Queen Mary
University of London

9th QMUL S&E Programming Competition

Run by

School of Electronic Engineering and Computer Science

Sponsored by

Wednesday, 26th January 2016

Note. *Your programs should read from standard input. The programs should process one input and print the result at the standard output. When judging your submission we will run your program several times on different inputs.*

Death Star DoS Attack

Princess Lea is a seasoned hacker and is preparing the following attack on the Death Star. She has a large collection of webpages that she has managed to compromise in the last weeks. Her plan is to replace all hyperlinks in those pages with the address of the Death Star's welcome page. As web users start pressing those hyperlinks, the unexpected traffic will bring down the Death Star's webpage.

As a first task, she needs to write a program that, given a bit of hypertext, it extracts all the links from the text.

Input Specification

Each input consists of:

- The contents of the body of an html document. The input might have up to 150 KB of text, so make sure your program is efficient and can deal with such input in within 5 seconds.

Output Specification

Print a list of all the links in the given html text in lexicographical order. If the html file contains no links you should instead print "No links found". If a link appears more than once you should only print it once. You should only consider links that use double quotes and follow the format, with no extra attributes other than href:

```
<a href="link">
```

Examples.

Input	Output

`<h1>Hellow World</h1>`
`If you are having fun`
`click <a href="http://www.link1.com">here</a>.`
`Otherwise`
`click <a href="http://www.link2.com">here</a>.`
	`http://www.link1.com`
	`http://www.link2.com`

`<h1>Hellow World</h1>`
`If you are having fun`
`click <a href="http://www.link2.com">here</a>.`
`Otherwise`
`click <a href="http://www.link1.uk">here</a>.`
`Or again <a href="http://www.link1.com">here</a>.`
`Same text again:`
`If you are having fun`
`click <a href="http://www.link1.com">here</a>.`
`Otherwise`
`click <a href="http://www.link2.com">here</a>.`
	`http://www.link1.com`
	`http://www.link1.uk`
	`http://www.link2.com`

Breaking Death Star Encryption

Luke Skywalker and his companions are preparing their attack to Darth Vader's Death Star. The success of their undertaking critically relies on their ability to break the Death Star's encryption used in all their opponents' communications. Han Solo has managed to steal the (rather naive) encyption prototocol:

1. Each day a secret codeword (a single byte) is produced and used as a simple encoding "mask" for all text.
2. Plain text is transformed into binary.
3. The codeword is repeatedly XOR-ed with each byte of the binary representation of the plain text.

Luke can use the Force to interfere with Death Star's communication officers' mind to obtain the secret codeword: on the day of the attack, the code is "01101110".

Write a program to decrypt encypted messages!

Input Specification

Each input consists of:

- An encrypted message, given as a binary string whose length is a multiple of 8. The encrypted message will be given in blocks of 40 binary bits on each line, except for the last line which might have less than 40 bits.
- The secret codeword for the day of the attack is "01101110".

Output Specification

The decoded message in plain text.

Procedure

You should first repeatedly XOR the binary codeword on the binary encrypted text, byte by byte, to retrieve the decoded text in binary. Then, using standard ASCII conversion, convert the message to ASCII characters.

Input	Output
0010111100011010000110100000111100001101 0000010101001110000011110001101001001110 0000101000001111000110010000000001000000	Attack at dawn.
0100110100111010000000010001111001001110 0011111010000101100001101000111000000101 1 0001101001001101010011100011101000000110 0000101110100111000011101000110100000111 1 0001101000001011010011100000000100001000 0100111000001111000010000000100000001111 0000011100011100000111010110010000000001 0000000001001110000110100000011000001011 0100111000101010000010110000111100011010 0000011001001110001111010001101000001111 0001110001001110000001110001110101001110 0001110100011011000011000000000100011110 0001101000000111000000110000111100000010 01000000	#Top Secret# The state of affairs on the Death Star is suboptimal.

Darth Vader Guessing Game

Lord Vader does not have many friends to play with. He has therefore invented the following game to play with his computer. The computer generates two numbers (*i*,*m*), each of which has at most 3 digits. Then, Vader has to come up with the *i*-th prime number that is greater than *m*.

Write a program for winning the game!

Input Specification

Each input consists of a pair (*i*,*m*) with each number between 1 to 999.

Output Specification

A prime number *p* such that there are exactly *i-1* prime numbers between *m* and *p*.

Examples.

Input	Output
(2,11)	17
[illegible]	[illegible]

Average Marks

Darth Vader was entering the level of ability for each of his staff in an online system. When he finished he noticed that his return key was not working at all. He ended up getting all grades in a single line without any space between them. Each grade is an integer between 1 and 10. All grades were entered written in base 10, without leading zeros. For example, if the grades were 3, 10, 1 and 10 they would be entered as 310110.

Input Specification

Each input consists of:

- a single line that contains a non-empty string S of at most 100 base 10 digits. There is a unique way to partition S into a list of substrings such that each substring represents an integer between 1 and 10 in base 10 without leading zeros.

Output Specification

Output a line with a decimal number representing the average of the grades given. The result must be output as a decimal number with exactly two digits after the decimal point, with remaining digits truncated.

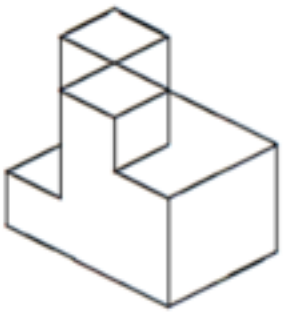
Examples.

Input	Output
310110	6.00
124	2.33

Princess Leia's Tower

Princess Leia is planning to build a new fortress to protect her kingdom. She asked an architect to come up with a sketch of how the different towers will look like. The architect gave her the sketch in the form of a heigh map, a two-dimensional matrix of integers indicating the height of the fortress at each of coordinate, e.g.

3	2	2
1	3	2



The princess was imediatelly concerned about how much she would spend in painting all the walls of the fortress. So she hired you to calculate the total surface area of all the walls of the fortress given a proposed plan.

Input Specification

Each input consists of:

- Two numbers n and m indicating the dimensions of the height map.
- n lines will follow containing m integers in each line.

Output Specification

You program should output the total surface area of all the walls in the proposed sketch of the tower.

Examples.

Input	Output
2 3 3 2 2 1 3 2	28
3 3 3 3 3 3 3 3 3 3 3	36

Death Star Room Allocation Problem

After a major refurbishment took place on the Death Star, Lord Vader is allocating rooms to all of his staff. In order to avoid conflicts, he has asked each member of staff to submit a list of persons they don't like, so as to make sure they are not assigned the same room. As the number of rooms of the spacecraft is limited, such an 'optimal' allocation may not be possible.

You should write a program that determines whether it's possible to split the members of staff into rooms so that they don't share a room with someone they don't like.

Input Specification

Each input consists of:

- The first line of input contains a number 0 < M < 8, which is the number of all available rooms.
- The second line of input contains a number 0 < N < 8, which is the number of members of staff. The i-th member of staff is identified by the id i.
- The following N lines will contain sequences of numbers represending the id's of those people the member of staff doesn't like. If the i-th member of staff likes everybody then the i-th list of sequences will be empty (an empty line).

Output Specification

YES, or NO, according to whether such an optimal allocation is possible.

Examples.

Input	Output
3 7 2 5 7 3 4 5 6 2 3 4 2	YES
3 7 2 5 7 3 4 5 6 2 3 4 2 5	NO