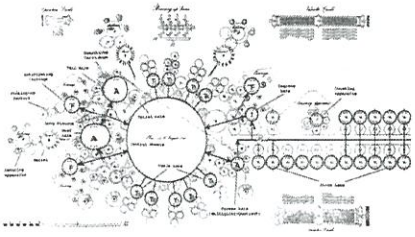
AP Computer Science Name Solution Date_____ Period____



APCS Practice Problem: Web Site processor

There are many web pages formatted in html, the hyper text markup language. HTML contains embedded tags – formatting instructions enclosed in angular brackets < and >. The formatting tags often come in pairs where the opening tag indicates the beginning of some formatting (e.g., italics, bold, underline) and the closing tag indicates the end of the formatting. The closing tag contains the same **keyword** or instruction as the opening tag, but preceded by a forward slash (the "/" character). The following example shows a line of HTML text and illustrates the way it might be shown on screen:

The quick brown fox jumps<u> over the lazy</u> dog

The *quick* **brown** fox **jumps** **over** the lazy dog

Processing HTML text may involve such tasks as removing HTML tags from the text or verifying that opening and closing tags come in matching pairs.

In this question, we deal with text represented as a String object. We assume that this HTML text contains only complete tags; all "<" and ">" properly delimit tags and do not otherwise occur inside tags or anywhere else in the text. The segments of text formatted with different tags **may overlap**. In the above example, the word **"over"** falls into the overlapping bold and underscored segments. You will write a few methods of the HTMLProcessor class for processing HTML text.

(a) Write the findFirstTag method of the HTMLProcessor class as started below. The method finds and returns the first tag in a given HTML text strings. The method returns null if no tags were found.

```
//precondition:      text is a segment of HTML text which may contain complete
HTML tags;  a tag is any substring starting with < and ending with the closest >

//postcondition:     returns the first HTML tag found in text (including the <
and > brackets) or null if not tags found.
public static String findFirstTag(String text)
{
    int where = text.indexOf("<");
    if (where == -1)    // no tags
        return null;
    int end = text.indexOf(">", where);
    return text.substring(where, end+1);
}
```

(b) Write a method remove for the HTMLProcessor class, as started below. The method finds the first occurrence of a given substring in a given String text and returns text with that substring removed. If the given substring is not found, the method returns text unchanged.

```
//precondition:      str is a non empty String

//postcondition:     if str is found in text, its first occurrence is removed
from text and the new string is returned;  otherwise the original string text is
returned.
```

String str,

```
public static String remove(String text){
```

```
// Look for str.
int where = text.indexOf(str);
if(where == -1)
    return text;
else{
    text = text.replace(str, "");
    return text;
}
```

```
}
```

(c) Write a method **removeAllTags** for the HTMLProcessor class. The method deals only with a subset of HTML tags: it assumes that a given text contains only complete simple tags, such as <u> and </u> or <cite> and </cite>, where a closing tag differs from the corresponding opening tag only by the "/" character after the "<". The closing tag must come after the opening tag. The method returns the text with all tags removed(or the original text if no tags were found). The method should throw an IllegalArgumentException if the tags in text do not match (no closing tag found after an opening tag). You can assume that the methods remove and findFirstTag work as specified, regardless of what you wrote in parts (a) and (b).
Write the method removeAllTags as started below;

//precondition: text is a segment of HTML text which may contain complete HTML tags; a tag is any substring starting with < and ending with the closest >

//postcondition: if all HTML tags in text come in matching opening-closing pairs, then all the tags are removed from text and the new text string is returned; if text has no tags, the original string is returned; throws an IllegalArgumentException if the tags do not match.

```
public static String removeAllTags(String text){
    String temp = "";
    while(true){
        // look for 1st <
        int where = text.indexOf("<");
        if(where == -1)
            return text;
        int end=text.indexOf(">",
                where);
        if(end == -1)
            throw new IllegalArgument
                Exception("unmatched<>");
        text = remove(text, substring
                (start, end+1), text);
    }
}
```