Discrete Structures for Computer Science (CS F222)

Assignment 2: Graph Theory

1. **XYZ** campus has decided to i**ncrease the bandwidth** of the campus. They have decided to make use of brand new fiber optic technology. However, the technology has a special property, that the **cost of laying the new cable is multiplied to the initial cost**. Equivalently speaking if the **initial cost** is **i** and the **cost of the new cable** is **x** then the **cost of adding the cable** is **i*x**. Let the **cost of connecting the campus to the internet backbone** is 1 and thus the **initial cos**t is always taken to be **1**.

   Given the **number of switches** and the **cost of connecting the switches**, find the **minimum cost of connecting all the switches on campus**. Assume that all the switches are implicitly labelled from 0 to **n-1** where **n** is the number of switches. Let the **0th** switch be the one connected to the backbone.

   Input:
   n is the number of switches
   m is the number of connections between the switches
   Following **m** lines contain triplets **i j k** where it shows a bidirectional connection between two switches **i** and **j** and it costs **k** birr for that connection.
   Output:
   An Integer that gives the **cost.**

   Sample Test Case 1
   Input:
   4
   6
   0 1 81
   0 2 43
   0 3 16
   1 2 22
   1 3 5
   2 3 4
   Output:
   25
   Explanation:
   The minimum cost will consist of the following edges [(0,3),(3,1),(3,2)]
   Sample Test Case 2
   Input:
   5
   6
   0 1 1
   1 2 2
   2 3 3
   3 4 5
   3 0 4
   4 0 6
   Output:
   11
   Explanation:
   The minimum cost will consist of the following edges [(0, 1), (1, 2), (2, 3), (3, 4)]

2. You being a lucky person have just won the lottery of a billion **pula** from a Nigerian Email. Since you have been living in **Thumkunta** all your life, now you have got an opportunity to explore the world. You want to visit a set of popular tourist destinations around the world. Given that the cost of travelling is **directly proportional** to the **distance** between the cities, find the **cheapest** way to travel around the world and return to your home in **Thumkunta**. As you want to make the most of your money, you **cannot visit any city more than once**.

   Input:
   n is the number of cities implicitly labelled from 0..**n-1**
   m is the number of connections between cities
   The next **m** lines contain pairs **p**,**q,r** stating that a **bidirectional** mode of transport exists between **p** and **q** and costs **r** to make use of that transportation mechanism.
   Output: [If there are multiple solutions, give any one]
   The path taken
   An Integer that gives the total distance travelled.
   Sample Test Case 1
   Input:
   4
   6
   0 1 81
   0 2 43
   0 3 16
   1 2 22
   1 3 5
   2 3 4
   Output:
   A => D => B => C => A
   Cost: 86
   Sample Test Case 2
   Input:
   3
   3
   0 1 1
   1 2 2
   2 0 3
   Output:
   A => B => C => A
   Cost: 6

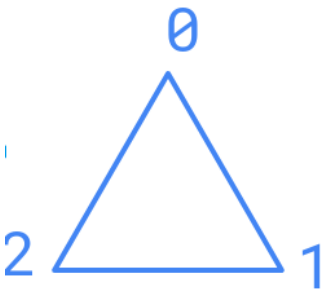
Hint: For your problem, the number of places to be visited will always be less than 5. There are several approaches to solve this problem. You are free to make use of the following approach if required.

You need to make use of the following methodology when you traverse the graph.

1. Firstly, for any node we need to first register the current node, to show that it has been visited.
2. This is followed by a check for the nodes which are connected to the current node and have not been visited by the current traversal.
3. We then go to such a connected and unvisited node.
4. Repeat the steps 1 to 3.
5. After we have visited the node,(after the recursive call to the function); we set the visited state of the current node to false.

This allows us to visit the nodes in all the possible permutations:

For example if we have a cycle of the form



If we start the graph traversal, starting at 0
Visited 0 1 2
status T F F

We then go to 1
Visited 0 1 2
status T T F

The go to 2
Visited 0 1 2
status T T T

Undoing the changes:

Exiting 2
Visited 0 1 2
status T T F

Exiting 1
Visited 0 1 2
status T F F

Going to 2
Visited 0 1 2
status T F T

Going to 1
Visited 0 1 2
status T T T

This covers both the ways of traversing the graph:

0->1->2
And
0->2->1

3. **In the November of 2029** in Prague, Czech Republic, **Adam Jensen** as a member of **Interpol** has been tasked in the aftermath of the Aug Incident to hack into the vault of **Tai Yong Medical**. Your job as an expert hacker is to find the **weaknesses** within the security network. A **critical point** within a network is a node, if disconnected can **split** the network into **two components**. Your job is to f**ind all such weaknesses** and report back to Jim Miller.

   Input:
   n is the number of nodes within the network
   m is the number of connections between nodes

   The next **m** lines contain pairs **p**,**q** stating that a **bidirectional** connection exists between the nodes.

   Output:
   All the nodes which are `**critical**` in nature

   Sample Test Case 1
   Input:
   6
   7
   0 5
   0 1
   1 3
   1 2
   3 4
   2 4
   2 3
   Output:
   0 1
   Explanation:
   If we remove 0 we get the components [5] and [1,2,3,4]
   If we remove 1 we get the components [0,5] and [2,3,4]

4. **Spoderman** was able to successfully **defeat** the devious symbiote **Vonom**, however, it so happened that Vonom was able to **gain some of spodermon's powers**. The scientists at **Shield** have been able to ascertain that they can find the degree to which vonom has acquired the powers by **comparing their web structures**. The webstructures match if we can find a **one to one mapping for each node in one web structure to the other**. Thus, if **two nodes in the web are adjacent** to another in **spodermon's web** they have to also be **adjacent in the web of vonom** as well.
You need to find out if vonom has **completely acquired all of his powers**.

   Note: You will have to account for n! possible mappings and you will be solving this problem for n ≤ 5.

   Input:
   n the **number of nodes** in spodermon's web
   m the **number of connections** within the web.

   The next **m** lines have pairs of the form **p** and **q** where the node **p** is connected to **q**

   a is the **number of nodes** in vonom's web
   b is the **number of connections** within the web

   The next **b** lines have pairs of the from **t** and **v** where the node **t** is connected to **v**

   Output:
   `**Yes**` if the two maps exactly match one-another and `**No**` otherwise

   Sample Test Case 1
   Input:
   4
   4
   0 1
   1 2
   2 3
   3 0
   4
   4
   0 2
   2 1
   1 3
   3 0
   Output:
   Yes
   Explanation:
   We can find a match of the following form:
   0->0
   1->2
   2->1
   3->3

5. In a **parallel dimension**, the **Sphinx** is a little more **mathematically oriented** and gives **Oedipus** the following puzzle. Given a list of numbers, is it possible to find which number will give the **largest value of xor** with respect to a user input **without** having to iterate through them one at a time. In other words, if we were to have **t inputs** and an **array of size n**, is it possible to find the **largest value of xor for each candidate**, **without** having to perform **t*n** computations.

   Input:
   n is the size of the list
   The following line has n integers delimited by a space

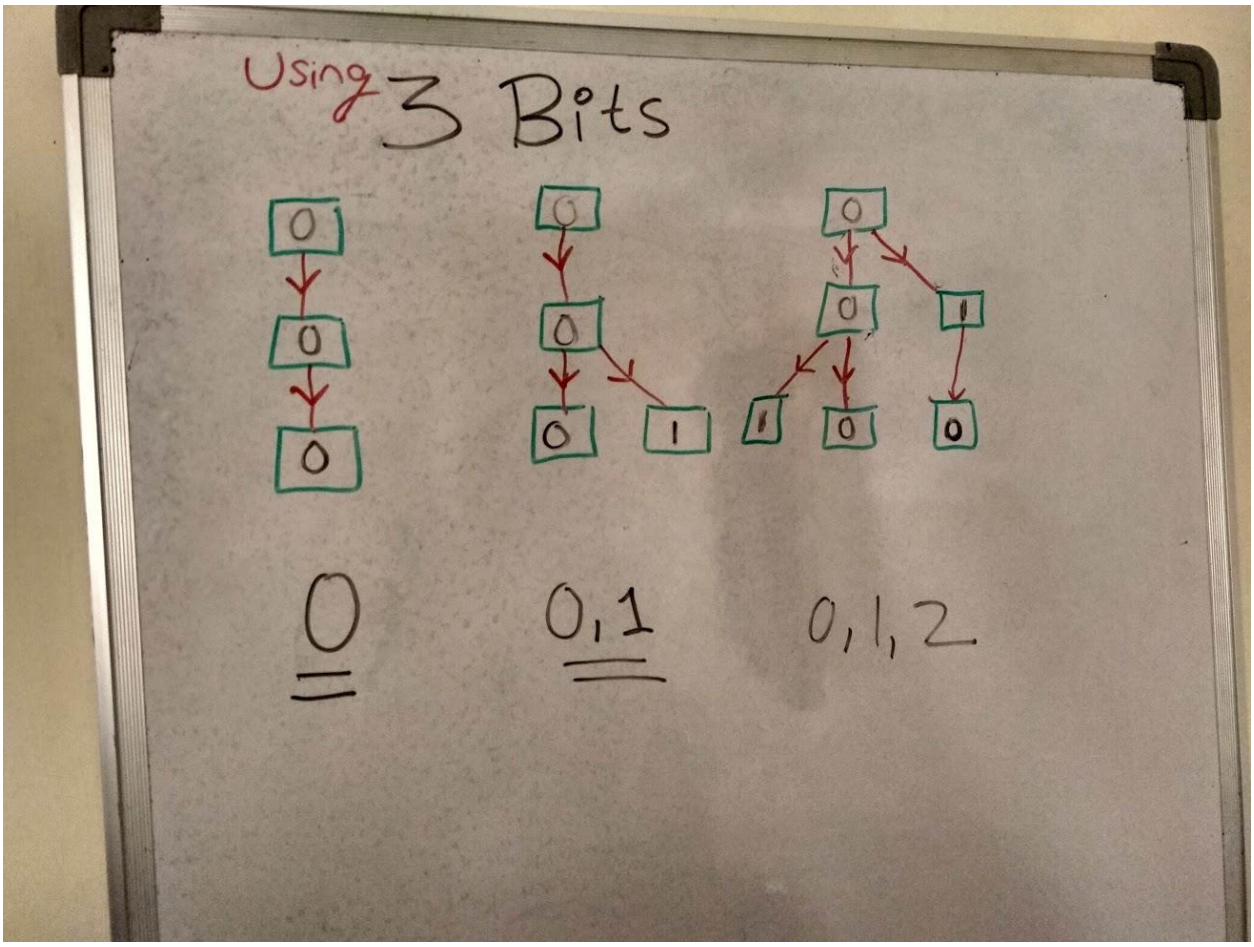
   t is the number of test cases
   The following line has t test cases delimited by a space

   Output:
   The value for the maximum value of xor for each input

   Sample Test Case 1
   Input:
   3
   0 1 2
   3
   3
   7
   2

   Output:
   3
   7
   3
   Explanation:
   max(3^0 , 3^1 , 3^2)
   = the maximum value is obtained for 3^0=3
   max(7^0 ,7^1 , 7^2)
   = the maximum value is obtained for 7^0=7
   max(2^0 , 2^1 , 2^2)
   = the maximum value is obtained for 2^1=3

Solution Approach for test case 1:



Explanation:

We have create a graph, for the binary representation of each number. Here, we try to combine the three numbers so as to represent them in the form of a graph. Thus, a single traversal of the graph will allow us to find which number gives the maximum xor. The methodology for creating the graph has been given below:

To only represent 0, for a 3 bit representation, we only need 3 zeros (000)
Now, to add 1 to the existing information, we just add a 1 the end as the first two zeros are common to both 0 and 1.(000,001)
To add a two to this aggregation, we have only a zero common between the three, thus, we only take the first zero to be common. (000,001,010)

Intuition:
The reason for performing this conversion is that we are now able to represent the three numbers as one graph, this allows us to trick the sphinx, as we do not need to iteratively compare the numbers within the array.

Spoiler:
The key towards finding the solution lies in the fact that you have to find the complement of the current number and traverse through the graph and find out which gives the most number of common elements.

6. **Okabe Rintarou** needs to save **Mayourii** from **dying on Friday the 13th**. In order to do so, he needs to **travel to the correct dimension** from his **current dimension**. After doing a lot of math, which is **beyond the scope of this question**, he realizes that he needs to travel to the **dimension y** from his **current dimension x. Where x is an integer.** He can only make use of **two operations**-

    a. Operation 1: **<< x**
    b. Operation 2:**<<(<<(<<x))** + **<<x** + 1
       Here **<<** stands for left shift. **<<x** gives the number obtained by doing a left shift on the bits of **x**

Since, he has only **limited energy z** within his time travelling device, find out if it is possible to reach the **required dimension** in the limited number of tries.
[In short: Is there any way to go from an integer x to another integer y given that you can only partake in z jumps.]

Input:
t is the number of test cases
Each Test Case has the following information
x is the starting dimension
y is the target dimension
z is the energy within the device.

Note:
Each hop will take one unit of energy to take place

Output:
For every test case print `Yes` if it is possible to do so and `No` in all the other cases.

Sample Test Case 1
Input:
1
2
162
10
Output:
Yes
2 => 4 => 8 => 81 => 162

1. <<2
   4
2. <<4
   8
3. <<(<<(<<8)) + <<(8) +1
   64+16+1 = 81
4. <<81
   162

In case of any queries contact: f20150647@hyderabad.bits-pilani.ac.in

Note:

The following resources are useful:
https://www.hackerearth.com/practice/algorithms/graphs/graph-representation/tutorial/

This link is a landing page in which you can obtain, information on how to solve graph related problems.