1101 計算機概論作業 7
繳交期限：12 月 24 日星期五下午 5:00

題目：在這個作業中，你將寫一支程式在一個二維矩陣中填入數字。程式必須使用下列 structure。

```
const int DIMENSION = 3;
struct sGame  {
   int x, y;
   int board[DIMENSION][DIMENSION];
};
```

這個 structure 的各欄位說明如下：board 是一個方陣。如上面的說明，當 DIMENSION 之值設定為 3 的時候，board 即為一個 3x3 的方陣。你可以想像有一個人在填這個方陣，而 x 與 y 即為這個人「現在正在填的位置」的座標。例如，若 x 與 y 的值都是 1，即可視為「現在正在填 board[1][1]這格。

請注意：這支程式有數個函數，各函數的功能都是獨立的，寫程式的過程中，應該要在完成一個函數之後，做好完整的測試，確認函數的功能都是正確的，之後再去寫下一個函數。絕對不可一口氣將所有函數全部寫完成再開始做測試。這樣會很難寫、很難偵錯。

程式請使用下列的 main()函數，不要更動。其中所呼叫的各函數將在後面逐一說明。

```
void main()
{
    struct sGame game;
    game = Initialize();
    do {
        game = TryTrySee(game);
        game = NextCell(game);
    } while (game.y < DIMENSION);
    Output(game);
    return;

}
```

Initialize()

這個函數必須具備以下功能：

1. 宣告一個 `struct sGame` 變數，名稱不限（以下假設名稱為 game）。

2. 將 game.x 以及 game.y 均設定初始值為 0。

3. 將 game.board 的所有元素之值均設為 0。

4. 回傳 game。

`struct sGame NextCell(struct sGame game)`

這個函數必須具備以下功能：

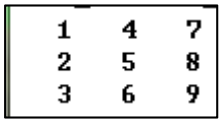
1. 將 game.x 的值增加 1。

2. 檢查 game.x 的值是否等於 DIMENSION。如果是，就將 game.x 的值設定為 0，同時將 game.y 的值增加 1。

也就是說，這個函數的功能是將「現在正在填的位置」移到下一格。如果已經是在這行的最後一格，則移到下一行的第 0 格。

`void Output(struct sGame game)`

將 game.board 的內容，以方陣的形式呈現出來，如下圖所示。



`bool OK(struct sGame game)`

這個函數的任務，是檢查「現在正在填的位置」的數字，與方陣中其他的數字有沒有重複。如果都沒有重複，就回傳 true，否則回傳 false。

`struct sGame TryTrySee(struct sGame game)`

這個函數的任務，是為「現在正在填的位置」找到一個合格的數值，也就是要為 game.board[game.x][game.y]找到合格的數值。方法是：

1. 將這格的數值增加 1。注意：直接增加 1，不要先設定 0 或其他初

始值。該設定的初始值，在 Initialize()中都已經做過了。

2. 呼叫 **OK()**來檢查這個值是否合格。如果合格就結束函數，回傳 game。
3. 如果 **OK()**認為不合格，就檢查這格的數值是否已經等於 DIMENSION 的平方。若已經等於，則結束函數，回傳 game。
4. 否則回到步驟 1。

回去對照前面所提供的 main()函數，就會發現程式在執行的演算法如下：

1. 初始化。設定為由(0,0)開始。
2. 為「現在正在填的位置」找到一個合格的數值。
3. 移到下一格。
4. 如果尚未完成就回到步驟 2。
5. 輸出。

程式的輸出應該如上面的圖所示。如果將 DIMENSION 的設定值改為 4 或 6，就會輸出 4x4 或 6x6 的結果如下圖。

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

1	7	13	19	25	31
2	8	14	20	26	32
3	9	15	21	27	33
4	10	16	22	28	34
5	11	17	23	29	35
6	12	18	24	30	36