

## 先解釋程式

```
typedef struct
{
    BYTE b;
    BYTE g;
    BYTE r;
} RGB;
```

typedef：定義一個已知資料型態的別名

struct：自訂結構的語法

所以這邊的意思就是，定義了一個別名是RGB的結構，這個結構裡面有三個BYTE變數

```
BITMAPFILEHEADER fileHeader;
BITMAPINFOHEADER infoHeader;
```

宣告BMP檔案標頭以及資訊標頭變數

```
FILE *pfin;
FILE *pfout;
```

宣告兩個FILE指標變數

變數是指標，因為FILE是指向檔案  
現在使用到哪裡

```
errno_t err;
```

宣告一個errno\_t變數

errno\_t是一種數據類型，代表錯誤號碼

```
err = fopen_s(&pfin, "c:\\wpfang\\a.bmp", "rb");
err = fopen_s(&pfout, "c:\\wpfang\\b.bmp", "wb");
```

Fopen\_s就是打開文件，打開成功返回0，失敗返回非0  
第一個參數應該要是一個指向檔案指標的指標，但我們前面宣告FILE的時候只是指標而已，所以用&取址  
後面的rb代表打開一個二進制文件，文件必須存在，唯讀  
Wb代表打開或新建一個二進制文件，唯寫

```
fread(&fileHeader, sizeof(BITMAPFILEHEADER), 1, pfin);
fread(&infoHeader, sizeof(BITMAPINFOHEADER), 1, pfin);
```

Fread用來讀取資料，第一個參數應該要是指向一塊記憶體  
的指標，但前面宣告BMP檔案標頭和資訊標頭的時候只是  
變數，所以用&取址  
指向的這塊記憶體至少要有第二個參數\*第三個參數的大小  
最後一個參數是指向FILE物件的指標

注意，要先讀取檔案標頭再讀取資訊標頭，因為pfin這個指標  
會隨著檔案的讀取而改變指向的位址，順序錯了會導致讀取錯誤

```
const int height = 512, width = 512;
```

宣告兩個常數變數，值都是512

```
if (infoHeader.biBitCount >= 1)
```

biBitCount代表一個像素點佔幾位元

```
int size = height * width;
```

宣告一個int變數，其值為height\*width  
也就是512\*512

```
RGB img[height][width];  
RGB img2[width][height];
```

宣告兩個RGB結構的二維陣列  
大小是height\*width和width\*height

```
fread(img, sizeof(RGB), size, pfin);
```

Fread讀取圖片  
這邊第一個參數沒有像前面一樣加上&是因為  
陣列變數其實就是一個指標

```
int i, j;  
for (i = 0; i < height; i++)  
    for (j = 0; j < width; j++)  
    {  
        (img2[i][j]).b = (img[width - 1 - j][i]).b;  
        (img2[i][j]).r = (img[width - 1 - j][i]).r;  
        (img2[i][j]).g = (img[width - 1 - j][i]).g;  
    }
```

用兩個迴圈遍歷圖片的所有像素  
將img[width-1-j][i]的值賦予img2[i][j]

0,0	0,1	0,2	0,3	0,4
1,0				
2,0				
3,0				
4,0				

img

0,0	0,1	0,2	0,3	0,4
1,0				
2,0				
3,0				
4,0				

img2

所以img2儲存了向右轉90度的img

```
fwrite(&fileHeader, sizeof(fileHeader), 1, pfout);  
fwrite(&infoHeader, sizeof(infoHeader), 1, pfout);  
fwrite(img2, sizeof(RGB), size, pfout);
```

用fwrite將修改後的圖片寫入文件，一樣順序不能錯  
先是檔案標頭再來資訊標頭，最後才是像素

```
err = fclose(pfin);  
err = fclose(pfout);
```

最後將兩個檔案關掉

程式部分解釋完就可以弄流程圖了，看下一頁

# 流程图

