Homework 3, CSCE 240, Summer 2016

Your assignment for Homework 3 is to use a stack to parse an input XML file, check the XML for validity, and to use the data to create data payload instances from the input XML.

The driver program is given in the code on the website. You must implement a class `XMLParser` class that includes the methods invoked by the driver.

- First read the DTD (Document Type Definition) file to create an `set` of the legal XML tags for this file. This will be done in the `XMLParser` class.
- Read the XML line by line, creating as you go an instance of `XMLItem` from the input line, and pushing those instances onto a `vector<XMLItem>` structure for later processing.
- Process the `vector` of `XMLItem`s
  - If the instance is an open tag, verify that it is legal in the DTD, and then push it onto the stack.
  - If the instance not a tag, it is data, and push it onto the stack.
  - If the token is a close tag, verify it is legal in the DTD, and then pop the stack until you get the first open token and compare the token against the close to ensure that they match.
- If a tag is not legal (that is, not in the DTD), then flag this with an error message and terminate the program.
- If the stack popping to verify proper nesting shows the nesting to be incorrect, then flag this with an error message and terminate the program.

Rules

- You may assume that each line is a single item, that is, either an open tag, or a close tag, or just data.

- You may assume that inside the brackets (and possible slash) there is only the tag, that is, there are no “attributes” that might usually be associated with XML.
- You may assume that if there are open and close brackets, the item is a tag, that is, that there is no “data” that starts and ends with open and close brackets.

Your `XMLItem` class should have a constructor that takes in a line of data, parses that for brackets (and slash?), and assigns the value of an enumerated data type to say whether the item is an open tag, a close tag, or data.

You probably ought to include a “bad” category in your enumerated type. This might not be useful now, but if you assign the bad category as the default, then you will be able to tell if you have in fact processed the input line at all.