第6章
影像處理的基礎知識

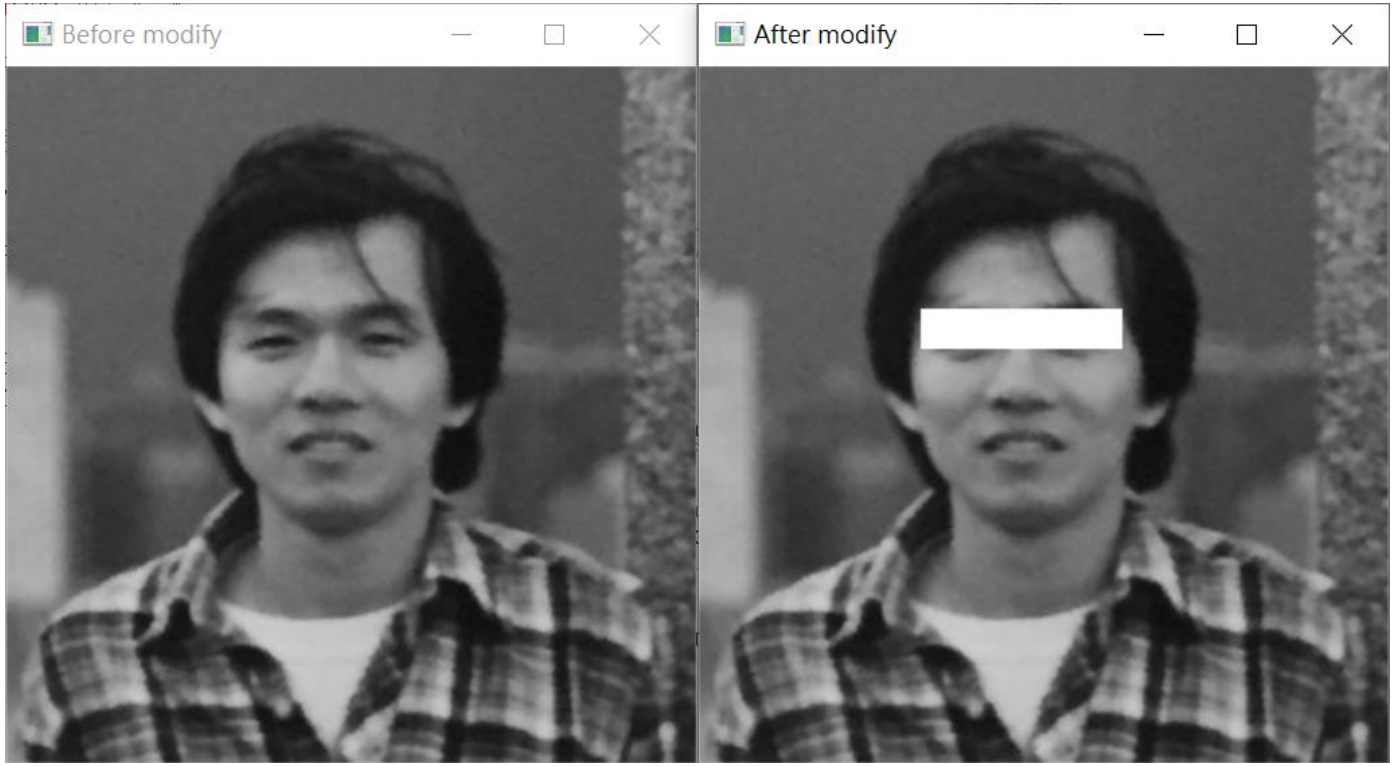
6-1：灰階影像的編輯

- **6-1-1：自創灰階影像與編輯的基礎實例**
- 程式實例ch6_1.py：自創5 x 12的灰階影像陣列，列印此灰階影像陣列，然後讀取(1, 3)座標的像素，列出所讀的值。修改(1, 3)座標的像素，最後列出灰階影像陣列與修改結果。

```
========== RESTART: D:/OpenCV_Python/ch6/ch6_1.py ==========
修改前 image=
[[0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]]
image[1,4] = 0
修改後 image=
[[  0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0 255   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0]]
image[1,4] = 255
```

6-1-2：讀取灰階影像與編輯的實例

- 程式實例ch6_2.py：讀取灰階影像，然後用白色長條遮住眼睛部位，分別顯示原始影像與修改後的影像。



6-2：彩色影像的編輯

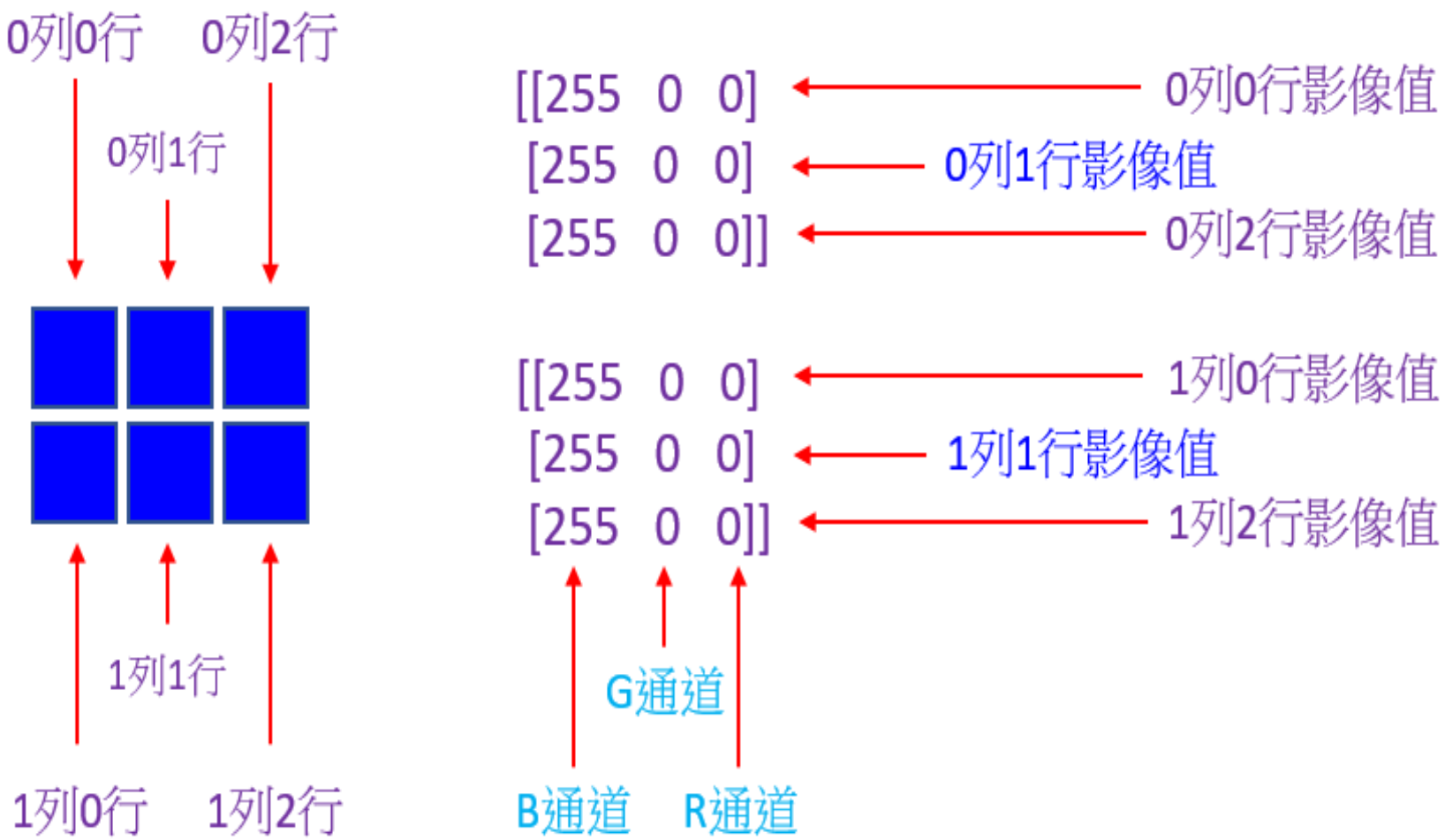
- **6-2-1：了解彩色影像陣列的結構**
- 程式實例ch6_3.py：建立3組2 x 3的彩色影像，第一組彩色影像陣列是藍色，第二組彩色影像陣列是綠色，第三組彩色影像陣列是紅色，列出陣列內容。

```
==================== RESTART: D:\OpenCV_Python\ch6\ch6_3.py ====================
blue image =
[[[255   0   0]
  [255   0   0]
  [255   0   0]]

 [[255   0   0]
  [255   0   0]
  [255   0   0]]]
green image =
[[[  0 255   0]
  [  0 255   0]
  [  0 255   0]]

 [[  0 255   0]
  [  0 255   0]
  [  0 255   0]]]
red image =
[[[  0   0 255]
  [  0   0 255]
  [  0   0 255]]

 [[  0   0 255]
  [  0   0 255]
  [  0   0 255]]]
```

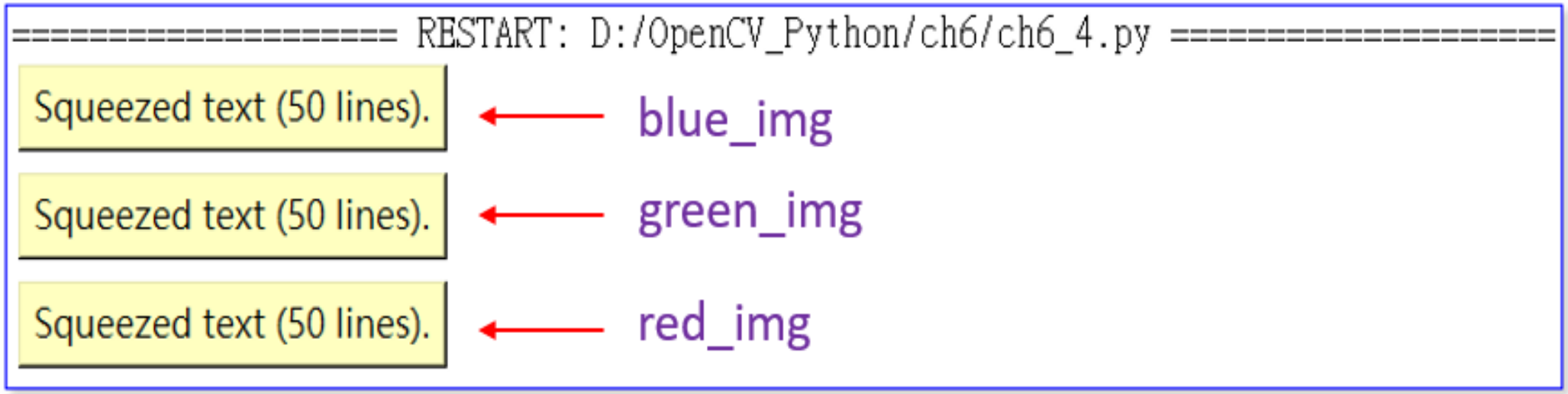
第0列影像元素

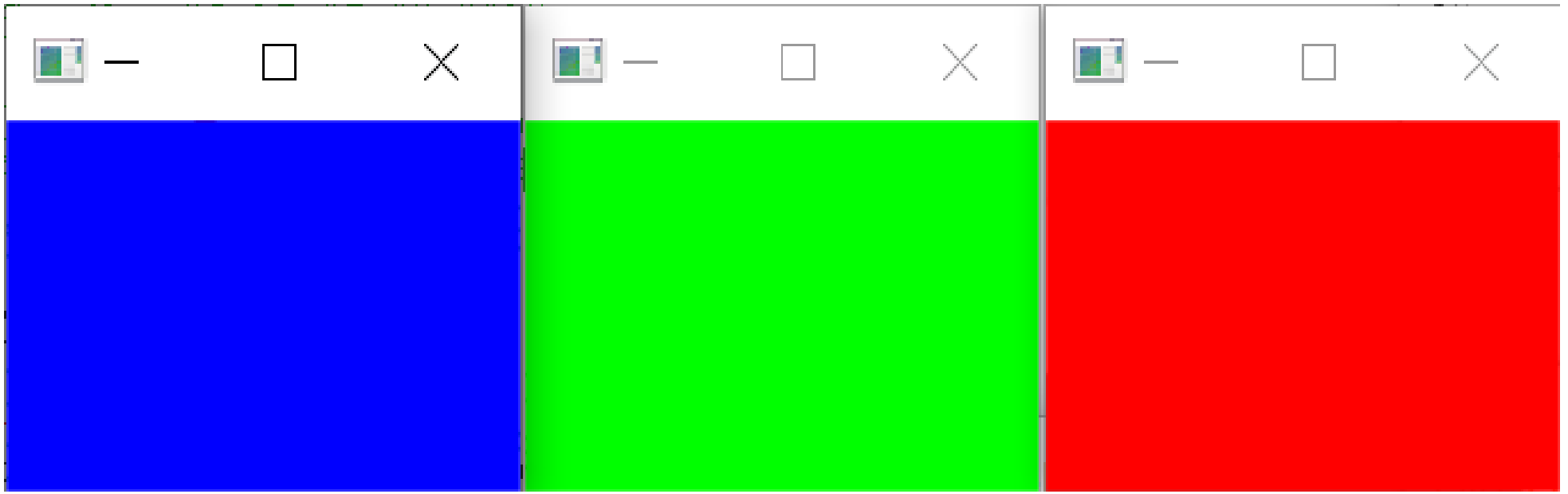
這是2 x 3 x 3的影像陣列

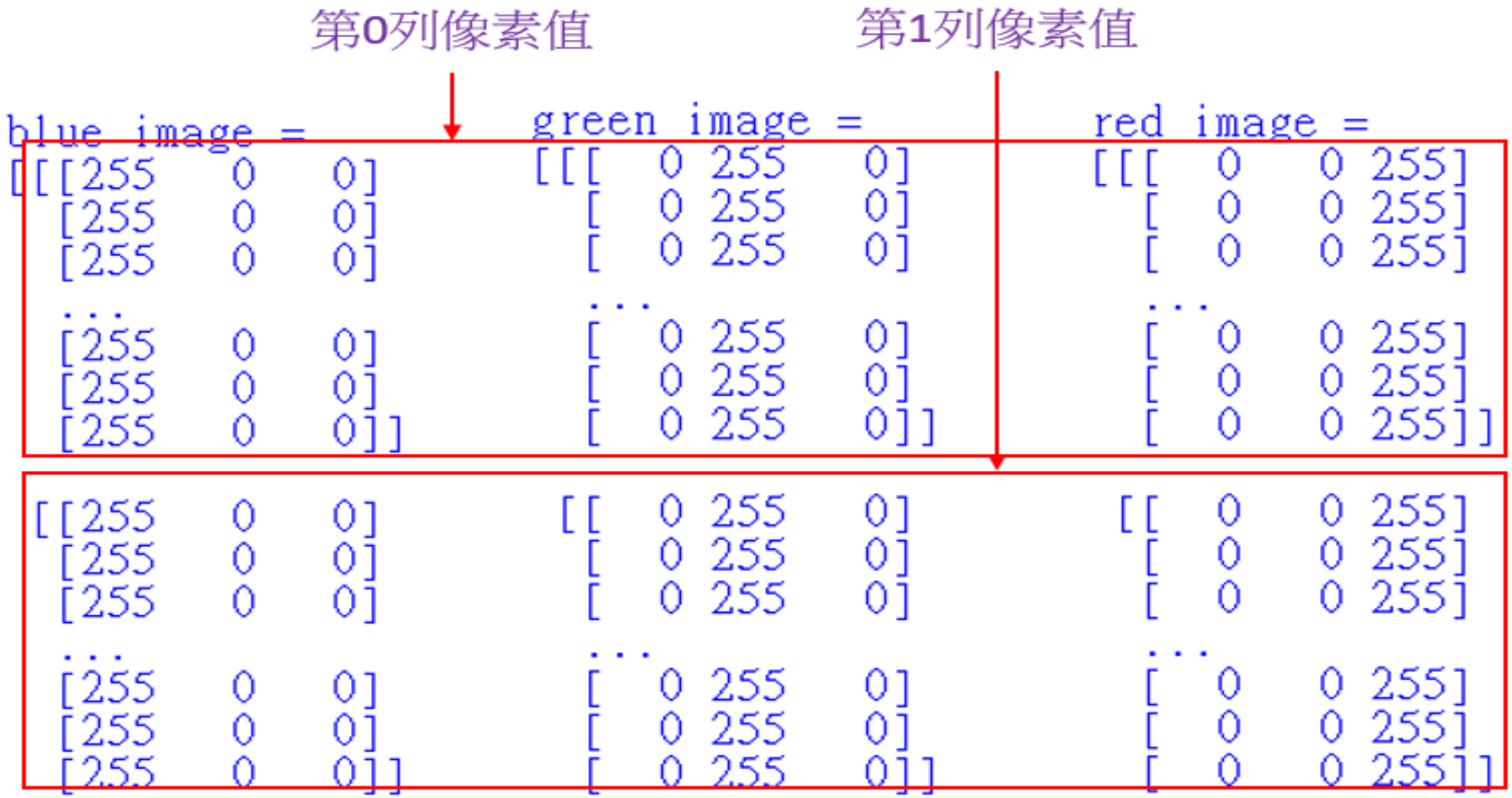
第1列影像元素


0列0行
0列2行
0列1行
1列1行
1列0行
1列2行
[[255 0 0]
[255 0 0]
[255 0 0]]
0列0行影像值
0列1行影像值
0列2行影像值
[[255 0 0]
[255 0 0]
[255 0 0]]
1列0行影像值
1列1行影像值
1列2行影像值
G通道
B通道
R通道

- 程式實例ch6_4.py：建立藍色、綠色、紅色的視窗，然後解釋彩色陣列內容的意義。






第0列像素值
第1列像素值
blue image =
[[[255 0 0]
[255 0 0]
[255 0 0]
...
[255 0 0]
[255 0 0]
[255 0 0]]
green image =
[[[0 255 0]
[0 255 0]
[0 255 0]
...
[0 255 0]
[0 255 0]
[0 255 0]]
red image =
[[[0 0 255]
[0 0 255]
[0 0 255]
...
[0 0 255]
[0 0 255]
[0 0 255]]
[[255 0 0]
[255 0 0]
[255 0 0]
...
[255 0 0]
[255 0 0]
[255 0 0]]
[[0 255 0]
[0 255 0]
[0 255 0]
...
[0 255 0]
[0 255 0]
[0 255 0]]
[[0 0 255]
[0 0 255]
[0 0 255]
...
[0 0 255]
[0 0 255]
[0 0 255]]

- **6-2-2：自創彩色影像與編輯的實例**
- 程式實例ch6_5.py：自創一個2 x 3 x 3的彩色影像陣列，先列印此彩色影像陣列。然後列印[0,1]像素點的BGR內容。接著第12列是修訂[0,1]的內容為[50,100,150]，最後再列印一次此影像陣列，驗證修改結果。

```
========== RESTART: D:/OpenCV_Python/ch6/ch6_5.py ==========
blue =
[[[255   0   0]
  [255   0   0]
  [255   0   0]

 [[255   0   0]
  [255   0   0]
  [255   0   0]]]
blue[0,1] = [255   0   0]
修訂後
blue =
[[[255   0   0]
  [ 50 100 150]
  [255   0   0]

 [[255   0   0]
  [255   0   0]
  [255   0   0]]]
```

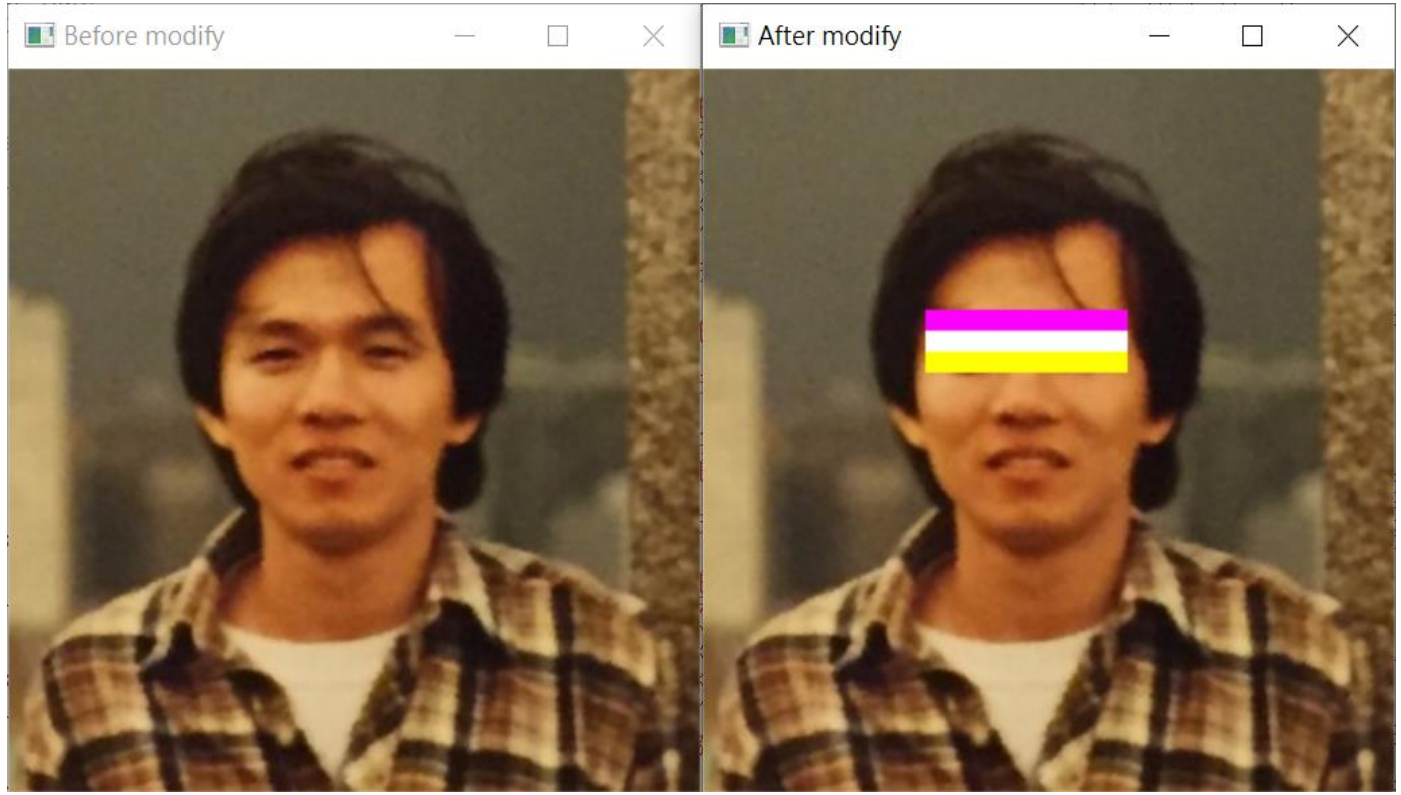
- 程式實例ch6_6.py：自創一個2 x 3 x 3的彩色影像陣列，先列印此彩色影像陣列。然後列印[0,1,2]像素點的通道值。接著第12列是修訂[0,1,2]的內容為50，最後再列印一次此影像陣列，驗證修改結果。

```
========== RESTART: D:\OpenCV_Python\ch6\ch6_6.py ==========
blue =
[[[255   0   0]
  [255   0   0]
  [255   0   0]]

 [[255   0   0]
  [255   0   0]
  [255   0   0]]]
blue[0,1,2] = 0
修訂後
blue =
[[[255   0   0]
  [255   0  50]
  [255   0   0]]

 [[255   0   0]
  [255   0   0]
  [255   0   0]]]
blue[0,1,2] = 50
```

6-2-3：讀取彩色影像與編輯的實例

- 程式實例ch6_7.py：讀取彩色影像，然後編輯影像，在編輯過程會列出長條左上角修改前與修改後的像素值。

```
========== RESTART: D:/OpenCV_Python/ch6/ch6_7.py ==========
修改前img[115,110] = [21 26 57]
修改前img[125,110] = [ 31  63 134]
修改前img[135,110] = [ 57 126 195]
修改後img[115,110] = [255   0 255]
修改後img[125,110] = [255 255 255]
修改後img[135,110] = [  0 255 255]
```


Before modify
After modify

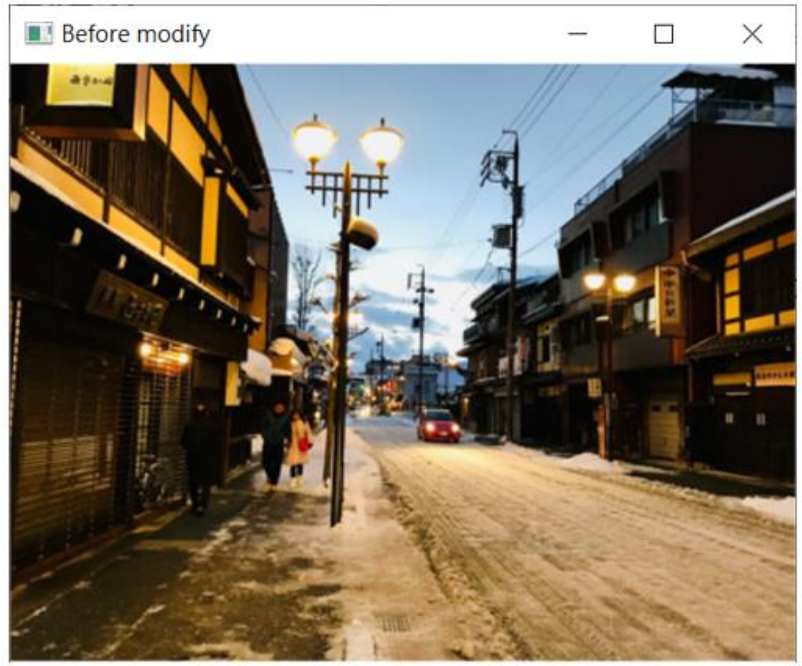
- 例如：10 ~ 12列可以用下列取代。
- img[115:210,110:210] = [255, 0, 255]
- 程式實例ch6_7_1.py：重新設計ch6_7.py，第10 ~ 12列使用單列代替迴圈。

6-3：編輯含alpha通道的彩色影像

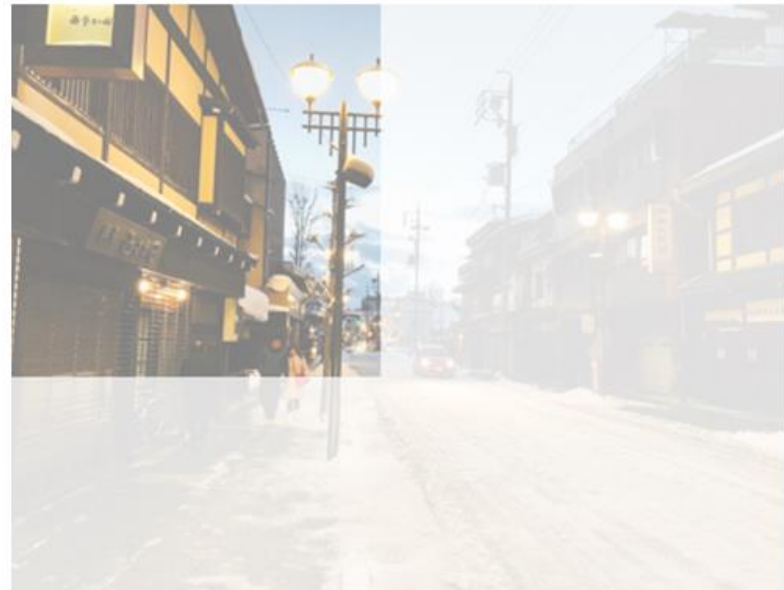
- 索引3是A(alpha)通道如下所示：
- [B, G, R, A]

- 程式實例ch6_8.py：在ch6資料夾有street.png檔案，這個檔案的透明度是32，這個程式會讀取street.png，同時顯示[10,50]和[50,99]的像素值，然後修改[0,0]至[200,200]間的alpha值為半透明的128，最後再列出[10,50]和[50,99]的像素值，讀者可以比較修改結果。同時將修改結果存入street128.png。

```
========== RESTART: D:/OpenCV_Python/ch6/ch6_8.py ==========
修改前img[10,50] = [ 27  82 109  32]
修改前img[50,99] = [ 19 168 248  32]
------------------------------------------------------------
修改後img[10,50] = [ 27  82 109 128]
修改後img[50,99] = [ 19 168 248 128]
```



street.png



street128.png

- 例如：上述程式第9列到第11列，可以使用下列含切片的程式碼取代。
- img[0:200,0:200,3] = 128
- 細節可以參考下列實例 ch6_8_1.py，執行結果則存入 street128_1.png。

6-4：Numpy高效率讀取與設定像素的方法

- 6-4-1：灰階影像的應用
- 在灰階影像的應用中item( )與itemset( )語法如下：
- ndarray.item(列, 行)　　# 回傳列,行索引的值
- ndarray.itemset(索引, 值)　# 將值設定給指定索引 的 ndarray 變數
- 程式實例ch6_9.py：建立一個3 x 5的灰階影像陣列，列印此陣列內容，這個程式第6列使用item( )讀取像素點內容，第7列使用itemset( )修改索引(1,3)的值為255，然後第9列輸出此陣列，第10列輸出特定索引(1,3)的內容。

```
========== RESTART: D:/OpenCV_Python/ch6/ch6_9.py ==========
image =
[[178 116 113  76 179]
 [ 69 180  60 105 113]
 [198  91 181  65 174]]
修改前image.item(1,3) = 105
------------------------------------------------------------
修改後image =
[[178 116 113  76 179]
 [ 69 180  60 255 113]
 [198  91 181  65 174]]
修改後image.item(1,3) = 255
```

- 程式實例ch6_10.py：使用itemset( )函數重新設計ch6_2.py。

6-4-2：彩色影像的應用

- 在彩色影像的應用中item( )與itemset( )語法如下：

ndarray.item(列, 行, 通道)　　# 回傳列,行,通道索引的值

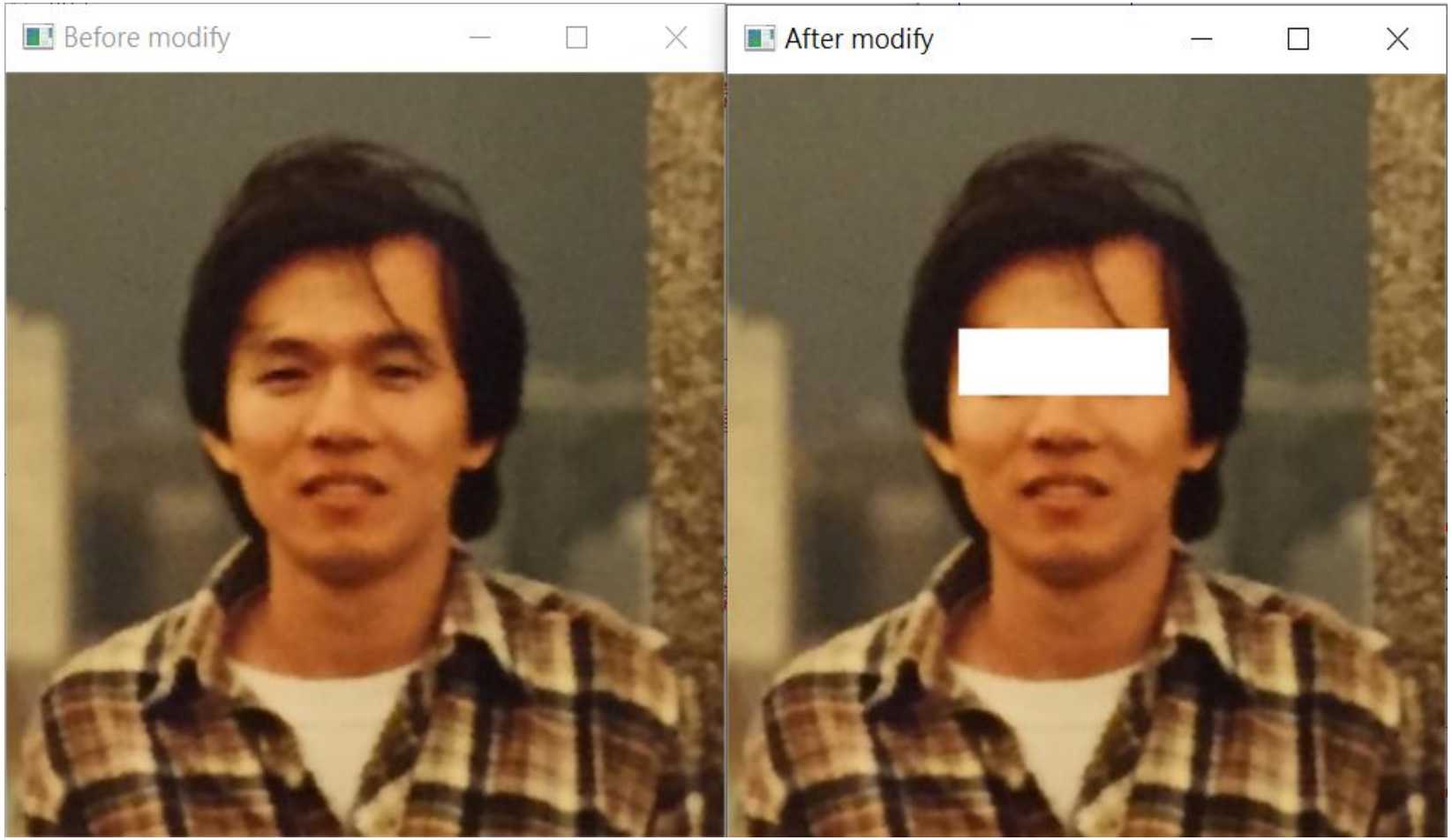
ndarray.itemset((列, 行, 通道), 值)　　# 將值設定給指定索引的ndarray變數

- 程式實例ch6_11.py：使用item( )和itemset( )函數重新設計ch6_6.py。

```
========== RESTART: D:/OpenCV_Python/ch6/ch6_11.py ==========
blue =
[[[255   0   0]
  [255   0   0]
  [255   0   0]]

 [[255   0   0]
  [255   0   0]
  [255   0   0]]]
blue[0,1,2] = 0
修訂後
blue =
[[[255   0   0]
  [255   0  50]
  [255   0   0]]

 [[255   0   0]
  [255   0   0]
  [255   0   0]]]
blue[0,1,2] = 50
```

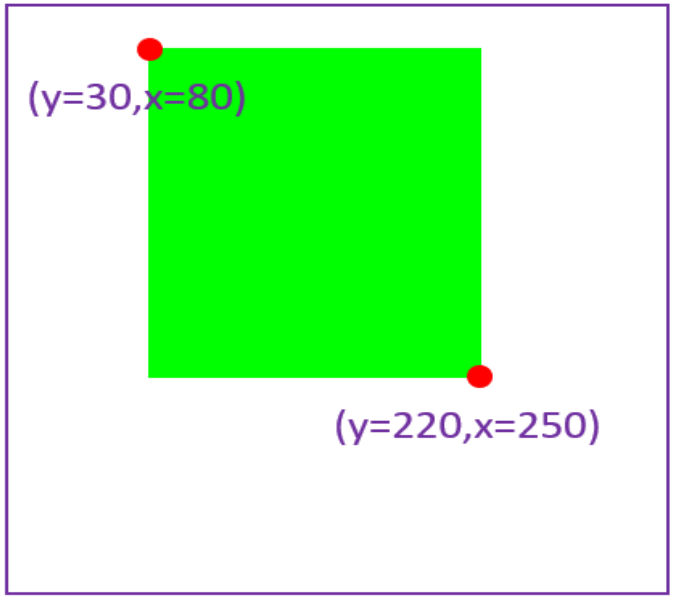
- 程式實例ch6_12.py：使用item( )和itemset( )函數重新設計修改ch6_7.py，讀取彩色影像與修訂，這個程式只用一個白色長條修訂部分影像。

```
========== RESTART: D:/OpenCV_Python/ch6/ch6_12.py ==========
修改前img[115,110,1] = 26
修改前img[125,110,1] = 63
修改前img[135,110,1] = 126
修改後img[115,110,1] = 255
修改後img[125,110,1] = 255
修改後img[135,110,1] = 255
```

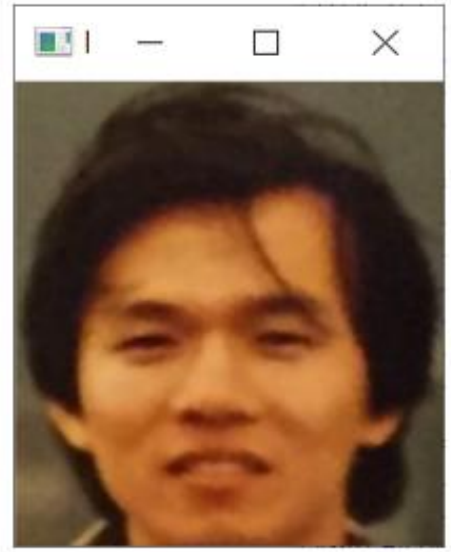

Before modify
After modify

6-5：影像感興趣區域的編輯

- **6-5-1：擷取影像感興趣區塊**
- 程式實例ch6_13.py：使用jk.jpg影像檔案，設計只取臉部，然後開啟視窗顯示臉部，同時存入jkface.jpg。這一個實例感興趣區域(ROI)的座標如下：

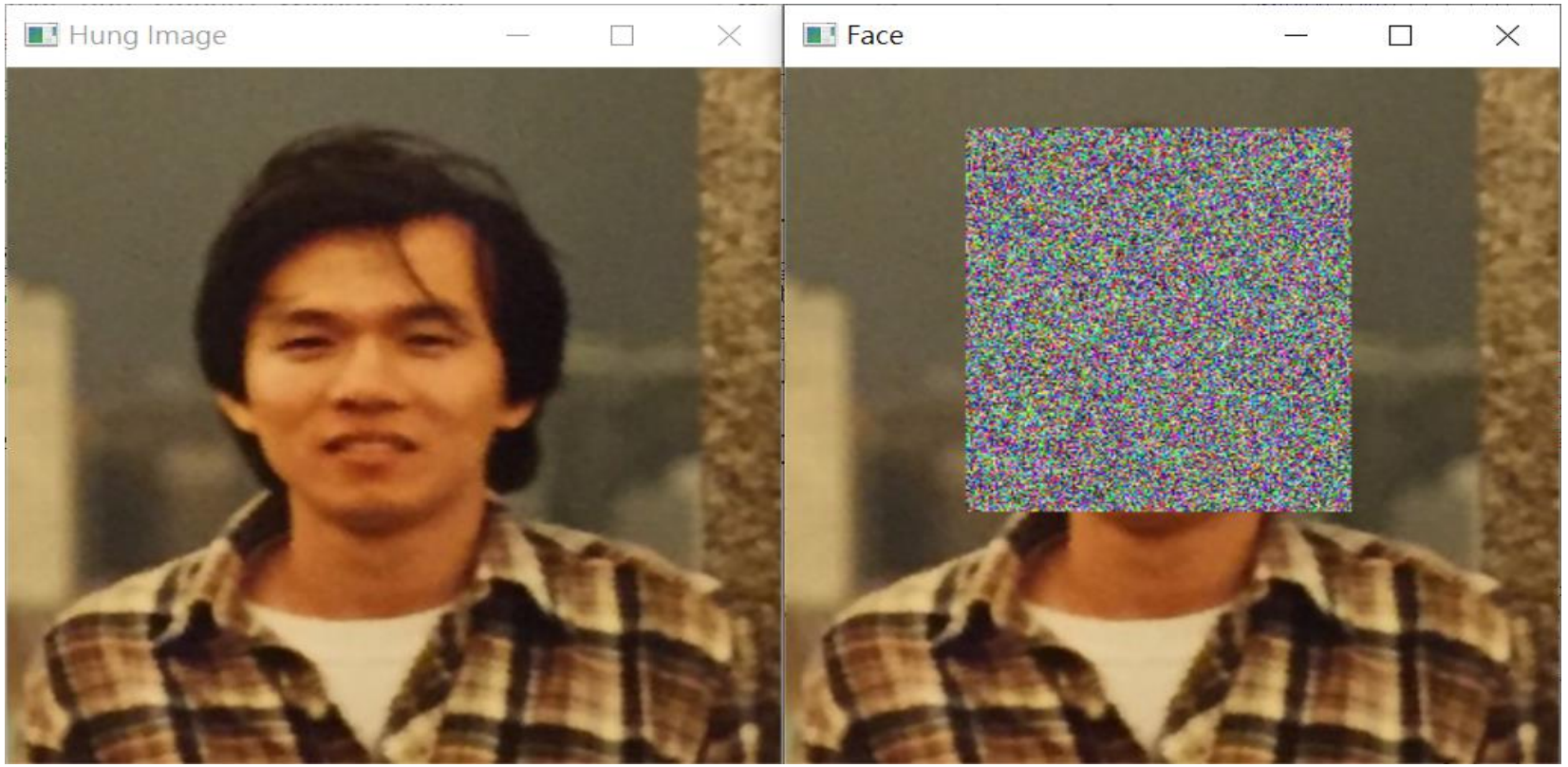



Hung Image


I

6-5-2：建立影像馬賽克效果

- 程式實例ch6_14.py：為感興趣區塊(ROI)設定馬賽克。



6-5-3：感興趣區塊在不同影像間移植

- 程式實例ch6_15.py：將感興趣的區塊在不同影像間複製，這個程式會將頭像複製到美鈔的影像上。