**Python sprint – Session 1**

Hey everyone!

So let’s start with our first session of Python sprint. Today the agenda is:

• Basic programming concepts

• Problem solving approach

• Some samples to try out

So we will start. Before getting into how to code with python, you need to strong in the basic programming concepts. We are sure that most of you would have learned it during your high school or college. Let’s refresh it a bit.

If…Else

Switch case

Are some of the examples of condition statements where if the condition is true, you perform certain action and if it is false, you perform something else.

For

While

Are examples of loops where you iterate through an array or do certain operations until a condition becomes true or false.

These are some of the concepts that every programmer need to be strong with. Loops and conditions appear to be easy, but you should know how and when to use them effectively.

When it comes to problems, there are various kinds of problems that we deal with in computer science coding. Some can be solved, some are hard and some cannot be. There are competitive coding problems, application oriented, Real world scenarios (Commonly in IoT) and much more. Before coding a solution to the problem, there are some steps that are need to be followed.

Read the problem carefully. Identify its type. Work down the solution. If possible, use mathematical concepts to obtain optimal solution.

Computer Science is nothing without math. So get yourself strong in basic mathematical concepts like number theory.

When you solve a problem, there are various approaches to deal with it. There is brute force, greedy, dynamic programming, divide and conquer. Each one has its own advantage and disadvantage. When you try a problem first, use brute force. Then try to obtain a more optimal solution.

Gaze yourself through the basics of all these. Below is the list of best websites to refer:

• Geeks for geeks for algorithms

• Tutorials point for coding

Let us deal with some problems. Try to understand the problem, work it out with pen and paper and then code it. After the session, each of you will be receiving a coding question that you need to solve.

Coding in python is really easy. The syntax is so flexible that you would get attached to it in no time. But if you are not strong in the basics, it is of no use. So realize that basics are important. Get yourself strong with that and we will move on.

If you find solution to the problem, just ping us in the group. We will ask you to upload your code. C, C++, Java anything is fine. Except python. Once the code is uploaded, if its correct, we will discuss the code in the group. Then we will go for another problem.

**Examples:**

These are some of the questions taken from contests. Try answering them.

1) After the lessons *n* groups of schoolchildren went outside and decided to visit Polycarpus to celebrate his birthday. We know that the *i*-th group consists of *si* friends (1 ≤ *si* ≤ 4), and they want to go to Polycarpus together. They decided to get there by taxi. Each car can carry at most four passengers. What minimum number of cars will the children need if all members of each group should ride in the same taxi (but one taxi can take more than one group)?

Input

The first line contains integer *n* (1 ≤ *n* ≤ 105) — the number of groups of schoolchildren. The second line contains a sequence of integers *s*1, *s*2, ..., *sn* (1 ≤ *si* ≤ 4). The integers are separated by a space, *si* is the number of children in the *i*-th group.

Sample : 5

1 2 4 3 3

Output

Print the single number — the minimum number of taxis necessary to drive all children to Polycarpus.

Sample : 4

2) Caps lock is a computer keyboard key. Pressing it sets an input mode in which typed letters are capital by default. If it is pressed by accident, it leads to accidents like the one we had in the first passage.

Let's consider that a word has been typed with the Caps lock key accidentally switched on, if:

• either it only contains uppercase letters;

• or all letters except for the first one are uppercase.

In this case we should automatically change the case of all letters. For example, the case of the letters that form words "hELLO", "HTTP" should be changed.

Write a program that applies the rule mentioned above. If the rule cannot be applied, the program should leave the word unchanged.

Input

The first line of the input data contains a word consisting of uppercase and lowercase Latin letters. The word's length is from 1 to 100 characters, inclusive.

Sample : cAPS

Output

Print the result of the given word's processing.

Sample: Caps

3) Kirito is stuck on a level of the MMORPG he is playing now. To move on in the game, he's got to defeat all *n* dragons that live on this level. Kirito and the dragons have *strength*, which is represented by an integer. In the duel between two opponents the duel's outcome is determined by their strength. Initially, Kirito's strength equals *s*.

If Kirito starts duelling with the *i*-th (1 ≤ *i* ≤ *n*) dragon and Kirito's strength is not greater than the dragon's strength *xi*, then Kirito loses the duel and dies. But if Kirito's strength is greater than the dragon's strength, then he defeats the dragon and gets a bonus strength increase by *yi*.

Kirito can fight the dragons in any order. Determine whether he can move on to the next level of the game, that is, defeat all dragons without a single loss.

Input

The first line contains two space-separated integers *s* and *n* (1 ≤ *s* ≤ 104, 1 ≤ *n* ≤ 103). Then *n* lines follow: the *i*-th line contains space-separated integers *xi* and *yi* (1 ≤ *xi* ≤ 104, 0 ≤ *yi* ≤ 104) — the *i*-th dragon's strength and the bonus for defeating it.

Sample : 2 2

1 99

100 0

Output

On a single line print "YES"), if Kirito can move on to the next level and print "NO" if he can't.

Sample : YES

4) A country has *n* cities. Initially, there is no road in the country. One day, the king decides to construct some roads connecting pairs of cities. Roads can be traversed either way. He wants those roads to be constructed in such a way that it is possible to go from each city to any other city by traversing at most two roads. You are also given *m* pairs of cities — roads cannot be constructed between these pairs of cities.

Your task is to construct the minimum number of roads that still satisfy the above conditions. The constraints will guarantee that this is always possible.

Input

The first line consists of two integers *n* and *m* (1 ≤ *n*≤ 103, 0≤ *m* ≤ n/2).

Then *m* lines follow, each consisting of two integers *ai* and *bi* (1 ≤ *ai*, *bi* ≤ *n*, *ai* ≠ *bi*), which means that it is not possible to construct a road connecting cities *ai* and *bi*. Consider the cities are numbered from 1 to *n*.

It is guaranteed that every pair of cities will appear at most once in the input.

Sample : 4 1

1 3

Output

You should print an integer *s*: the minimum number of roads that should be constructed, in the first line. Then *s* lines should follow, each consisting of two integers *ai* and *bi* (1 ≤ *ai*, *bi* ≤ *n*, *ai* ≠ *bi*), which means that a road should be constructed between cities *ai* and *bi*.

If there are several solutions, you may print any of them.

Sample : 3

1 2

4 2

2 3

If you are done with these, try some more problems. These are some best websites for coding.

• Top coder

• Hackerank

Good luck!

Have fun coding!