

C++解數獨

程式目標：

- 以0代替空格輸入數獨，輸出數獨解答

運行結果：

```
9 0 0 0 0 4 0 0 1
0 1 6 0 0 8 0 7 0
0 0 0 0 0 0 0 8 0
7 4 0 3 0 5 0 0 0
0 0 0 0 1 0 0 0 0
0 0 0 7 0 2 0 3 8
0 5 0 0 0 0 0 0 0
0 7 0 2 0 0 3 5 0
1 0 0 6 0 0 0 0 9

9 8 7 5 3 4 2 6 1
3 1 6 9 2 8 4 7 5
4 2 5 1 7 6 9 8 3
7 4 8 3 6 5 1 9 2
2 6 3 8 1 9 5 4 7
5 9 1 7 4 2 6 3 8
8 5 2 4 9 3 7 1 6
6 7 9 2 8 1 3 5 4
1 3 4 6 5 7 8 2 9
請按任意鍵繼續 . . .
```

測試資料_01

```
0 4 6 9 0 3 0 0 0
0 0 3 0 5 0 0 6 0
9 0 0 0 0 2 0 0 3
0 0 5 0 0 6 0 0 0
8 0 0 0 0 0 0 1 0
0 1 0 7 8 0 2 0 0
0 0 0 0 0 0 0 5 0
0 8 1 3 0 0 0 0 7
0 0 0 8 0 0 1 0 4

1 4 6 9 7 3 5 8 2
7 2 3 4 5 8 9 6 1
9 5 8 6 1 2 4 7 3
3 7 5 1 2 6 8 4 9
8 9 2 5 3 4 7 1 6
6 1 4 7 8 9 2 3 5
4 6 7 2 9 1 3 5 8
2 8 1 3 4 5 6 9 7
5 3 9 8 6 7 1 2 4
請按任意鍵繼續 . . .
```

測試資料_02

```
2 0 0 6 0 0 0 5 0
0 8 0 9 0 2 0 0 0
0 0 0 0 4 0 6 0 0
0 0 0 4 0 9 0 0 6
5 0 0 0 0 0 0 9 0
0 9 0 2 0 0 1 4 0
0 0 1 0 0 0 0 0 3
0 4 0 3 1 0 0 0 0
0 2 0 0 0 0 0 0 8

2 1 9 6 7 3 8 5 4
4 8 6 9 5 2 7 3 1
7 3 5 1 4 8 6 2 9
1 7 2 4 3 9 5 8 6
5 6 4 7 8 1 3 9 2
8 9 3 2 6 5 1 4 7
6 5 1 8 2 4 9 7 3
9 4 8 3 1 7 2 6 5
3 2 7 5 9 6 4 1 8
請按任意鍵繼續 . . .
```

測試資料_03

1.如何判斷填入的一個數字是否符合要求

範圍	條件A	條件B
X相同的九個數字	互不相同	在1~9中
Y相同的九個數字	互不相同	在1~9中
劃分在同一個小格子的九個數字	互不相同	在1~9中

2. 初始化數據

	0	1	2	3	4	5	6	7	8
0	0	1	2	3	4	5	6	7	8
1	9	10	11	12	13	14	15	16	17
2	18	19	20	21	22	23	24	25	26
3	27	28	29	30	31	32	33	34	35
4	36	37	38	39	40	41	42	43	44
5	45	46	47	48	49	50	51	52	53
6	54	55	56	57	58	59	60	61	62
7	63	64	65	66	67	68	69	70	71
8	72	73	74	75	76	77	78	79	80

- 第一格從0號開始，直到最後一格編號為80
- 我將編號定為int **count**變數

2-A. 根據count的值算坐標

- 在觀察數據後可以推得
- $y = \text{count} / 9;$
- $x = \text{count} \% 9;$
- //後面有寫程式驗證公式正確

2-B. 處理小格子

九個格子一組(小格子)，以**小格子左上角的數字為基準**
(在小格子內數字轉換完後都會變成該格左上角的坐標)

0	1	2	3	4	5	6	7	8	9
1	0	1	2	3	4	5	6	7	8
2	9	0,0	11	12	0,1	14	15	0,2	17
3	18	19	20	21	22	23	24	25	26
4	27	28	29	30	31	32	33	34	35
5	36	1,0	38	39	1,1	41	42	1,2	44
6	45	46	47	48	49	50	51	52	53
7	54	55	56	57	58	59	60	61	62
8	63	2,0	65	66	2,1	68	69	2,2	71
9	72	73	74	75	76	77	78	79	80

每個小格子的坐標以(y,x)表示

一個小格子

小格子的坐標
小格子左上角那格的坐標

(0,0)	(0,1)	(0,2)
(0,0)	(0,3)	(0,6)
(1,0)	(1,1)	(1,2)
(3,0)	(3,3)	(3,6)
(2,0)	(2,1)	(2,2)
(6,0)	(6,3)	(6,6)

//左上角坐標定為(sy,sx)

在觀察數據後可以發現
(y/3,x/3)是小格子的坐標

而小格子坐標再*3就會是小格子左上角的坐標

最終推導得出：

sy=x / 3 * 3;

sx=x / 3 * 3;

//後面有寫程式驗證公式正確

2-C. 寫程式驗證上述公式(程式碼在下一頁)

以count轉換y,x後的測試輸出

(0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)	(0,6)	(0,7)	(0,8)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)	(1,8)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)	(2,7)	(2,8)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)	(3,7)	(3,8)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)	(4,7)	(4,8)
(5,0)	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)	(5,7)	(5,8)
(6,0)	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)	(6,7)	(6,8)
(7,0)	(7,1)	(7,2)	(7,3)	(7,4)	(7,5)	(7,6)	(7,7)	(7,8)
(8,0)	(8,1)	(8,2)	(8,3)	(8,4)	(8,5)	(8,6)	(8,7)	(8,8)

以y,x轉換sy,sx後的測試輸出

(0,0)	(0,0)	(0,0)	(0,3)	(0,3)	(0,3)	(0,6)	(0,6)	(0,6)
(0,0)	(0,0)	(0,0)	(0,3)	(0,3)	(0,3)	(0,6)	(0,6)	(0,6)
(0,0)	(0,0)	(0,0)	(0,3)	(0,3)	(0,3)	(0,6)	(0,6)	(0,6)
(3,0)	(3,0)	(3,0)	(3,3)	(3,3)	(3,3)	(3,6)	(3,6)	(3,6)
(3,0)	(3,0)	(3,0)	(3,3)	(3,3)	(3,3)	(3,6)	(3,6)	(3,6)
(3,0)	(3,0)	(3,0)	(3,3)	(3,3)	(3,3)	(3,6)	(3,6)	(3,6)
(6,0)	(6,0)	(6,0)	(6,3)	(6,3)	(6,3)	(6,6)	(6,6)	(6,6)
(6,0)	(6,0)	(6,0)	(6,3)	(6,3)	(6,3)	(6,6)	(6,6)	(6,6)
(6,0)	(6,0)	(6,0)	(6,3)	(6,3)	(6,3)	(6,6)	(6,6)	(6,6)

完整輸出

(0,0)	(0,1)	(0,2)	(0,3)	(0,4)	(0,5)	(0,6)	(0,7)	(0,8)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)	(1,8)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)	(2,7)	(2,8)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)	(3,7)	(3,8)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)	(4,7)	(4,8)
(5,0)	(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)	(5,7)	(5,8)
(6,0)	(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)	(6,7)	(6,8)
(7,0)	(7,1)	(7,2)	(7,3)	(7,4)	(7,5)	(7,6)	(7,7)	(7,8)
(8,0)	(8,1)	(8,2)	(8,3)	(8,4)	(8,5)	(8,6)	(8,7)	(8,8)
(0,0)	(0,0)	(0,0)	(0,3)	(0,3)	(0,3)	(0,6)	(0,6)	(0,6)
(0,0)	(0,0)	(0,0)	(0,3)	(0,3)	(0,3)	(0,6)	(0,6)	(0,6)
(0,0)	(0,0)	(0,0)	(0,3)	(0,3)	(0,3)	(0,6)	(0,6)	(0,6)
(3,0)	(3,0)	(3,0)	(3,3)	(3,3)	(3,3)	(3,6)	(3,6)	(3,6)
(3,0)	(3,0)	(3,0)	(3,3)	(3,3)	(3,3)	(3,6)	(3,6)	(3,6)
(3,0)	(3,0)	(3,0)	(3,3)	(3,3)	(3,3)	(3,6)	(3,6)	(3,6)
(6,0)	(6,0)	(6,0)	(6,3)	(6,3)	(6,3)	(6,6)	(6,6)	(6,6)
(6,0)	(6,0)	(6,0)	(6,3)	(6,3)	(6,3)	(6,6)	(6,6)	(6,6)
(6,0)	(6,0)	(6,0)	(6,3)	(6,3)	(6,3)	(6,6)	(6,6)	(6,6)

```

int main()
{
    int x, y;
    //nextl 是拿來換行的
    int nextl = 1;
    for (int i = 0; i <= 80; i++)
    {
        x = i % 9;
        y = i / 9;
        cout << "(" << y << "," << x << ")" ";
        nextl++;
        if (nextl == 10)
        {
            nextl = 1;
            cout << endl;
        }
    }
    int sx, sy;
    cout << endl;
    for (int i = 0; i <= 80; i++)
    {
        x = i % 9;
        y = i / 9;
        sx = (x / 3) * 3;
        sy = (y / 3) * 3;
        cout << "(" << sy << "," << sx << ")" ";
        nextl++;
        if (nextl == 10)
        {
            nextl = 1;
            cout << endl;
        }
    }
    return 0;
}

```

3. Test函數(用來檢測當前數據是否符合要求)

- 先把count丟進來算一次y,x
- 水平檢查(如果同一列有一樣的數字就return false)
- 垂直檢查(如果同一行有一樣的數字就return false)
- 小格子內檢查(如果小格子內有一樣的數字就return false)
- 如果上述三項檢查均通過才會return true

```
bool Test(int count)
```

```
{
```

```
    int y = count / 9;
```

```
    int x = count % 9;
```

```
    //水平檢查
```

```
    for (int i = 0; i < 9; ++i)
```

```
        if (sudoku[y][i] == sudoku[y][x] && i != x)
```

```
            return false;
```

```
    //垂直檢查
```

```
    for (int i = 0; i < 9; ++i)
```

```
        if (sudoku[i][x] == sudoku[y][x] && i != y)
```

```
            return false;
```

```
    //小格檢查
```

```
    int sy = y / 3 * 3;
```

```
    int sx = x / 3 * 3;
```

```
    for (int i = sy; i < sy + 3; ++i)
```

```
        for (int k = sx; k < sx + 3; ++k)
```

```
            if (sudoku[i][k] == sudoku[y][x] && i != y && k != x)
```

```
                return false;
```

```
    //default
```

```
    return true;
```

```
}
```

4. guess函數(使用DFS來試數字)

```
void guess(int count)
{
    //格子最後一號只有到80,當count==81時表示已經把答案跑出來了
    //此刻直接印出
    if (count == 81)
    {
        for (int i = 0; i < 9; ++i)
        {
            for (int j = 0; j < 9; ++j)
                cout << sudoku[i][j] << " ";
            cout << endl;
        }
        return;
    }
    int y = count / 9;
    int x = count % 9;
    if (sudoku[y][x] == 0) //給這一格是空格
    {
        for (int i = 1; i <= 9; ++i)
        {
            //給這一格值,同時該格非0表示該格已經使用過了(標記已使用)
            sudoku[y][x] = i;
            //測試這個值放在這一格是不是符合要求(丟進Test函數進行測試)
            if (Test(count))
                //通過的話就進入下一格
                guess(count + 1);
        }
        //9個數字都嘗試完了可是不符合
        //重置該點數值為0讓他回到上一層
        sudoku[y][x] = 0;
    }
    //這一格有數字
    else
        guess(count + 1);
}
```

C++解數獨完整程式碼

```
#include <iostream>
using namespace std;
int sudoku[9][9];

bool Test(int count)
{
    int y = count / 9;
    int x = count % 9;
    //水平檢查
    for (int i = 0; i < 9; ++i)
        if (sudoku[y][i] == sudoku[y][x] && i != x)
            return false;
    //垂直檢查
    for (int i = 0; i < 9; ++i)
        if (sudoku[i][x] == sudoku[y][x] && i != y)
            return false;
    //小格檢查
    //小格左上角那一格的x,y坐標為sx,sy
    int sy = y / 3 * 3;
    int sx = x / 3 * 3;
    for (int i = sy; i < sy + 3; ++i)
        for (int k = sx; k < sx + 3; ++k)
            if (sudoku[i][k] == sudoku[y][x] && i != y && k != x)
                return false;
    //default
    return true;
}
```


//count是格子的編號,從左上角開始左往右數,0號到80號

```
void guess(int count)
```

```
{  
    //因為格子最後一號只有到80,當count==81時表示已經把一個答案跑出來了  
    //此刻直接印出  
    if (count == 81)  
    {  
        for (int i = 0; i < 9; ++i)  
        {  
            for (int j = 0; j < 9; ++j)  
                cout << sudoku[i][j] << " ";  
            cout << endl;  
        }  
        return;  
    }  
    int y = count / 9;  
    int x = count % 9;  
    if (sudoku[y][x] == 0)  
    {  
        for (int i = 1; i <= 9; ++i)  
        {  
            //給這一格一個值,同時該格非0表示該格已經使用過了  
            sudoku[y][x] = i;  
            //測試這個值放在這一格是不是符合要求(丟進Test函數進行測試)  
            if (Test(count))  
                //通過的話就進入下一格  
                guess(count + 1);  
        }  
        //9個數字都嘗試完了可是不符合  
        //先重置該點數值為0再讓他回到上一層  
        sudoku[y][x] = 0;  
    }  
    //這一格本來就有數字  
    else  
        guess(count + 1);  
}
```

```
int main()
{
    for (int i = 0; i < 9; ++i)
        for (int j = 0; j < 9; ++j)
            cin >> sudoku[i][j];
    cout << endl;
    //第一格是0號，count代0進入開始跑
    guess(0);
    system("PAUSE");
    return 0;
}
```