中原大學資訊工程系 演算法分析第二次機測

Deadline: 6 / 26 / 2020 (星期五)

(限期中考前一週測完，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，版本採用 3.7 (原則上下載與安裝 Anaconda)。
4. 可參考課本、相關書籍或 Algorithms.py 等解題，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。<u>若發現抄襲屬實，兩組均以零分計。</u>
5. 所有輸入及輸出均為標準格式，即程式在命令提示字元環境下執行時可以鍵盤輸入資料，本機測不採讀檔方式進行。
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: 10827XXX / 10827XXX
#   姓名: 江○○ / 李○○
#   中原大學資訊工程系
```

程式命名依該組學號在前之同學 [學號+題號] 為原則。例如:

10427001_1.py

10427001_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之輸入資料 (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。程式之輸出結果錯誤、輸出格式與範例不符、或在執行後超過 60 秒仍未結束，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

I. 找零錢問題 (Making Change Problem)

電腦演算法中，**找零錢問題** (Making Change Problem) 是一個具有代表性的問題，問題描述如下：給定零錢數 x 與硬幣的面額，目的是找零錢 x，且使用的硬幣數最少。請設計程式，解決找零錢問題。

請注意：這個程式須能處理各種可能的面額，而且產生**最佳解** (Optimal Solution)。

輸入說明:
輸入包含零錢數與面額，第 1 行為零錢數，第 2 行為面額，從小到大排列，並以空格隔開。若零錢數為 0，則結束。

輸出說明:
輸出包含第幾個案例、各種面額的數量按順序列出。若該面額為 0 枚，則不列出。

輸入範例:
51
1 5 10 50
36
1 5 10 25
8
1 4 6
0

輸出範例:
Case 1
1 元 1 枚
50 元 1 枚

Case 2
1 元 1 枚
10 元 1 枚
25 元 1 枚

Case 3
4 元 2 枚

II. 霍夫曼碼 (Huffman Codes)

霍夫曼碼在資料壓縮中是常見的技術之一，被廣泛使用在文字、訊號、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**可變長度編碼** (Variable-Length Codeword) 的方式。

以下述字元編碼為例，設計程式完成霍夫曼碼的**編碼** (Encoding) 與**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)。最後，給定特定的二元編碼，試使用霍夫曼碼對其進行解碼。

輸出說明:
就每組輸入列出結果，包含：(1) 每一個字元的霍夫曼碼；及(2) 解碼之結果。

輸入範例:
6
a 45
b 13
c 12
d 16
e 9
f 5
01001101
6
A 2
B 6
C 15
D 12
E 8

F 3
010101001100
0

輸出範例:

Huffman Codes #1
a 0
b 101
c 100
d 111
e 1101
f 1100
Decode = ace

Huffman Codes #2
A 0100
B 011
C 11
D 10
E 00
F 0101
Decode = FACE

III. 少女團體 (Teen Squad)

假設妳是一群有手機的少女團體的其中一員。妳有一些消息要告訴少女團體的每個女孩。問題是妳們都不在同的地方，所以妳們只能用手機傳達這些訊息。更糟的是，妳的父母因為妳過度使用，拒絕支付妳的電話費。所以妳必須以最便宜的方法透過電話散佈這些消息。妳會 call 幾個妳的朋友，她們會 call 一些她們的朋友，如此直到所有人都知道這些消息。

妳們每個用的手機服務提供者都不一樣，因此 A call B 與 A call C 的價錢可能不一樣。另外，並不是所有妳的朋友都喜歡彼此，而且有些人永遠都不要 call 她不喜歡的人。現在妳的工作是找出最便宜的方法，讓所有的人都知道這些消息。

輸入說明:

每一組測試資料的第一行包含 n (0 <= n <= 100) 和 m (0 <= m <= 100)，0 0 代表結束。女孩的編號從 1 到 n，妳是女孩 1。接下來的 m 行中，每行都包含三個數字 u、v 和 w 意思是女孩 u call 女孩 v (或女孩 v call 女孩 u) 的成本是 w (0 <= w <= 1000)，沒有提到的表示因為討厭對方而不可能 call。

輸出說明:

對每組測試資料，輸出發佈消息最便宜方法的花費。

輸入範例:

```
4 4
1 2 10
1 3 8
2 4 5
3 4 2
5 7
1 2 2
1 4 10
1 5 6
2 3 5
2 5 9
3 5 8
4 5 12
0 0
```

輸出範例:

Case 1
Minimum Cost = 15

Case 2
Minimum Cost = 23

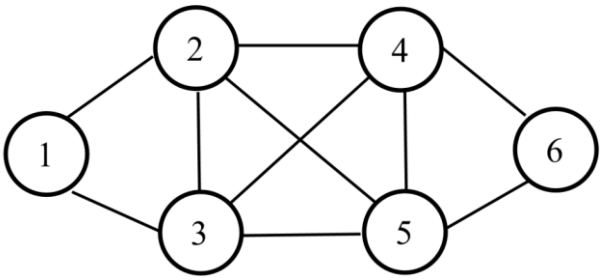
IV. 歐拉旅程問題 (Euler Tour Problem)

歐拉旅程 (Euler Tour) 問題，又稱為「一筆畫問題」。假設給定一張**無向圖** (Undirected Graph)，目的是對於圖的每個**邊** (Edge) 都僅走訪一次，而且起點與終點相同，形成一個**旅程** (Tour)。

假設無向圖可以定義為 $G(\boldsymbol{V}, \boldsymbol{E})$，其中頂點集合為 $\boldsymbol{V}$，邊的集合為 $\boldsymbol{E}$，則頂點數 (Number of Vertices) 為$|\boldsymbol{V}|$，邊的個數 (Number of Edges) 為 $|\boldsymbol{E}|$。在此，頂點的編號為 1 ~ $|\boldsymbol{V}|$。

若以下列的無向圖為例，則可能的 Euler Tour 為：

< 1, 2, 4, 5, 6, 4, 3, 5, 2, 3, 1 >



輸入說明:

輸入均為無向圖，第 1 行為**頂點數**與**邊的個數**。接著，每一行列出各個邊的兩個頂點。若頂點數與邊的個數為 0，則結束。

輸出說明:

輸出包含第幾個案例與 Euler Tour。若無 Euler Tour 則顯示訊息「No Euler Tours」。由於 Euler Tour 不是唯一解，因此程式只要列出其中一解即可。

輸入範例:

3 3 (粗體用來方便觀察每個案例)
1 2
1 3
2 3
6 10 (本案例如上圖)
1 2
1 3
2 3
2 4

2 5
3 4
3 5
4 5
4 6
5 6
4 5
1 2
1 3
1 4
2 4
3 4
0 0

輸出範例:

Case 1
< 1, 2, 3, 1 >

Case 2
< 1, 2, 4, 5, 6, 4, 3, 5, 2, 3, 1 >

Case 3
No Euler Tours

V. 尋寶遊戲 (Getting Gold)

我們正在建立一個電腦遊戲，一個冒險遊戲。玩家在一個平面上到處行走，試著找到黃金，並且不要掉到陷阱中，而玩家對周遭的訊息知道得很有限。玩家可以上、下、左、右 (不能對角) 移動。假如他走到黃金所在的格子，他就可以撿起黃金。假如玩家走到某格子，而這格子緊鄰著一或多個陷阱，他會感應到附近有陷阱，但是不知道陷阱在什麼方位，或緊鄰幾個陷阱。假如玩家試著要走到牆的格子，他會知道這是一面牆，並且停留在原來的位置。

為了分數的目的，我們想要告訴玩家他可以"安全的"得到多少個黃金。也就是在某一最佳策略下，玩家總是確認他走到的格子是安全的，且可以得到最多的黃金。

玩家一開始沒有地圖的資訊，除了確定玩家一開始的位置不是陷阱之外，他擁有的就只有感應到旁邊有陷阱的能力了。(請參考範例輸入，你會更清楚題目的意思)。

輸入說明:
輸入含有多組測試資料。

每組測試資料的第一列，含有 2 個正整數 W, H ($W \le 50, 3 \le H \le 50$)，代表平面矩陣的寬與高。接下來有 H 列，每列有 W 個字元，表示地圖的內容。字元可能為下列幾種:

P - 玩家一開始的位置
G - 一個黃金
T - 一個陷阱
- 一面牆
. - 正常的地面

輸入只會有一個 P，並且地圖的邊界一定都是牆。

如果讀到 0 0 則代表輸入結束。

輸出說明:
對每組測試資料輸出一列 ，輸出玩家在不讓自己處於掉入陷阱風險的情況下，最多可以得到幾個黃金。

輸入範例:
7 4
#######

#P.GTG#
#..TGG#
#######
8 6
########
#...GTG#
#..PG.G#
#...G#G#
#..TG.G#
########
0 0

輸出範例:

1
4