# CSC 383, Sections 401 and 410
# Fall, 2011
# Assignment 2
# Due 11:59pm CT, Wednesday, September 14th

**Requirements**. You are to write a program for parsing an XML file. An XML contains XML tags and text. The tags are used to mark up the structure of the data represented in the file.

An **XML tag** is always bounded by angle brackets, e.g., `<para>`. There are three types of tags:

- If the second character is a forward slash (e.g., `</para>`), this a **closing tag**.

- If the second to last character is a forward slash (e.g., `<br />`), this is a **self-closing tag**.

- If a tag is neither of the above, it is an **opening tag**.

- The **tag name** is the text inside that does not include the slashes.

In this assignment we will deal with very simple tags. Except for the slashes mentioned above, the only characters that may appear between the brackets are letters, numerals, and hyphens. A space may only appear before the slash of a self-closing tag.

A properly formed XML document is such that:

- Every opening tag is followed at some point by a closing tag with the same tag name.

- Opening and closing tags are properly nested. For example, if the following tags appear in the document:

  ```
  <chapter>
    <para>
  ```

  They must eventually be followed first by a `</para>` tag and then a `</chapter>` tag.

- Self-closing tags may appear anywhere.

Between tags may be any text not containing a less-than character or a greater-than character.

Your program is to read an XML file from standard input and print to standard output an indented version of the file. While doing this it should check whether the file is properly structured by the rules mentioned above. If not, it should print one of two error messages, the number of the input line where the error was found, and exit.

There are two possible errors:

1. A closing tag is found that does not match the most recently seen opening tag.

2. The end of the file is reached but there are one or more opening tags with no closing tags.

For reading the input, I suggest using a `Scanner` object. You may assume that whitespace is present between tokens.

To print the XML file, each tag is to appear by itself on its own line. Text between one tag and the next will all appear on a single line. Also, tags are to be indented four spaces for each level of nesting.

You should write two files:

- A class definition for a token, where a token is either a tag or a word. Call the file `XMLToken.java`.

- A program for parsing and printing the input as described above. Call the program `ParseXML.java`.

I will provide test files, including some with incorrectly formed XML.

Submit a zip file called `ParseXML.zip` containing the two files mentioned above.

**Grading rubric**: This assignment is worth 50 points, with points assigned as follows:

- There are two source files as described above (5 points)
- Variable and method names descriptive and mnemonic (4 points)
- Comment block with information specified (4 points)
- Program prints the input properly nested and indented (20 points)
- Program catches and reports all three error types (15 points)
- Code is properly indented (2 points)

Beyond the above rubric, 2 points will be deducted for each missed requirement. If I say to do something a certain way, do it that way.