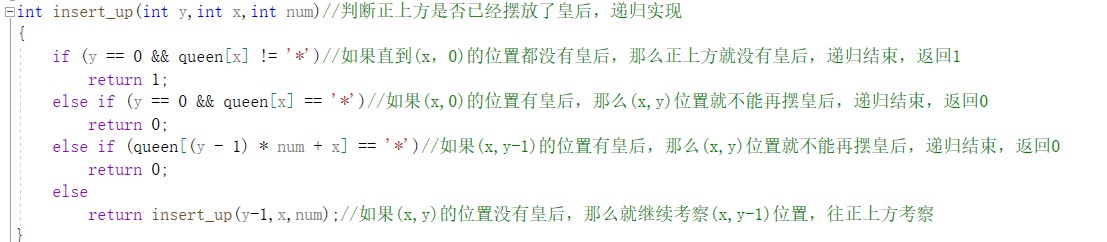
说明：本程序主体结构较为简单，由main函数调用4个功能函数配合输入（第一个红色方框内的while循环）与皇后位置排列（第二个方框内的for循环）完成。接下来首先会对4个功能函数进行说明，紧接着解读皇后位置排列的核心代码。

①born函数：生成num\*num长度的string字符串

born函数的作用在于生成num\*num长度的string字符串，这是因为二维的方阵排列在本程序中是转换为一维string字符串来执行的，目的是调用string类本身的find、length等字符操作函数，以提升程序性能。

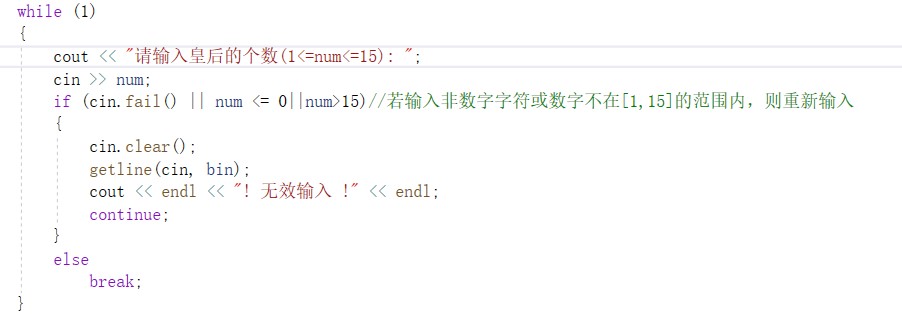
## ②insert\_up、insert\_left\_up、insert\_right\_up函数：

## 这三个函数分别用于放置皇后棋子时检测选定位置的正上方、左上方与右上方有无其他皇后棋子，若有，则返回0，若无，则返回1。这三个函数均采用简单的递归实现，思路一致，因此，此处以insert\_up函数为例进行说明：

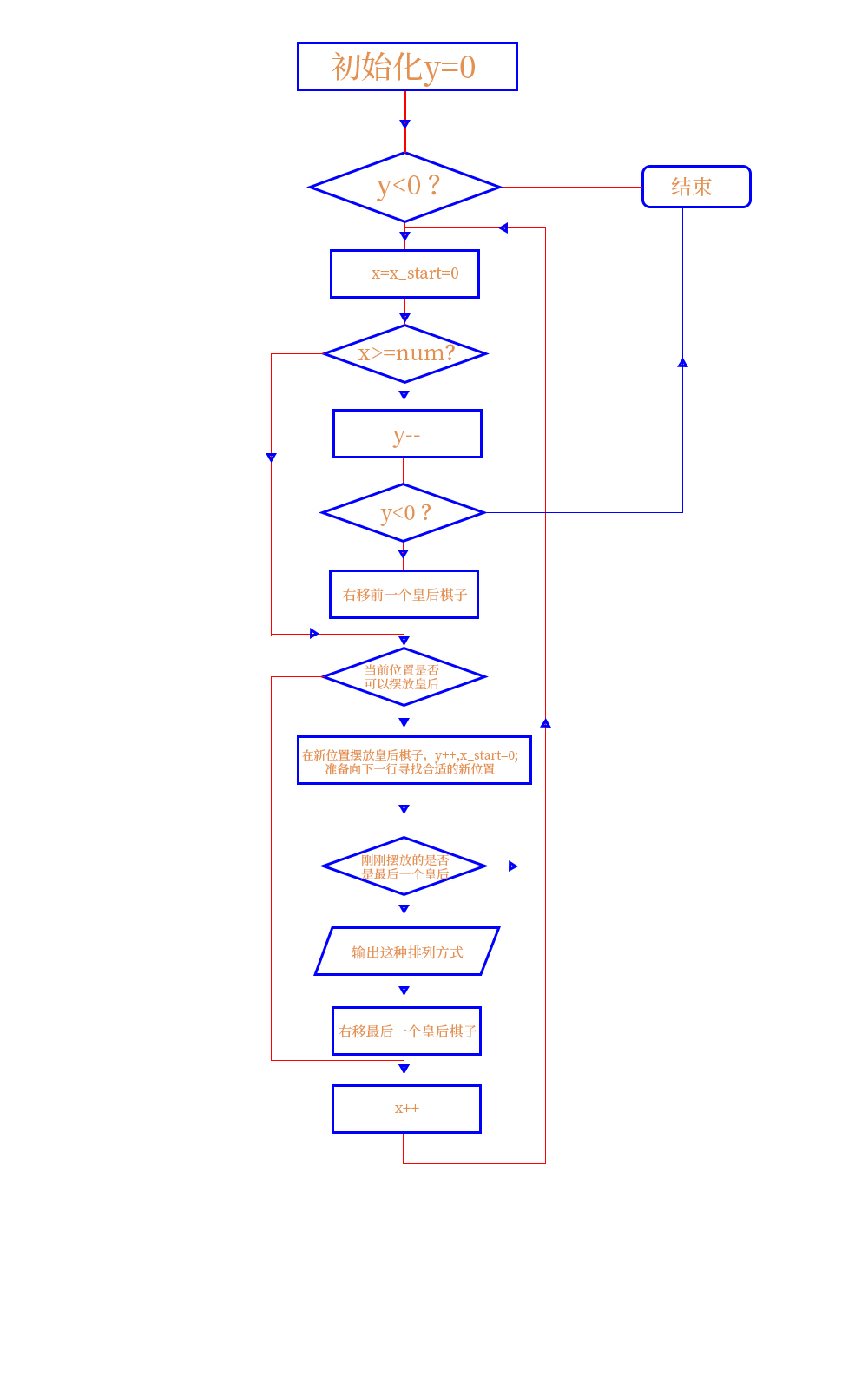


首先确定递归结束的三个条件：如果直到(x，0)的位置都没有皇后，那么正上方就没有其他皇后，递归结束，返回1；如果(x,0)的位置有其他皇后，那么当前(x,y)位置就不能再摆皇后，递归结束，返回0；如果(x,y-1)的位置有其他皇后，那么当前(x,y)位置就不能再摆皇后，递归结束，返回0。递归的过程是：如果当前(x,y)的位置没有皇后，那么就继续考察(x,y-1)位置，向正上方考察，直到递归结束。

这里需要特别注意的是，之所以把边界位置（x , 0）的有无其他皇后棋子单独分开写，而不是直接写成若无其他皇后棋子则继续考察正上方方向，是为了防止string越界，这一点很重要。

③main函数中输入皇后数量的while循环：有处理非法输入的功能

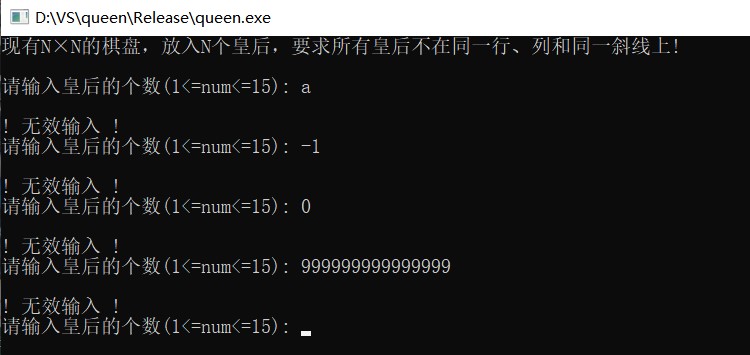
综合考虑程序运行时间以及每增加一个皇后，结果数量的增长量级，将皇后数量上限定在15（包括）。通过if判断，如果输入了非数字字符，会触发cin.fail( ) ,此外，当输入的数字不在[1,15]的范围内时，也会触发if分支的程序段，吃掉无效输入，并提示重新输入皇后个数，如果未触发if分支，则会执行else分支，跳出循环。

**3.皇后排列算法核心代码流程（非递归循环结构、流程图）**

思路说明：首先明确循环终止条件：y<0意味着需要调整不存在的-1行的皇后棋子的摆放位置，接下来主要进行两个if判断与相应操作：第一个if分支中：若x>=num,则表示第y行目前已近不存在可以摆放皇后的位置，应当回退到y-1行调整该行的皇后位置。而通过程序语句实现皇后位置调整，需要两步操作，先调用string本身的find函数，定位上一行皇后的位置，在用字符0覆盖原来的字符\*。第二个if分支是对当前行皇后位置的具体摆放操作，首先调用insert\_up、insert\_left\_up与insert\_right\_up函数定位当前行可以摆放皇后棋子的位置。如果恰好摆放的是最后一个皇后棋子，那么需要对当前的摆放方式进行输出操作，并且count加1。之后考察当前皇后棋子是否可以继续通过右移操作找到合适的位置，若不能则向上调整之前的皇后棋子的位置。

**4.测试**

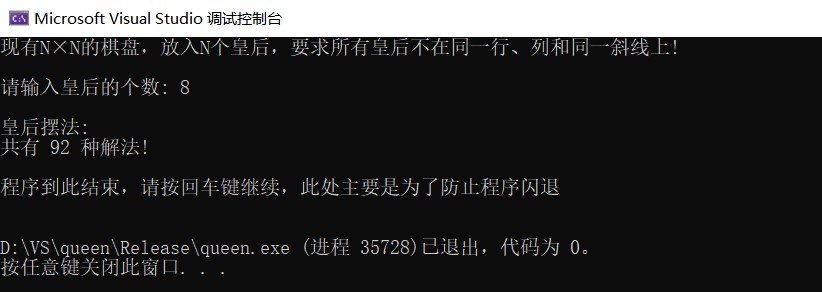
**① 非法输入处理功能测试**



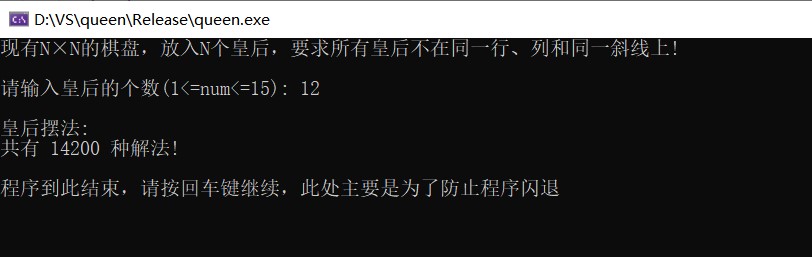
②正常功能测试（限于篇幅，仅6皇后给出详细排列，8皇后与12、15皇后只给出排列的方式数量）

6皇后共有4种排列方式

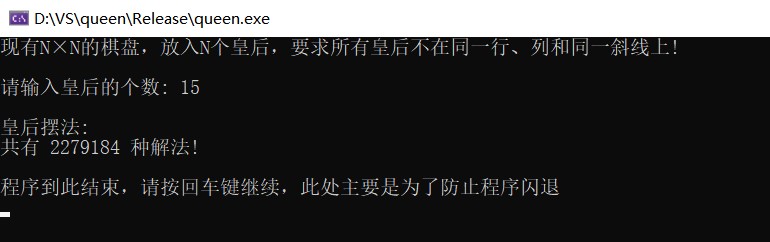


8皇后共有92种排列方式

12皇后共有14200种排列方式



15皇后共有2279184种排列方式



备注：已通过网上查找相关资料验证过上述结果