This coding test has 3 sections and 3 problems.

**Section 1: OOP and Software Design**

For this test:

* Those who you will use C#, you may create a Windows console application to show the output.
* Those who you will use C/C++, you may create a DOS-mode/Windows-console application to show the output.
* Those who you will use TypeScript or any JavaScript based language, you may show the output on browser (if any).

Purpose of this section is to test how good you are with the knowledge of OOP and how well you can organize (i.e. design) your code. Make sure you use all OOP concepts such as, interfaces, abstract classes, method overloading, overriding, etc. whenever applicable. If you can apply a few of the design patterns you know of, it will be even more excellent!

**Problem:**

When an HTML page is rendered on a browser, the browser actually parses the full HTML of that page and creates an object representation of the HTML code - which we know is called DOM (document object model).

A DOM is nothing but a tree of node objects. Each node can either be a leaf object or a collection of more nodes.

Consider this HTML snippet below:

<div>

<span>Heading</span>

<p>Lorem ipsum dolor ismet…</p>

<ul>

<li>one</li>

<li>two</li>

</ul>

</div>

The <span> in the second line is a leaf node, whereas the <ul> is a container node. But all of them are nodes nonetheless.

Container nodes have some special operations or methods:

AddNode(*node*) - this method adds *node* inside the container along with its other children, if any.

RemoveNode(*node*) - this method removes *node* from the container.

Every (both container and leaf) node has some common operations or methods:

GetInnerHTML() - this method simply returns the HTML version of all nodes inside a container element. For leaf nodes, this will return empty string. So for the <div> element in the above code snippet, this method will return:

<span>Heading</span>

<p>Lorem ipsum dolor ismet…</p>

<ul>

<li>one</li>

<li>two</li>

</ul>

GetHTML() - this method returns the HTML version of all nodes inside a container element, along with its own HTML markup. So for the <ul> element in the above code snippet, this method will return:

<ul>

<li>one</li>

<li>two</li>

</ul>

Your task is to write a program with necessary classes that will allow you to build an HTML tree with any tag and then print the output of the HTML.

Please note that you won’t have to build a tree from an actual HTML string, therefore you won’t have to parse HTML. You will build the tree in code in your main() function by adding/removing nodes, and print output of the complete tree there.

**Section 2: Basic Javascript**

Purpose of this section is to test how good you are with the knowledge of frontend technologies, such as HTML, CSS and specially Javascript. Feel free to use any Javascript libraries you want (such as jQuery) or you can use raw Javascript as well. But the problems below should be solved from scratch, without using any ready-made solutions found on the Internet.

**Problem:**

Write necessary Javascript code that shows the strength of a password as the user types it into a password field:

<input id="myPassword" type="password" name="" value="">

Strength levels:

Weak - less than 5 alphanumeric characters

Medium - 5 to 10 alphanumeric characters

Strong - 5 to 10 alphanumeric characters, plus any other non-alphanumeric character

Very Strong - more than 10 characters, plus 2 or more non-alphanumeric characters

You have to output the strength just beneath the password field, in this format:

Password Strength: Weak

**Section 3: Programming Logic**

For this test:

* Those who you will use C#, you may create a Windows console application to show the output.
* Those who you will use C/C++, you may create a DOS-mode/Windows-console application to show the output.
* Those who you will use TypeScript or any JavaScript based language, you may show the output on browser (if any).

Purpose of this section is to test your proficiency in general programming and logic. For these problems, you don’t have to know any specific algorithms (tree, graph, BFS, DFS, etc), but having basic common-sense is very much necessary. Also, you won’t have to follow OOP for these problems, if you don’t want to. Solving problems is the only thing necessary here, but solving in a performance-optimized way will be valued and appreciated.

**Problem:**

Your computer has some files. For this problem, it is given that all filenames are of equal length.

You have created a program for filename pattern matching. When you type a command like "dir PATTERN", the program shows all the filenames that match the specified pattern.

A pattern can contain letters ('a'-'z'), '.' character, and '?' character only, and nothing else.

Each '?' matches any single character (including '.'), and all other characters match themselves only. For example, the pattern "conte?t.info" matches the filenames "contest.info" and "content.info", but not "contemn.info" or "contests.nfo".

**Input**

Every line will contain a list of filenames, each separated by a space character.

Input Constraints:

- There can be 1- 50 filenames at max.

- Each filename will contain 1 - 50 characters.

- All filenames will be of equal length.

- Each filename will contain only lowercase letters ('a'-'z') and '.' character.

**Output**

Print the pattern that will match all the filenames while containing as few '?' characters as possible.

**Sample Input**

contest.txt context.txt

config.sys config.inf configures

**Sample Output**

conte?t.txt

config????