Problem D
高斯濾波器

Time limit: 1 second
Memory limit: 256 megabytes

題目內容

本題題本極長，請耐心看完?

在影像處理的領域裡，Image Filter，是一個時常被提出的概念，圖片在被對應的 Image Filter 處理後會有不同的型態出現。

眞實世界中，存在許多不同種類的濾波器，大致上可以分成以下兩種

- 平滑化 (濾除雜訊): 均值濾波器, 中值濾波器, Gaussian Filter
- 銳化 (強化邊緣): Laplace Filter, Sobel Filter

本題需實作一個簡化後的 Gaussian Filter

1. **Gaussian Filter 的用途**
   減少圖像雜訊以及降低細節層次，其視覺效果就像是經過一個半透明屏幕在觀察圖像。

經過Gaussian Filter後的圖像

   爲何 Gaussian Filter 能夠將圖形模糊化呢?

2. **Gaussian Filter 矩陣**
   以下近似 3*3 Gaussian Filter 的 generalized weighted smoothing filter 矩陣，圖像與 3*3 Gaussian Filter 做卷積將會達到濾除雜訊、低通、模糊化的效果。以下圖片爲 Gaussian Filter 矩陣的範例

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

generalized weighted smoothing filter

有可能會問，爲何對這個矩陣卷積就能將圖片模糊化?

- 卷積後會從九個點變一個點，當然會模糊化阿!! （卷積是什麼會在後面提到)

如果要將圖片模糊化，爲何不 9 個點相加除以 9 就好了?

- 不論是對任何圖像作低通，都不希望圖像失眞，而降低圖像失眞最好的辦法就是增加圖像中心點的權重，所以中心點的權重最高，越往邊角權重就越低!

3. **Gaussian Filter 數學函式**

實際的 Gaussian Filter 由此方程式產生

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

爲了簡化計算，假設 sigma = 0.707(根號 0.5)

$$G(x,y) = \frac{1}{\pi} e^{-x^2+y^2}$$

Gaussian Filter 的中心點 (x,y) 須爲 (0,0)，下例爲一個 3*3 的 (x,y) 值矩陣

$$\begin{bmatrix} (-1,-1) & (0,-1) & (1,-1) \\ (-1,0) & (0,0) & (1,0) \\ (-1,1) & (0,1) & (1,1) \end{bmatrix}$$

將此矩陣的 x,y 值套入 Gaussian Function 並正規化後就可以得到 3*3 的 Gaussian filter 了!

$$\begin{bmatrix} 0.045 & 0.122 & 0.045 \\ 0.122 & 0.332 & 0.122 \\ 0.045 & 0.122 & 0.045 \end{bmatrix}$$

在這題中，我們的 Gaussian filter 就直接用上面給的這一個
得到高斯函數後，就可以開始對圖像作卷積

以一張眞實照片舉例

這是一張雜訊很高的照片，我們將利用 Gaussian Filter 將其消除雜訊

首先我們必須先瞭解圖片是由 R, G, B 三個 channel 所組成的，因此我們必須先將圖片轉換成灰階狀態，其公式爲

$$0.299 \times R + 0.587 \times G + 0.114 \times B$$

計算完畢後，請將其四捨五入至整數位，做完的圖片會大概長這樣

最後再用上面計算的 Gaussian Filter 對整張圖片做卷積，便可獲得經由 Gaussian Filter 後的圖片

4. **convolution 卷積**
   由第三點 Gaussian Filter 算出來的那個 3×3 的矩陣將原本的整張圖片掃過去，這個動作被稱爲 **convolution 卷積**，那用下面這張圖解釋什麼是卷積

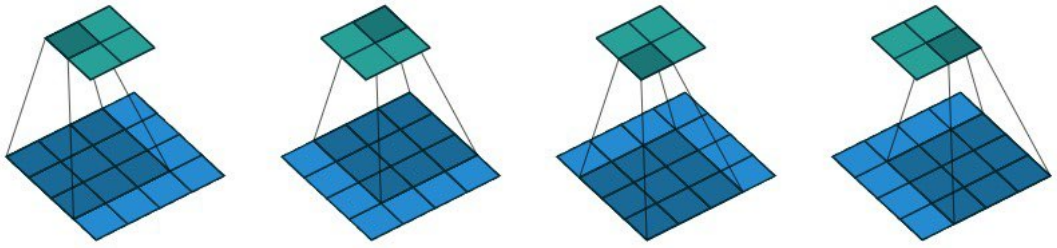


Figure 2.1: (No padding, unit strides) Convolving a 3×3 kernel over a 4×4 input using unit strides (i.e., $i = 4$, $k = 3$, $s = 1$ and $p = 0$).

四張小圖裡，下面藍色那個 4×4 的矩陣，可以視作我們上面提到的灰階圖片

每張小圖裡，會看到下方有一個 3×3 的深藍色小矩陣，那個深藍色矩陣爲 Gaussian Filter 當次處理的範圍，最後這個 3×3 的 Gaussian Filter 每一次會移動一格，直到把整張圖片全部掃過

那每一次 Gaussian Filter 對圖片裡 3×3 那個負責的區塊，要怎麼做處理?

答案是 內積
我們只要把 3×3 的區塊，跟 Gaussian filter 做內積就可以做到卷積的動作了!

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$B = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

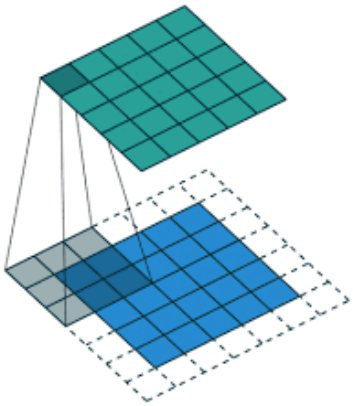
內積公式: $C = A \cdot B = C = \sum_{i=1}^{3} \sum_{j=1}^{3} a_{ij} \times b_{ij}$

以 $Figure1$ 舉例，Gaussian Filter 在上圖四個位置都進行過一次內積後，會得到四個數值，這四個數值分別對應到圖片上半部的綠色區塊，而那整個綠色區塊，就是整張圖片經過卷積的結果。

快結束了剩下最後一步。

5. **Padding**
聰明的你們可能會發現，經過卷積後，原本的圖片變小張了，以上圖舉例，原來 4×4 的圖片變小了，只剩下 2×2 的大小，可以思考一下，會發現每進行一次卷積，大小的長寬都會減少 2。

一般來說，遇到這種狀況會在做卷積前，周圍補一圈 0



用虛線畫出來的地方全部補 0 ，這樣就讓卷積過後，圖片的大小依然不變了

輸入格式

第一行有兩個數字 n m 代表圖片的長度與寬度
接這輸入三個 $n \times m$ 的陣列，中間以空白隔開，陣列跟陣列間以空白隔開，分別代表 RGB 三個 channel
在每個陣列裡有 n 行，每行有 m 個整數 num，每個整數中間以空白個隔開

輸出格式

輸出一個 $n \times m$ 的陣列

輸出有 n 行，每行有 m 個數字，每個數字間以空白隔開，爲 Gaussian Filter 處理過後的灰階圖片

技術規格

- $1 \leq n, m \leq 512$
- $0 \leq num \leq 255$

範例輸入 1

```
2 3
1 2 3
4 5 6
2 3 2
5 6 5
10 20 1
2 71 6
```

範例輸出 1

```
3 4 2
4 6 4
```