[ICPC2009 国内予選問題 B] 島はいくつある？

- 地図として与えられるデータは図 B-1 であるが、「地図の海域は海で囲まれており、その外側へ歩いて出ることはできない」ので、図 B-1 を囲む周辺を海とした図 B-2 の地図の形で扱うと処理が少し楽になる。

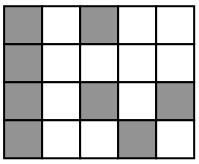

図 B-1 データとして与えられる地図

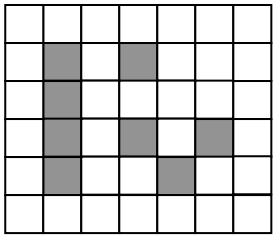

図 B-2 プログラム内部で扱う地図

- 地図のデータは、2 次元配列で表現できる。例えば c[i][j] に入力データセットの $c_{i,j}$ を格納する（c[i][j] = 0 ⇒ 海、c[i][j] = 1 ⇒ 陸）。入力データセットで与えられた地図の周辺に対応する部分の値は 0（海）とする （c[0][j] = c[h+1][j] = 0 (0 ≦ j ≦ w+1), c[i][0] = c[i][w+1] = 0 (0 ≦ i ≦ h+1)）。島番号として、2, 3, 4 のように 2 以上の整数を用いてみる。
- 地図上の正方形領域を組織的にスキャン（例えば、左上→右上→1 行下の左端→その行の右端→・・・→右下の順）し、島番号の付いていない陸(c[i][j]=1)が見つかれば、その陸から深さ優先探索(DFS)で隣接する陸に新たな島番号を付ける。最後に付けられた島番号の値-1 が、求める島の個数となる。
- 深さ優先探索は、再帰的な関数で容易に実現できる（プログラム例の関数 label 参照）。

プログラム例

```
/*  ACM-ICPC2009 国内予選 Problem B */
// http://www.waseda.jp/assoc-icpc2009/preliminary/contest/all_ja.html#section_B
// filename = pb.c
// コンパイル： cc -O2 pb.c
// 実行方法：   ./a.out < B0 > B0.result 等
// 確認方法：   diff B0.ans B0.result 等
// アルゴリズム：       マップをスキャンし、島番号が付けられていない陸地があれば、
//              その場所から深さ優先探索で隣接する陸地に新しい島番号をつける。
//              最後に付けた島番号から島の個数が分かる。
#include <stdio.h>
#define MAXW 50          // 地図の幅の最大値
#define MAXH 50          // 地図の高さの最大値

int w;                   // 地図の幅 1 <= w <= 50
int h;                   // 地図の高さ 1 <= h <= 50
int map[MAXH+2][MAXW+2];   // 地図データ 周囲を海で囲むため幅、高さともに２増やす
                           // 値 0 => 海、値 1 => 島番号無しの陸地
```

```
                              // 値 2 以上 => 島番号付き陸地
int num;                      // 島番号(最初の島番号は２)
                              // 最終的に num - 1 が求める島の個数となる
void label(int row, int column) // 深さ優先探索(DFS)で隣接陸地に島番号を付ける
{
  int i,j;
  map[row][column] = num;         // 現在の場所に島番号をふる
  // 島番号の付いていない隣接陸地に対して、その場所から再帰的に島番号をふる
  for(i=-1; i<=1; i++)            // 上下方向
    for(j=-1; j<=1; j++)          // 左右方向
      // そこが島番号無しの陸地なら、その位置で label を再起的に呼び出す
      if(map[row+i][column+j] == 1) label(row+i, column+j);
}
int main()
{
  int i,j;
  while(1){
    scanf("%d %d", &w, &h);       // 地図の幅(w)と地図の固さ(h)の入力
    if(w==0 && h==0) break;       // w=0, h=0 なら終了
    for(i=1; i<=h; i++)           // 地図データの読み込み
      for(j=1; j<=w; j++)
        scanf("%d", &map[i][j]);
    for(j=0; j<=w+1; j++)         // 地図の周辺は海
      map[0][j] = map[h+1][j] = 0;      // 最上行と最下行を 0(海)とする
    for(i=1; i<=h; i++)
      map[i][0] = map[i][w+1] = 0;      // 最左列と最右列を 0(海)とする
    num = 1;                      // 島番号の初期化
    for(i=1; i<=h; i++)           // 地図のスキャン
      for(j=1; j<=w; j++)
        if(map[i][j] == 1){       // 現在スキャンしている位置が島番号無しの陸地なら
          num++;                  // これは新しい島なので、島番号を増やして、
          label(i,j);             // その場所から DFS で隣接陸地に新島番号を付ける
        }
    printf("%d¥n", num-1);        // 島の個数は島番号-1
  }
}
```