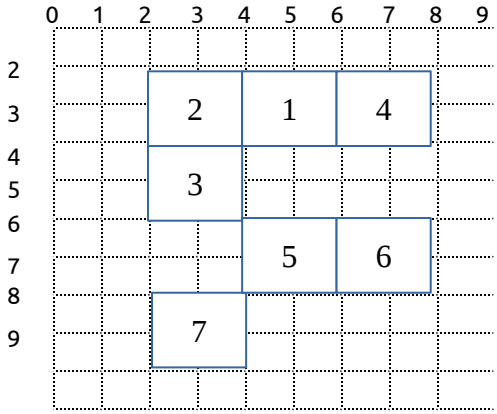
Padosan

A Square is a closed geometric figure with 4 equal sides, and its interior angles all right angles (90°). From this it follows that the opposite sides are parallel.
Consider a grid comprising several identical squares. Squares sharing one of its sides with another are **adjacent squares (Padosan)**. Overlapping squares and squares sharing a single vertex point are not considered adjacent.

In the following figure, square 1 is adjacent to squares 2 & 4, square 2 is adjacent to squares 1 & 3, square 3 is adjacent to square 2, square 4 is adjacent to square 1, square 5 is adjacent to square 6, square 6 is adjacent to square 5 and square 7 is isolated (not adjacent to any of the other squares).



Write a program to determine the number of adjacent squares for a given square.

Input Specification:

1) The first line of the input will contain integer **N** where **N** is number of squares (1<=N<=50).

2) The next **N** lines will contain 8 positive integers in each line, each pair of the integers represents the (x, y) coordinates of one of the vertices of **N**th square.

Output specification:

On each line print the square number and the number of the squares adjacent to it, separated by a space, for each square starting from square no. 1 to square no. **N** terminated by new line character.

Sample 1		**Sample 2**	
Sample Input - 1	**Sample Output - 1**	**Sample Input – 2**	**Sample Output - 2**
7 1 1 3 1 3 3 1 3 3 1 5 1 5 3 3 3 5 1 7 1 7 3 5 3 1 3 3 3 3 5 1 5 1 7 3 7 3 9 1 9 3 7 5 7 5 9 3 9 5 4 7 4 7 6 5 6	1 2 2 2 3 1 4 1 5 1 6 1 7 0	4 1 1 3 1 3 3 1 3 3 1 5 1 5 3 3 3 1 3 3 3 3 5 1 5 3 3 5 3 5 5 3 5	1 2 2 2 3 2 4 2

ShabdKhosh

You have to write a program to chain some words. A word is properly chained if it starts with a trailing sub-string of its predecessor word with a minimum overlap of three (3) characters. Given a number of words, you have to reorder them to appropriately chain them. The first word in the input is used as a starting word in the chain. It may happen that there is no chaining possible for a given set of words. If chaining is possible, assume that there will be
a unique word chain.
Note: A word is a sequence of alphabetic characters.

Input Specification

- The first line will be an integer N, indicating the number of words that will follow. Assume N will never be greater than twenty (20)
- The next N lines of input will contain words, which are to be chained. Assume that the maximum length of a word will never exceed thirty (30) characters.

Output Specification

- Your program should output the chain of words, one word on a separate line. If there is no chain possible from the given words, the program should print IMPOSSIBLE

Sample 1		**Sample 2**	
Input 1	***Output 1***	***Input 2***	***Output 2***
2 start finish	IMPOSSIBLE	8 whisper format perform sonnet person shopper workshop network	whisper person sonnet network workshop shopper perform format

Happy Numbers

Let the sum of the squares of the digits of a positive integer s_0 be represented by s_1. In a similar way, let the sum of the squares of the digits of s_1 be represented by s_2, and so on. If s_i=1 for some 1≤ i, then the original integer s_0 is said to be **happy**. For example, starting with 7 gives the sequence **7**, **49** (=7^2), **97** (=4^2+9^2), **130** (=9^2+7^2), **10** (=1^2+3^2), **1** (=1^2), so 7 is a happy number, which reaches 1 on 6 iterations.

The first few happy numbers are 1, 7, 10, 13, 19, 23, 28, 31, 32, 44, 49, 68, 70, 79, 82, 86, 91, 94, 97, 100, …. The number of iterations i required for these to reach 1 are, respectively, 1, 6, 2, 3, 5, 4, 4, 3, 4, 5, 5, 3, ... A number that is not happy is called **unhappy**. Once it is known whether a number is happy (unhappy), then any number in the sequence s_1, s_2, s_3, ... will also be happy (unhappy). Unhappy numbers have eventually periodic sequences of s_i which do not reach 1 (e.g., 4, 16, 37, 58, 89, 145, 42, 20, 4, ...). You need to write a program to find all the happy numbers in a given closed interval, which reach 1 within 10 iterations.

Input Specification

- It is a single line input of two positive integers separated by a space.

Output Specification

- Print all happy numbers in the interval and the number of iterations required by it to reach 1, separated by a space and each in a new line.

Sample Input 7 11 **Sample Output** 7 6 10 2	**Sample Input** 44 68 **Sample Output** 44 5 49 5 68 3