



算法基础

郭炜 微博 <http://weibo.com/guoweiofpku>

<http://blog.sina.com.cn/u/3266490431>

刘家瑛 微博 <http://weibo.com/pkuliujiaying>



北京大学
PEKING UNIVERSITY

信息科学技术学院《算法基础》 郭炜 刘家瑛

深度优先搜索

Sudoku

Sudoku (POJ2676)

1		3				5		9
		2	1		9	4		
			7		4			
3			5		2			6
	6						5	
7			8		3			4
			4		1			
		9	2		5	8		
8		4				1		7

将数字1到9, 填入9x9矩阵中的小方格, 使得矩阵中的每行, 每列, 每个3x3的小格子内, 9个数字都会出现。

输入输出

Sample Input

1
103000509
002109400
000704000
300502006
060000050
700803004
000401000
009205800
804000107

Sample Output

143628579
572139468
986754231
391542786
468917352
725863914
237481695
619275843
854396127

解题思路

几乎纯暴力搜索

将空白格子的位置放入一个数组，然后用Dfs尝试每个空白格子所放的数字。

剪枝：

放入一个数字后，就要做个标记，表面在当前行，当前列，以及当前小块已经放过这个数字了，那么以后就不会在同行，同列或同小块放同样数字

```

#include <iostream>
#include <vector>
#include <algorithm>
#include <cstring>
using namespace std;

short rowFlags[9][10]; //rowFlag[i][num] = 1表示在 第 i行 已经放了数字num
short colFlags[9][10]; //colFlag[i][num] = 1表示在 第 i列 已经放了数字num
short blockFlags[9][10]; //blockFlag[i][num]=1表示在第i个块已经放了数字num
int board[9][9]; //整个棋盘
struct Pos {
    int r,c;
    Pos(int rr,int cc):r(rr),c(cc) { }
};
vector<Pos> blankPos; //所有空白格的位置
inline int GetBlockNum(int r,int c)
{ //由行，列号求小块号
    int rr = r / 3;
    int cc = c / 3;
    return rr * 3 + cc;
}

```

```
void SetAllFlags(int i,int j, int num,int f)
{ // 把num放在 (i,j)位置, 设置相应标记, 或从 (i,j)取走num, 清除相应标记
    rowFlags[i][num] = f;
    colFlags[j][num] = f;
    blockFlags[GetBlockNum(i,j)][num] = f;
}

bool IsOk(int i,int j,int num)
{ // 看num能否放在 i,j位置
    return !rowFlags[i][num] && !colFlags[j][num] &&
           ! blockFlags[GetBlockNum(i,j)][num];
}
```

```
bool Dfs(int n)
{ //处理前n个空格
    if( n < 0)
        return true;
    int r = blankPos[n].r;
    int c = blankPos[n].c;
    for( int num = 1; num <= 9; ++ num ) {
        if( IsOk(r,c,num) ) {
            board[r][c] = num;
            SetAllFlags(r,c,num,1);
            if( Dfs(n-1) )
                return true;
            SetAllFlags(r,c,num,0);
        }
    }
    return false;
}
```



```

int main()
{
    int t;
    cin >> t;
    while( t -- ) {
        memset( rowFlags,0,sizeof(rowFlags));
        memset( colFlags,0,sizeof(colFlags));
        memset( blockFlags,0,sizeof(blockFlags));
        blankPos.clear();
        for( int i = 0; i < 9; ++i )
            for( int j = 0; j < 9; ++ j ) {
                char c;
                cin >> c;
                board[i][j] = c - '0' ;
                if( board[i][j] )
                    SetAllFlags(i,j,board[i][j],1);
                else
                    blankPos.push_back(Pos(i,j));
            }
    }
}

```

```
if( Dfs( blankPos.size()-1)) {  
    //倒着搜就很快，正着搜就慢，数据问题  
    for( int i = 0; i < 9; ++i ) {  
        for( int j = 0; j < 9; ++ j )  
            cout << char(board[i][j]+'0');  
        cout << endl;  
    }  
}  
}  
}
```