

中原大學資訊工程系 演算法分析第二次機測

Deadline: 6 / 14 / 2024 (星期五)
(限期末考前測完，逾期不得補繳)

【程式設計說明】

1. 每組限 2~3 人，組員須固定，本學期不得任意變更。原則上以專題組員為主。
2. 組員應合作共同解題，但嚴禁跨組合作。
3. 程式設計必須使用 Python 程式語言，版本請採用目前最新版本(原則上，請直接下載與安裝 Anaconda)。
4. 可參考課本、參考書籍或網站資料等進行解題，解題方法及演算法不限，但絕對嚴禁抄襲他組程式，組員均有責任保護程式不被他組抄襲。若發現抄襲屬實，兩組均以零分計。
5. 輸入與輸出採用標準格式或讀寫檔案方式進行。
6. 每一支程式均須附上組員姓名及學號，例如：

```
# 演算法分析機測  
# 學號: 11027XXX / 11027XXX  
# 姓名: 陳○○ / 林○○  
# 中原大學資訊工程系
```

程式命名依該組學號在前的同學 [學號+題號] 為原則。例如：

```
11027001_1.py  
11027001_2.py
```

【機測須知】

1. 評分以解題成功之題數多寡與執行時間決定。
2. 程式必須能處理不同的輸入資料(但輸入格式與範例相同)，並輸出正確結果(輸出格式必須與範例相同)，組員應能說明程式設計內容，方可視為成功。程式的輸出結果錯誤、輸出格式與範例不符、或在執行後超過 5 秒(以每筆測資為基準)仍未結束，均視為失敗。若程式測試失敗給予基本分數，未繳交程式則以零分計。
3. 本機測於規定之期限前，各組應攜帶程式原始碼至電學大樓 603 室找助教測試(電話：265-4726)。每組限繳交一次，不可分題或多版本繳交，逾期不得補繳。
4. 助教將使用不同之輸入資料作為測試與評分依據，同學應在繳交前充分測試程式。
5. 機測成績納入學期平時成績計算，請同學把握！

指導教授: 張元翔

【執行時間測試】

機測預計採用個人電腦 CPU Intel i7、8G RAM、作業系統以 Windows 10 為主。建議同學在繳交程式前先使用下列 Python 程式進行初步的執行時間測試：

```
import time
start_time = time.time()
.....
total_time = time.time() - start_time
print(total_time)
```

I. 最長增加子序列 (Longest Increasing Sequence)

最長增加子序列 (Longest Increasing Subsequence, LIS) 問題可以說明如下：

給定陣列 A ，目的是找到陣列 A 的子序列，這個子序列中的元素須由小到大排列，且子序列的長度最長。舉例說明，若陣列 A 為：

$$A = \langle 3, 10, 2, 1, 20 \rangle$$

則：

$$LIS = \langle 3, 10, 20 \rangle$$

長度為 3。

試編寫 Python 程式解決 LIS 問題。提示：本機測須使用動態規劃法。

輸入說明：

每組測試資料是陣列 A ，其中的元素均為正整數 (而且不會有相同的數字)，元素間以空格隔開；0 代表結束。

輸出說明：

輸出 LIS 的長度與 LIS；LIS 的元素以逗號隔開，且最後一個元素不能有逗號。每組測資間以空行隔開。

輸入範例：

```
3 10 2 1 20
50 3 10 7 40 80
0
```

輸出範例：

```
Length of LIS = 3
LIS = < 3, 10, 20 >
```

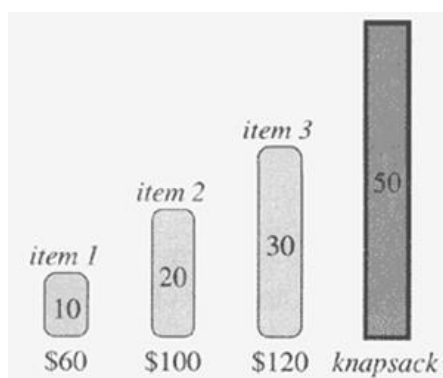
```
Length of LIS = 4
LIS = < 3, 7, 40, 80 >
```

II. 0-1 背包問題 (0-1 Knapsack)

0-1 Knapsack 問題 (又稱為 Bin-Packing 問題) 是電腦科學中非常具有代表性的問題。問題描述如下: 有一小偷到一家店內偷東西, 他發現 n 項物件, 每項物件各有不同價值及不同重量, 小偷的目的是帶走總價值最高的物件, 但他能帶走的**背包** (Knapsack) 有重量限制。試編寫程式解決 0-1 背包問題 (即每項物品僅能**取走或不取**, 無法取走部分), 並須求得**最佳解** (Optimal Solution)。

輸入說明:

輸入物品 Knapsack 重量 W 及物件總數 n , 接著分別是各項物件的重量及價值 (均為正整數), 中間以空格隔開; 0 代表結束。以下為輸入範例:



輸出說明:

求出最高總價值, 並列出取走物件的編號; 物件須按編號由小到大順序排列, 並以逗號隔開, 且最後一個物件編號不能有逗號。每組測資間以空行隔開。

輸入範例:

```
50
3
10 60
20 100
30 120
0
```

輸出範例:

```
Total Value = 220
Items = 2, 3
```

III. 霍夫曼碼 (Huffman Codes)

霍夫曼碼在資料壓縮中是常見的技術之一，被廣泛使用在音訊、影像、視訊等多媒體壓縮應用中。霍夫曼碼的主要原理是由於表示資料的方式可以分成兩種，若使用**固定長度字碼** (Fixed-Length Codeword)，則每一個字元是以固定長度的編碼方式；霍夫曼碼是比固定長度編碼更為有效的編碼方式，採用**可變長度編字碼** (Variable-Length Codeword) 的方式。

以下述字元編碼為例，試設計程式完成霍夫曼碼的**編碼** (Encoding) 及**解碼** (Decoding)。

	a	b	c	d	e	f
Frequency (in thousands)	45	13	12	16	9	5
Fixed-length codeword	000	001	010	011	100	101
Variable-length codeword	0	101	100	111	1101	1100

輸入說明：

每組輸入包含的字元數 n (均為正整數)，0 表示結束，緊接為每一個字元及其發生頻率，所有字元均可能是英文字母大或小寫，且頻率均為正整數 (但不會事先排序)，中間以空格隔開。最後，則是給定一段二元碼。

輸出說明：

就每組輸入列出結果，包含：每一個字元的霍夫曼碼。此外，輸出解碼的結果。每組測資間以空行隔開。

輸入範例：

```
6
a 45
b 13
c 12
d 16
e 9
f 5
010110011111011100
6
A 2
B 6
C 15
D 12
```

E 8

F 3

011010010

0

輸出範例:

a = 0

b = 101

c = 100

d = 111

e = 1101

f = 1100

Decode = abcdef

A = 0100

B = 011

C = 11

D = 10

E = 00

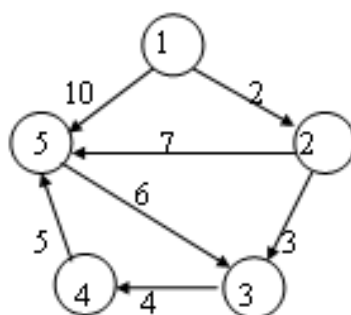
F = 0101

Decode = BAD

IV. 最短路徑問題 (Shortest Paths Problem)

給定一有向圖 $G=(V, E)$ 與源頂點 (Source Vertex)，單一源最短路徑 (Single-Source Shortest Paths) 問題的目的是找到 Source Vertex 與其他頂點的最短距離。

舉例說明，下圖為典型的有向圖，有向圖的權重 (weights) 代表兩頂點的距離，在此均為正整數，並具有方向性。假設頂點的編號分別為 $1 \dots n$ ，試設計程式輸出 Source Vertex 與其他頂點的最短距離。



輸入說明：

每組輸入含頂點個數 n (原則上頂點數不超過 20，0 代表結束) 與邊 (Edges) 的個數，接著為 Source Vertex 的編號，最後列出連接每個邊的兩個頂點與權重，其間以空格隔開。

輸出說明：

Source Vertex 至其他頂點的最短距離。

輸入範例 (見上圖)：

```
5
7
1
1 2 2
1 5 10
2 3 3
2 5 7
3 4 4
4 5 5
5 3 6
0
```

輸出範例：

$$1 \text{ to } 2 = 2$$

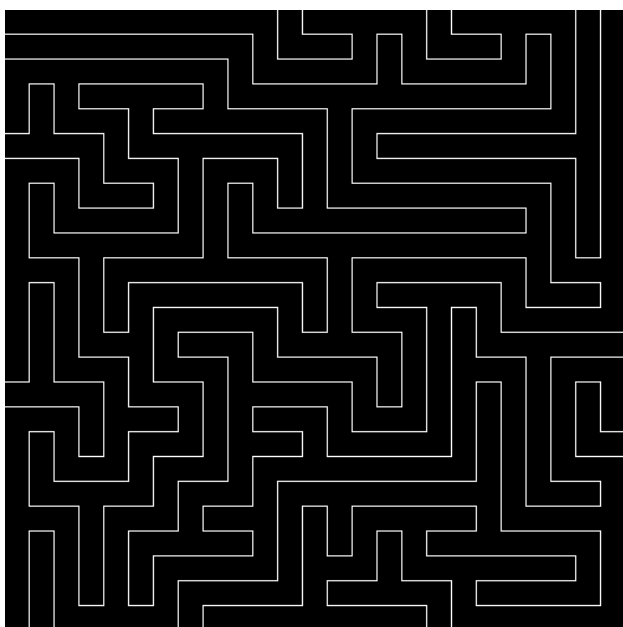
$$1 \text{ to } 3 = 5$$

$$1 \text{ to } 4 = 9$$

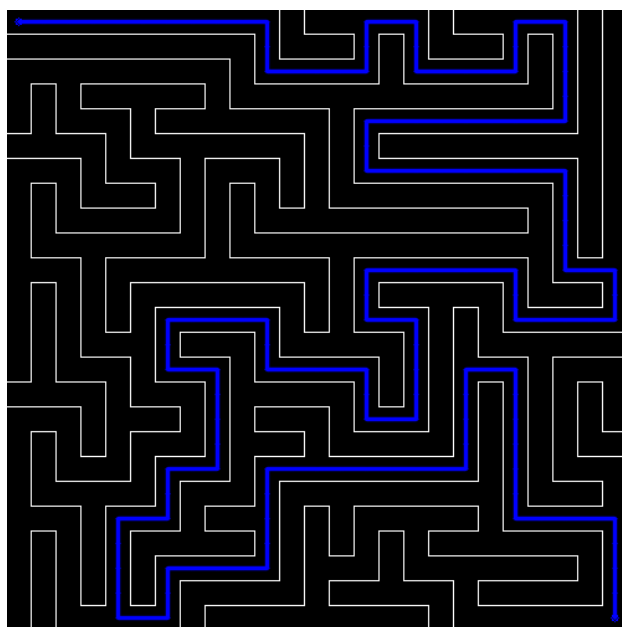
$$1 \text{ to } 5 = 9$$

V. 迷宮問題 (Maze Problem)

迷宮 (Maze) 是常見的益智遊戲，典型的迷宮如下圖，這個迷宮的最短路徑，以藍線表示。



輸入影像



輸出影像

本問題的輸入是一張數位影像，定義如下：

1. 數位影像為色彩影像 (即 24-bit RGB 數位影像)，影像大小固定為 500×500 像素。
2. 每個像素包含 3 個 8-bit 的資料，分別代表 B、G、R 值。輸入影像僅含黑色 (0, 0, 0) 與白色 (255, 255, 255)。
3. 每個小方格是 20×20 像素，因此共有 25×25 個小方格。
4. 出發點固定為左上角、終點固定為右小角。

試設計 Python 程式，根據輸入影像，輸出一張數位影像，最短路徑以藍線標示之，典型的迷宮與最短路徑如圖所示。

補充說明：本機測問題提供的迷宮一定有路徑。

【提示】你可以使用 OpenCV 程式庫進行數位影像處理。首先，須安裝 OpenCV：

```
pip install opencv-python
```


數位影像的讀取方式為：

```
import numpy as np
import cv2
img = cv2.imread("maze1.bmp", -1)
```

讀入的 `img` 是一個 Numpy 二維陣列，其中 `img[x, y, :]` 代表座標 (x, y) 的 B、G、R 值，例如：`img[0, 0, 0]` 為座標 $(0, 0)$ 的 B 值、`img[0, 0, 1]` 為座標 $(0, 0)$ 的 G 值、依此類推。

輸入說明

本機測題使用讀檔方式進行，例如：

Enter file name: **maze1.bmp** 


輸出說明

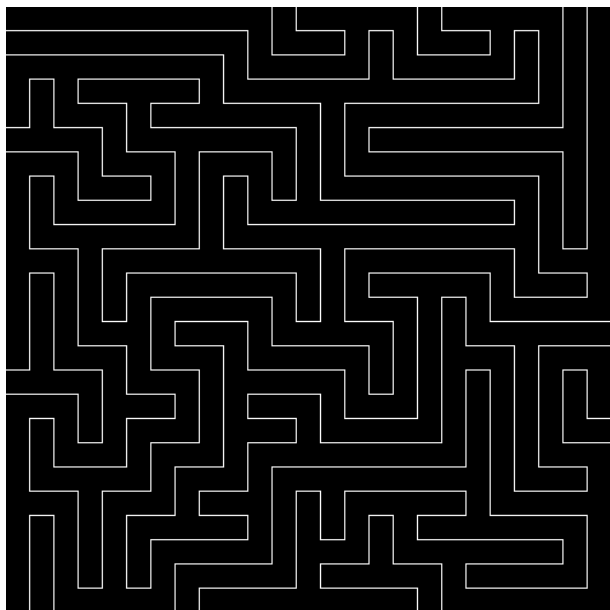
輸出色彩影像，根據輸入影像繪製最短路徑(藍色)，藍線為 2 像素寬。

檔案名稱為：**maze1sol.bmp**

【注意】若為 **maze2.bmp**，則輸出 **maze2sol.bmp**，依此類推。

輸入範例

Enter file name: **maze1.bmp** 



輸出範例

