Currency Exchange

# Description

A new clerk is employed in a currency exchange office but he doesn’t have enough experience to differentiate between currencies. He decided to ask indirect questions to the customers to widen his knowledge. He knows that the **USA Dollar ($) should be known by all customers**. Each customer **at least** knows his currency and USD dollar. **An American customer knows only his own currency**. All he can do with the customer is to show him a currency and ask him if he knows it or not (so as nobody notice his lake of experience).

Suppose the clerk has N currencies including USD dollar $. Show the **most efficient way** that the clerk gets to know the shape of the USD dollar currency. He is given a **N (customers) × M (currencies) matrix** containing whether a customer ***i*** knows the currency ***j*** or not. The clerk has to find the index of the USD Dollar $ with the smallest number of questions.

Note:

1. The clerk doesn't know the nationality of the customers and thus doesn't know the nationality of the American customer.
2. Each customer knows at least the USA dollar $ and more currencies

## Complexity

The complexity of your code should be **less than O(N2)**

Input: **Already Implemented**

The first line of input is an integer T (T < 2,000), that indicates the number of test cases. Each case consists of two integers that represent the number of customers (N) and the number of currencies (M) in a separate line. The next N lines represent the N×M matrix which contains customer ***i*** knows the currency ***j*** “Y” or not “N”.

Output: **Already Implemented**

The index of the US Dollar $ (zero-based).

Function: **Implement it!**

int CheckUSD(int N, int M, bool [,] knows)

It takes the number of customers (N), the number of currencies (M) and the Boolean matrix that indicates whether customer ***i*** knows the currency ***j*** or not. It should return the index of the US Dollar ($).

# Template

* [C# template](CurrencyExchange/CurrencyExchange.sln)

# Test Cases

N = 8, M = 6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 |
| 0 | N | N | N | N | N | Y |
| 1 | N | N | N | N | N | Y |
| 2 | N | N | N | N | N | Y |
| 3 | N | N | N | N | N | Y |
| 4 | N | N | N | N | N | Y |
| 5 | N | N | N | N | N | Y |
| 6 | N | N | N | N | N | Y |
| 7 | N | N | N | N | N | Y |

index of US Dollar is **5**

N = 5, M = 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 |
| 0 | N | Y | Y | Y |
| 1 | N | Y | Y | Y |
| 2 | N | Y | Y | Y |
| 3 | N | N | Y | N |
| 4 | N | Y | Y | Y |

index of US Dollar is **2**

# C# Help

If you need any help regarding the syntax of C#, **ask any TA during the office hours**.

## Creating 1D array

int [] array1D = new int [size]

## Creating 2D array

int [,] array2D = new int [size1, size2]

## Getting the size of 1D array

int size = array1D.GetLength(0);

## Getting the size of 2D array

int size1 = array2D.GetLength(0);

int size2 = array2D.GetLength(1);

## Sorting single array

Sort the given array "items" in ascending order

Array.Sort(items);

## Sorting parallel arrays

Sort the first array "master" and re-order the 2nd array "slave" according to this sorting

Array.Sort(master, slave);