# KMIT – ARJUNA

## Season-5

**KMIT-APA-5004** **Programming Assignments** **Sunday 9th Feb, 2020**

## 1 Analysis of Tiffins Ordering Pattern

A popular online food delivery chain wanted to analyze the patterns of the tiffin orders placed by its users at different time intervals. The information of the users, time intervals and tiffin-items ordered are stored in three different arrays namely username [], timestamp [] and tiffin items []. We can assume that a user, **username[i]** has ordered the item, **tiffinitem[i]** at time **timestamp[i].**

A 3-sequence is a list of 3 tiffin item ordered in the increasing order of timestamp. (3-sequence are not necessarily distinct.) The items are sorted in ascending order by the time of ordering.

Note the following:

- Both username[i] and tiffinitems[i] contain only lowercase characters.
- It is guaranteed that there is at least one user who ordered at least 3 tiffin items.
- No user orders two tiffin items at the same time.

For example:

**tim tim tim bob bob bob bob ann ann ann**
**1 2 3 4 5 6 7 8 9 10**
**idly mdosa upma idly mdosa pattu pdosa idly mdosa upma**

Find the 3-sequence ordered by the largest number of users. If there is more than one solution, return the lexicographically smallest such 3-sequence.

### Input/Output

| Input | Output | |
|---|---|---|
| *tim tim tim bob bob bob bob ann ann ann*<br>*1 2 3 4 5 6 7 8 9 10*<br>*idly mdosa upma idly mdosa pattu pdosa idly mdosa upma* | **idly**<br>**mdosa**<br>**upma** | |

### Explanation:

- First line indicates list of users (Space seperated)
- Second line indicates timestamps of each user (Space seperated)
- Third line indicates tiffin items ordered by each user (Space seperated)

**The tuples in this example are:**

- ["tim", 1, "idly"], ["tim", 2, "mdosa"], ["tim", 3, "upma"], ["bob", 4, "idly"], ["bob", 5, "mdosa"], ["bob", 6, "pattu"], ["bob", 7, "pdosa"], ["ann", 8, "idly"], ["ann", 9, "mdosa"], ["ann", 10, "upma"]

- **The 3-sequence**

| | | |
|---|---|---|
| ○ **("idly", "mdosa", "upma") was ordered at least once by 2 users.**<br>○ **("idly", "mdosa", "pdosa") was ordered at least once by 1 user.**<br>○ **("idly", "pattu", "pdosa") was ordered at least once by 1 user.**<br>○ **("idly", "mdosa", "pattu") was ordered at least once by 1 user.**<br>○ **("mdosa", "pattu", "pdosa") was ordered at least once by 1 user.** | | |
| Sample-2<br><br>tim tim tim bob bob bob bob ann ann ann ann ann ann tim tim tim<br>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16<br>idly mdosa upma idly pattu pdosa idly idly mdosa upma idly pattu pdosa idly pattu pdosa | **idly**<br>**pattu**<br>**pdosa** | |
| **Explanation:**<br>The 3-sequence *("idly", "pattu", "pdosa")* was ordered at least once by 3 users (ascending order by the time of their ordering) | | |

## 2 How many allies far from me?

In medieval period, in East Asia, there were several small kingdoms. Every kingdom was an ally with other neighboring kingdoms. The whole region appears as a giant connected graph.

Given the graph, find out the total number of kingdoms allied at **t** distance away from each other (i.e., t distance connectivity). For example, two kingdoms are directly connected are at **1** distance connectivity. While the two kingdoms having a common ally without having directly connectivity (immediate neighbor), are at **2** distance connectivity.

First line of input line contains, two integers n and e, where ***N*** is the kingdoms (nodes) and ***E*** are the edges. Next e line will contain two integers *U and V* meaning that node u and node v are connected to each other in undirected fashion. Next line contains single integer, *m,* which is number of queries. Next *m* lines, each have two inputs, one as source node and other as a required **t** distance connectivity which should be used to process query.

### Input/Output

| Input | Output | Comments |
|---|---|---|
| 9 10<br>0 1<br>1 2<br>0 6<br>1 3<br>2 3<br>3 6<br>6 7<br>8 6<br>6 5<br>4 5<br>3<br>3 2<br>4 3<br>1 1 | 4<br>4<br>3 | • First line 9 10 represents the number of kingdoms (nodes) and number of edges (allies)<br>• Each line of the next 10 lines represents IDs of kingdoms that are direct allies (undirected connection)<br>• Next m lines, each have two inputs, one as source node and other as a required t distance connectivity which should be used to process query<br>• 3 represents number of queries<br>• 3 2: Source node: 3, and we have to find out total number of nodes at a distance of 2 from node 3.<br>○ 0 (3->1->0), 7 (3->6->7), 9 (3->6->8), 5 (3->6->5) = **4**<br>• 4 3: Source node: 4, and we have to find out total number of nodes at a distance of 3 from node 4.<br>○ 0 (4->5->6->0), 3 (4->5->6->3), 7 (4->5->6->7), 8 (4->5->6->8) = **4**<br>• Source node: 1, and we have to find out total number of nodes at a distance of 1 from node 1.<br>○ 0 (1->0), 3 (1->3), 2 (1->2) = **3** |

## 3 Find delivery outlet

A popular courier services company has many delivery outlets spread in various streets in a city. The streets of the city are arranged in a 2D grid of N rows and N columns. Each cell of the grid is assigned either 0 or 1. 0 represents a delivery-outlet and 1 represents delivery-hub.

Given the grid containing 0s and 1s, find out delivery-outlet such that its distance to the nearest delivery hub is maximized and **return the distance**.

Note:

- The distance between two cells (x0, y0) and (x1, y1) is |x0-x1| + |y0-y1|
- If there are delivery outlets only or delivery hubs only, then return -1.

### Input/Output

| Input | Output | Comments |
|---|---|---|
| **3**<br>**1 0 1**<br>**0 0 0**<br>**1 0 1** | **2** | • First line 3 size of the square grid.<br>• Next 3 lines represent delivery-outlet (0) and delivery-hub (1).<br>• The delivery-outlet (1, 1) is as far as possible from all delivery-hubs with distance 2**.** |
| **3**<br>**1 0 0**<br>**0 0 0**<br>**0 0 0** | **4** | • First line 3 size of the square grid.<br>• Next 3 lines represent delivery-outlet (0) and delivery-hub (1).<br>• The delivery-outlet (2, 2) is as far as possible from all delivery-hubs with distance **4**. |
| **3**<br>**1 0 1**<br>**0 0 0**<br>**0 0 1** | **2** | • First line 3 size of the square grid.<br>• Next 3 lines represent delivery-outlet (0) and delivery-hub (1).<br>• The delivery-outlet (2, 0) is as far as possible from all delivery-hubs with distance **2**. |