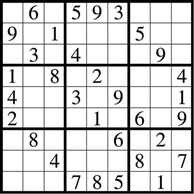
**数独**（Sudoku）是一种运用纸、笔进行演算的逻辑游戏。玩家需要根据9×9盘面上的已知数字，推理出所有剩余空格的数字，并满足每一行、每一列、每一个粗线宫内的数字均含1-9，不重复。 每一道合格的数独谜题都有且仅有唯一答案，推理方法也以此为基础，任何无解或多解的题目都是不合格的。

如下图所示，就是一个数独的题目

[](https://images0.cnblogs.com/blog/93227/201306/18213333-19b0575f757948dc9296fae5e9ad4696.jpg)

关于数独的详细介绍，参看“[百度百科——数独](http://baike.baidu.com/view/961.htm" \t "_blank)”

数独的基本解法就是利用规则的**摒弃法**

**一些定义**

每一行称为数独的**行**，每一列称为数独的**列**，每一个小九宫格称为数独的**宫**。数独的基本规则就是每一行、每一列、每一宫中，1-9这9个数字都只出现一次。

用（行，列）表示上图的单元格，例如（1，1）表示第一行第一列的单元格，（2，4）表示第二行第四列的单元格

如上图，每个空白单元格中能填的数字都是有限制的。

例如：（1，1）就只能填7和8；而（6，4），只能填8；

那些只能填一个数字的空白单元格，我们称之为**唯一数单元格**，上图中（6，4）就是**唯一数单元格**

解题的顺序，就是从唯一数单元格开始，由于唯一数单元格只能填一个数，故先在这个单元格里填数。在这个单元格里填数，由于规则的定义，那么这个单元格所在的行、所在的列、所在的宫的其他单元格就不能再填这个数了。这些单元格能填的数的可能性就少了。有可能会产生新的**唯一数单元格**。

在相当的一些的数独题目中，从唯一数单元格开始填数，不停的在**唯一数单元格**填数就可以把数独解出来。

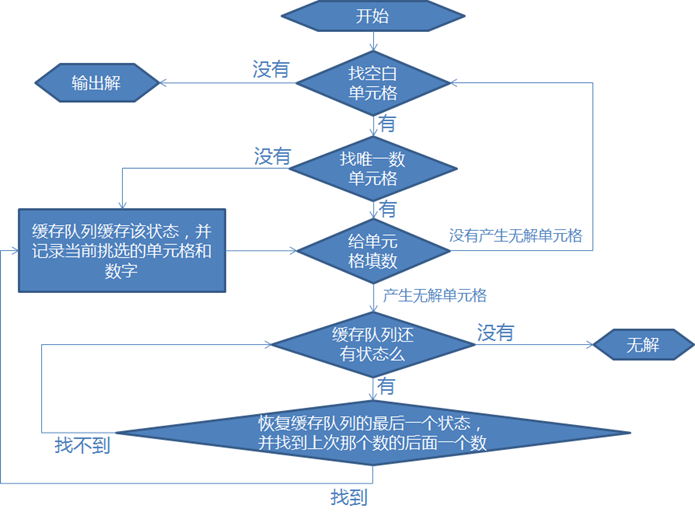
如果在解题的过程中，发现某些空白单元格没有数字能填这样的单元格称之为**无解单元格**，那就说明：要么这个数独没有解；要么之前的解题过程有问题，需要返回检查之前的解题过程查看。

但是还有不少的数独的题目，在解题的过程中，在还有空白单元格的情况下，却找不到**唯一数单元格**，也就是意味着每个空白单元格中能填的数字至少有2个。我们称之为**无唯一数单元格**的状况

这个时候怎么办？我们找到其中一个**可能数最少的空白单元格**（这个没有定论，可以是可能数最少的空白单元格；也可以是第一个空白单元格；也可以是可能数最多的空白单元格，选哪个空白单元格对后面的解题是否有影响，没有证明过，不好妄下定论。凭感觉选**可能数最少的空白单元格**是最好的选择），由于能填的数字不止一个，先把当前的状态保存起来，再在能选的数字中选择一个数字填写（从小到大选择），然后继续求解下去。如果能解出最后的结果，说明当前的选择是正确的；如果后面的求解过程有问题，说明当前的数字的选择有问题，那么再挑选另一个数填写，继续求解。如果，所有的选择都求不出最后的结果，还是说明：要么这个数独没有解；要么之前的解题过程有问题，需要返回检查之前的解题过程查看。如此反复，直到求出最终的答案。

会有种极端的情况（可能性不大）。那就是在当前的空白单元格的所有可能的数字都选择了一遍，都没有解。而之前又没有出现**无唯一数单元格**的状况。那就说明这个数独根本就没有解

下图是**数独求解的流程图**

[](https://images0.cnblogs.com/blog/93227/201306/17132251-441e8728896d40078263ae86d39a2b49.png)

下面谈谈该算法的具体实现

1. **数独状态的表示**
2. **如何获得一个单元格的可填数的个数**

由于是用二进制来表示单元格的状态，那么可填数的个数就是该数字中1的个数。我们之前有一个很方便的方法快速计算一个数中1的个数，参看[算法的强大——快速计算一个正二进制整数中包含多少个1](http://www.cnblogs.com/grenet/archive/2011/06/10/2077228.html" \t "https://www.cnblogs.com/grenet/p/_blank)。

1. **状态的缓存**

依据之前的说法，在碰到**无唯一数单元格**的情况时，要把当前的状态缓存起来。考虑到实际情况，从算法的角度上来说，用栈（先进后出）这个数据结构来实现比较合适。可以自己写一个栈的实现。但是，目前很多的编程语言都实现了基本的数据结构，提供了基本的数据结构的类和方法供我们调用。

以Visual Studio为例，它有Stack这个类，实现了栈的基本操作。有两个栈的方法：Push（压栈）——把数据写到栈里面；Pop（出栈）——把数据从栈里提出来，并删除栈中的数据。

1. **代码说明**