

TOI推廣計畫

解題-梗圖著色



Icon made by [<https://www.freepik.com/>] from www.flaticon.com

題目

「選擇阿低，成就第一！」，由知名Youtuber <阿低英文> 所設立的阿低全科補習班，將於4月1日正式開班授課啦！在學科方面擁有強大的師資陣容，包含：李科太太生物課、九媚數學課、菜阿嘎台語課.....，除了學科外還特別邀請到梗圖插畫家當墾來上電腦繪圖美術課，讓壓力大的學生們感受藝術的陶冶。

當墾電腦繪圖美術課正式開課前，還提供知名梗圖著色體驗課程。只包括黑白兩色的上色體驗，給定著色範圍及圖案，黑色代表數字1、白色代表數字0，我們的任務是將每列的兩個黑色(數字1)間的未上色的部分(數字0)塗滿黑色。首先由左至右檢查，在搜尋到的第一個黑色與第二個黑色間著色，則算部分一上色完成，接下來繼續塗未著色第三、四個黑色間.....以此類推，一張圖有可能有多個部分待著色。

上完體驗課後對正式課程有興趣的同學們，可以手刀報名阿低全科補習班！名額有限，要搶要快！

輸入格式

第一行有兩個正整數 m 、 n ($1 \leq m \leq n \leq 100$)，代表圖的長與寬。接下來 m 行，每列 n 個數字表示這張圖未上色的樣子。所有的圖案只由1 (黑色)、0 (白色)兩個數字組成，彼此間以空格間隔。

輸出格式

輸出經過著色後的圖案。



將每列兩個黑色(數字1)邊框間未上色的部分塗滿黑色。
由左至右檢查，在搜尋到的第一個黑色與第二個黑色間著色，則算部分一上色完成，接著繼續塗未著色第三、四個黑色間.....以此類推，一張圖有可能有多個部分待著色。

輸入範例

```
4 7
0 1 1 0 0 0 0
0 0 1 0 0 0 1
1 0 1 0 0 1 1
0 0 0 0 0 0 0
```

輸出範例

```
0 1 1 0 0 0 0
0 0 1 1 1 1 1
1 1 1 0 0 1 1
0 0 0 0 0 0 0
```



解題重點:

1. 圖案讀取

陣列讀取、寫入操作

2. 著色範圍

搜尋判斷上色區域

Icon made by [<https://www.flaticon.com/authors/smashicons>] from www.flaticon.com

1. 圖案讀取



Icon made by [<https://www.flaticon.com/authors/eucalyp>] from www.flaticon.com

◆ 1. 圖案讀取

➤ 陣列操作：

(1) 創立陣列 (圖片 m 行 n 列)

[每次處理一行，一行讀 n 次] → 共讀 m 次

輸入範例 $m \times n = 4 \times 7$

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| m | 0 | 1 | 1 | 0 | 0 | 0 | 0 | ← |
| | 0 | 0 | 1 | 0 | 0 | 0 | 1 | ← |
| | 1 | 0 | 1 | 0 | 0 | 1 | 1 | ← |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ← |
| n | | | | | | | | |

✓ 範例程式：

array[105] 紀錄圖案

```
int array[105] = {};  
for( int i = 0; i < m; ++i )  
{  
    for( int j = 0; j < n; ++j )  
    {  
        scanf("%d", &array[j]);  
    }  
}
```

◆ 1. 圖案讀取

➤ 陣列操作：black 紀錄每行黑色數量

(2) 記錄每行黑色數量

若小於2 → 則保持原狀輸出

輸入範例 $m \times n = 4 \times 7$

black = 2



| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

✓ 範例程式：

```
int array[105] = {};  
int black = 0;  
  
for( int i = 0; i < m; ++i )  
{  
    for( int j = 0; j < n; ++j )  
    {  
        scanf("%d", &array[j]);  
  
        if(array[j] == 1)  
            black += 1;  
    }  
    if(black < 2)  
    {  
        for( int j = 0; j < n; ++j )  
            printf("%d", array[j]);  
    }  
}
```

2. 著色範圍



Icon made by [<https://www.freepik.com/>] from www.flaticon.com

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start : 著色起點

end : 著色終點

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

current

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

↑
end

↑
start

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start：著色起點

end：著色終點

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

current

第一部分上色

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

↑
start

↑
end

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start : 著色起點

end : 著色終點

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |



再找下個部分

current



| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|



end



start

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start : 著色起點

end : 著色終點

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

再找下個部分

current

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

↑
end

↑
start

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start : 著色起點

end : 著色終點

找到下個部分著色終點

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

current

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

↑
start

↑
end

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start：著色起點

end：著色終點

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |



第二部分上色

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|



start



end

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start：著色起點

end：著色終點

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |



第二部分上色

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

↑
start

↑
end

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start : 著色起點

end : 著色終點

再找下一部份

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

↑ ↑
start end

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |



current



◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start：著色起點

end：著色終點

跑到第n個，未找到著色終點1

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |



current



| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|



end



start

◆ 2. 著色範圍

➤ 判斷著色：

(1) 設定每行著色起、始點

找到起始點 → 再找下個部分

start：著色起點

end：著色終點

共著色

輸入範例 $m \times n = 4 \times 7$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |



| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

◆ 2. 著色範圍

(1) 設定每部分著色起、始點

```
int array[105] = {};  
int colored[105] = {};  
  
for( int i = 0; i < m; ++i )  
{  
    int start = -1, end = -1;  
    for( int j = 0; j < n; ++j )  
    {  
        scanf("%d", &array[j]);  
        if(array[j] == 1)  
        {  
            [redacted]  
        }  
    }  
}
```

int colored[105] 紀錄著色後圖案

start : 著色起點
end : 著色終點

初始化 = -1

current

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|

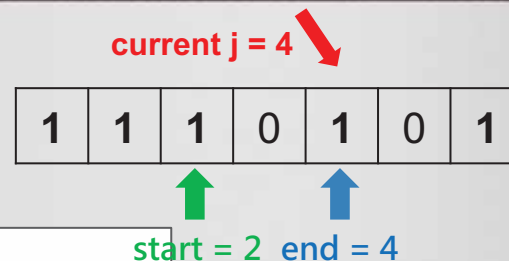
↑ end
↑ start

◆ 2. 著色範圍

(2) 搜尋+紀錄每部分著色起始點黑色頭、尾

```
int array[105] = {};  
int colored[105] = {};  
for( int i = 0; i < m; ++i )  
{  
    int start = -1, end = -1;  
    for( int j = 0; j < n; ++j )  
    {  
        scanf("%d", &array[j]);  
        if(array[j] == 1)  
        {  
            [redacted]  
        }  
    }  
}
```

```
int start = -1, end = -1;  
for( int j = 0; j < n; ++j )  
{  
    scanf("%d", &array[j]);  
    if(array[j] == 1)  
    {  
        if(start == -1)  
            start = j;  
        else if(start != -1)  
        {  
            end = j;  
            for( int k = start; k <= end; ++k )  
            {  
                colored[k] = 1;  
            }  
            start = -1; end = -1;  
        }  
    }  
}
```



遇到左邊黑色1頭

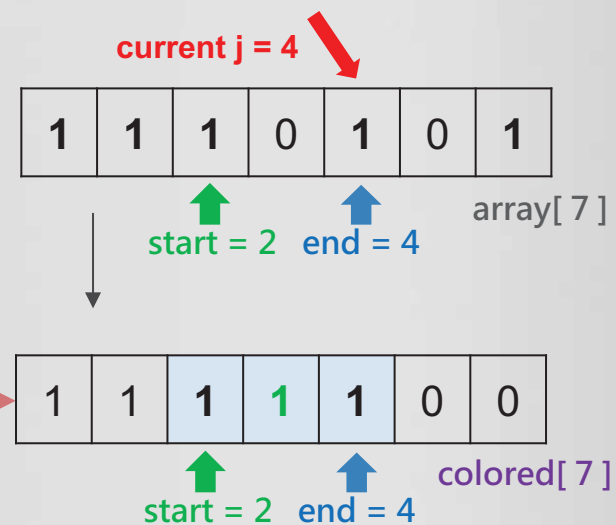
遇到右邊黑色1尾

紀錄
著色起始點位置
於 start / end

◆ 2. 著色範圍

(3) 在黑色頭尾間，使用 **colored** 陣列 著色

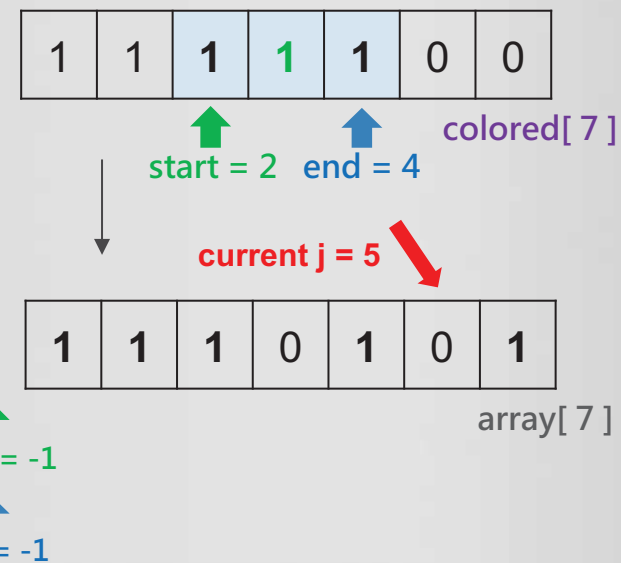
```
int start = -1, end = -1;
for( int j = 0; j < n; ++j )
{
    scanf("%d", &array[j]);
    if(array[j] == 1)
    {
        if(start == -1)
            start = j;
        else if(start != -1)
        {
            end = j;
            for( int k = start; k <= end; ++k )
            {
                colored[k] = 1;
            }
            start = -1; end = -1;
        }
    }
}
```



◆ 2. 著色範圍


(4) 一部份著色完成，再找下區著色 → 初始化起始點

```
int start = -1, end = -1;
for( int j = 0; j < n; ++j )
{
    scanf("%d", &array[j]);
    if(array[j] == 1)
    {
        if(start == -1)
            start = j;
        else if(start != -1)
        {
            end = j;
            for( int k = start; k <= end; ++k )
            {
                colored[k] = 1;
            }
            start = -1; end = -1;
        }
    }
}
```



◆ 3. 範例程式

```
#include<stdio.h>

int main()
{
    int m = 0, n = 0;
    scanf("%d %d", &m, &n);
    for( int i = 0; i < m; ++i ){
        
    }
    return 0;
}
```

```
int array[105] = {};
int colored[105] = {};
int black = 0;

int start = -1, end = -1;
for( int j = 0; j < n; ++j ){
    scanf("%d", &array[j]);
    if(array[j] == 1){
        black += 1;
        colored[j] = 1;

        if(start == -1)
            start = j;
        else if(start != -1){
            end = j;
            for( int k = start+1; k < end; ++k ){
                colored[k] = 1;
            }
            start = -1; end = -1;
        }
    }
}

for( int j = 0; j < n; ++j ){
    if(black < 2)
        printf("%d ", array[j]);
    else if(black >= 2)
        printf("%d ", colored[j]);
}
printf("\n");
```