電機所控制組 110318051 張皓博

HW1\_handmade講解

我使用Visual Studio來寫這個C++作業，執行方式為把.exe檔跟InputImage.bmp放在同個資料夾底下後，執行.exe即可產生：

1. Rotated.bmp (順時針90度的圖片)
2. BlueChannel.bmp (G→0,R→0的RGB圖片)
3. GreenChannel.bmp (B→0,R→0的RGB圖片)
4. RedChannel.bmp (B→0,G→0的RGB圖片)
5. doubleSize.bmp (放大兩倍的圖片)
6. oneHalfSize.bmp (縮小1/2倍的圖片)

在一開始我先宣告了一個structure叫header，用來存放header的各種資訊。

我分別寫了6個function：

1. **void bmpRead():**

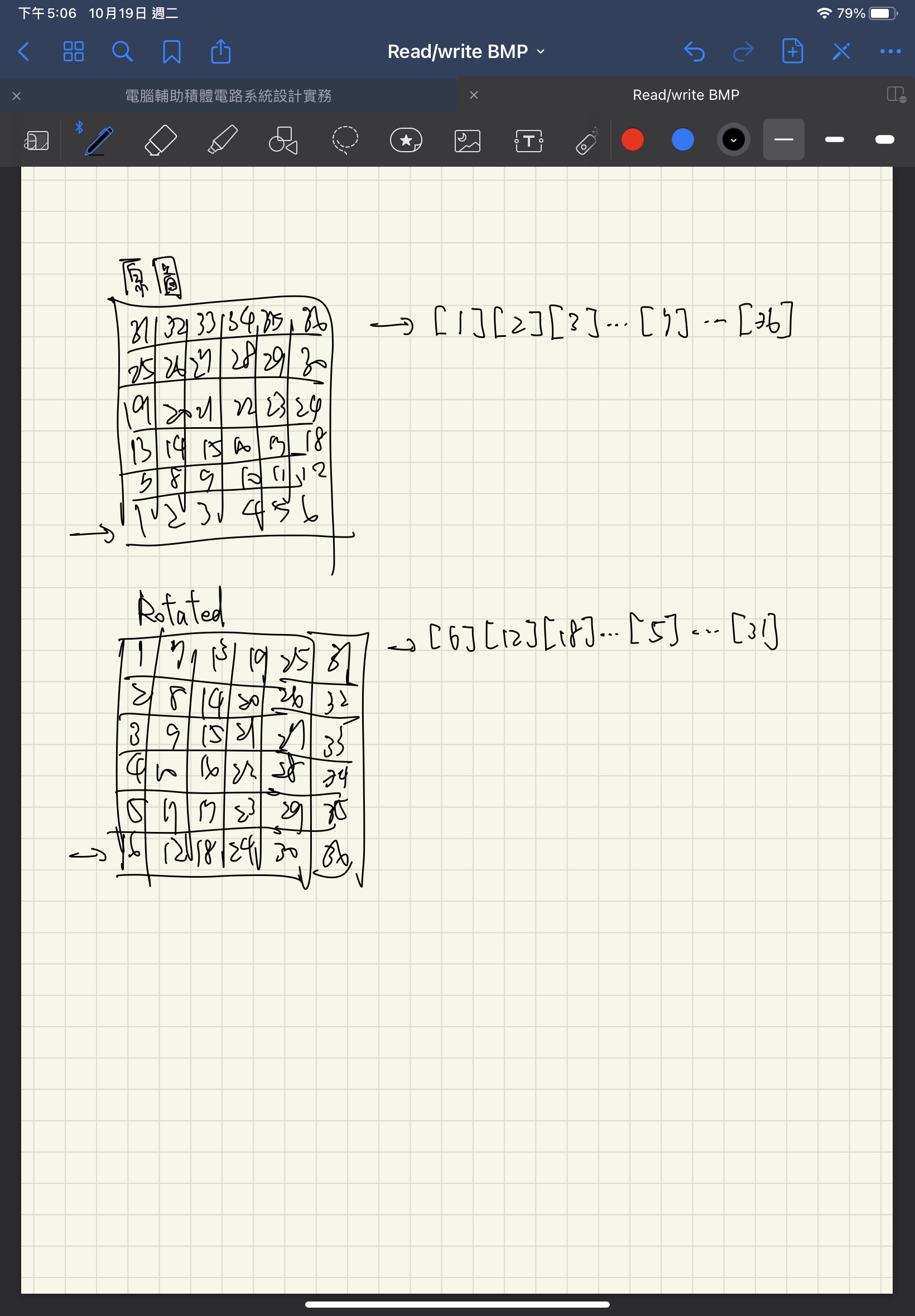
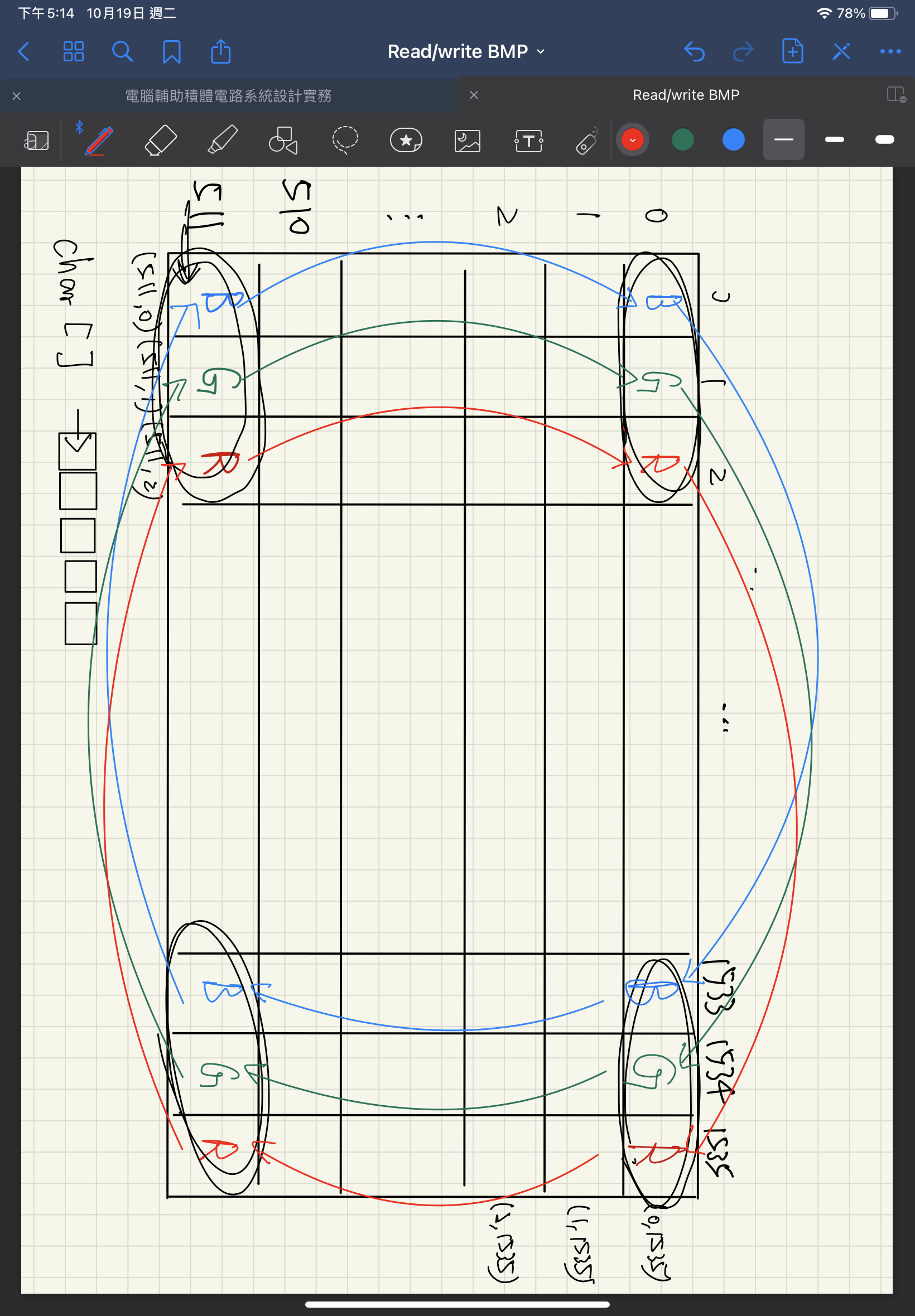
先利用fopen開啟InputImage，然後啟用讀取二進位的模式，利用fread函式根據header的每個info長度一個個分別存入名為header的structure中，最後將raw data存成一維陣列。

1. **void bmpWrite(char const\* pictureName, struct header header, unsigned char pixels\_Output [], unsigned int width, unsigned int height):**

可以傳入圖片名稱、圖片的header資訊、圖片的raw data資訊以及圖片的長寬，方式跟bmpRead是一樣的，不同的是從fread變成fwrite，將header資訊跟改造後的raw data依序寫回圖片。

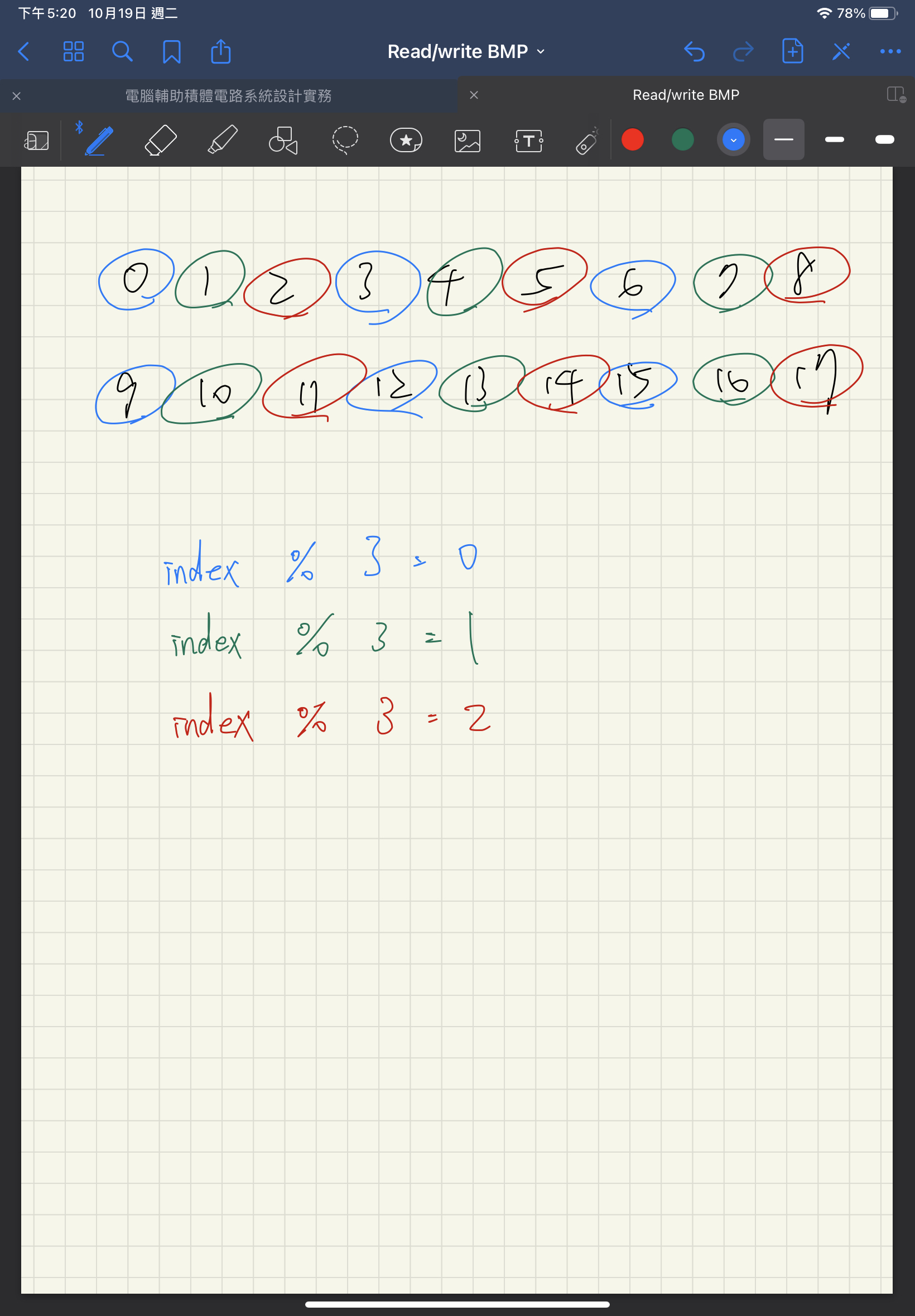
1. **void ImageRotation():**

思考邏輯如下圖，原圖raw data讀取時從左下到右上，我把讀取後的一維陣列轉換成二維陣列方便思考，將原圖的二維陣列順時針旋轉後得到Rotated的二維陣列，其中每一個pixels都包含著BGR三個值，所以在寫入時要同時把BGR依序寫進去，完成旋轉後再從左下到右上寫回一維陣列，才能正確寫入圖片。

1. **void channelSeparation():**

這裡的想法就是先宣告三個預設值都是0的一維陣列，然後blueChannel就只放入B的值，GreenChannel只放G的值，RedChannel只放R的值，造就了BlueImage(B,0,0)、GreenImage(0,G,0)、RedImage(0,0,R)。



1. **void double\_Size():**

bmp檔在調整圖片大小時，除了image data需要更動外，header內容也需要調整，所以先調整了bfSize、biWidth、biHeight，在image data的部分，將原本的圖片的每個點在height & width的部分都重複兩次，達到放大圖片的效果。

1. **void one\_half\_Size():**

跟double\_Size類似，先調整了header的bfSize、biWidth、biHeight，在image data的部分，將原本的圖片的height & width，每兩個點取一次到新的imagePixels，達到縮小圖片的效果。

最後在main()呼叫這些函數，以完成作業要求。

上一次寫C++大約是3年前的事情，被Python寵壞的我花了很長的時間重新熟悉type跟宣告，也花了很多時間弄懂bmp的概念，懂得bmp的存放方式之後，寫起程式就相對得心應手。