CSCI 3020 ASSIGNMENT 1 (50 POINTS)

DUE FRIDAY, SEPTEMBER 4, 11:59 PM

GOALS

This assignment has you convert a text file into XML in order to get a feel for XML and help you review C++.

GETTING HELP

As you may have noticed, we have not done a review of C++ in the videos/lecture. Since it has been a while for some, you may be a little rusty. That is perfectly ok, you just need to ask for help. Don't be stubborn, just do it.

- Your textbook from CSCI 1010/2010 covers all C++ that is necessary for this assignment.
- Post general questions on the discussion board in D2L. Please refrain from posting more than 1 line of code, otherwise your post may be removed.
- See the tutors in the lab.
- Send me email (nicholsonja@apsu.edu) or come to my office. I'll be happy to help. When emailing me, please do not send me a code snippet; send your entire .cpp so that I can compile and test it.

ASSIGNMENT

The two files **sample1.txt** and **sample2.txt** contain example records for a company in an old format. The records are a list of account managers and the accounts they manage for the company. The files are plain text files. The data in the files consist of

- Identification information for sales managers of company accounts
- The city and state the sales managers are based out of
- The list of company accounts they manage.

Your company is updating their systems and wants the file converted to an XML-based format. The old format is a record that spans multiple lines:

```
EMP_ID LASTNAME FIRSTNAME CITY STATE
COMPANY1
COMPANY2
...
--END_MANAGER_DATA--
```

For every record

- The first line is the manager's employee id, his/her name, and the city/state they operate out of. There are five fields separated by spaces.
- After the first line, there is a list of up to 15 company names with one name to a line. You know when the company list for a manager ends when you see a line that looks like this:

```
--END_MANAGER_DATA--
```

There is no particular number of employees records in the file. Your program should be able to handle an input data file of any length.

Your program needs to convert the old format to XML. It should

- Prompt the user for a text-based data file to read.
  - If there is a problem opening the file, inform the user with an appropriate error message and the program should end.
  - Your program should be able to accept any input file, not just files named **sample1.txt** and **sample2.txt**.
  - You can assume the input file given is in the correct format.
- If you can successfully open the input file, prompt the user for the XML that should be written to.
  - If the filename does not end in ".xml" you should add it to the filename.
  - If user's filename already ends in ".xml", you should not add anything to the filename.

Once you know you can read the input file and have the output filename, start converting the text format to the following XML format:

- Start with the XML prolog

```
<?xml version="1.0" encoding="UTF-8"?>
```

- The next line should be a comment with your name in the following format

```
<!-- Processed by John Nicholson's converter -->
```

  You should obviously replace my name with your name.

- The remaining lines should be tags and data
  - The root elements should be **accounts**
  - For every record in the data file, you should create an **account** element which has the following elements
    - A **manager** element that has an attribute **employeeId**, and two elements, the **firstName** and **lastName** for the account manager.
    - A **location** element that has two elements, the **state** and **city** elements for the city the account manager operates from.
    - A **companies** element that has an attribute **count** that indicates how many companies the account manager is responsible for. In the companies element there are 1 or more **company** elements, one for each company the account manager manages.

Your output XML document must be well formed. After you generate an XML output file, open it in jEdit and test it using the menu option **Plugins -> XML -> Parse as XML**. The file you generate should produce no errors.

See the **sample1.xml**, which was generated from sample1.txt, for an example of the required output format. You are not required to indent, but the line breaks should be similar.

ADDITIONAL C++ REQUIREMENTS

Additional requirements

1. You should not have one, ginormous main() function. You program should be modular, that is, define several (more than 2) meaningful functions. All functions should have a comment describing their purpose.
2. You may not use any global variables.
3. You may not use cstring, that is, do not do this:

   #include <cstring>

   You can us string, that is, this is ok:

   #include <string>
4. You may not use the **exit()** function.
5. Except for items #3 and #4 above, you may use any feature of C++ defined in the language or STL. Your CSCI 1010/2010 book covers all the necessary C++ needed for this assignment.
6. You may not use any non-standard C++ library or feature specific to your compiler. For example, Visual Studio users may not use the XML libraries provided by Microsoft.
7. You must include a comment at the top of your main.cpp file in this format:

```
/*******************
John Nicholson
CSCI 3020 Section 09
Fall 2015
Assignment 1

Programmed on Windows 7 using Visual C++ 2010 Express

This program converts ...
*******************/
```

   If this comment is missing, you will receive 0 points on the assignment. You should obviously change the name to your name, the section to your section, the system to the system you used, and write a real description of the program. You must include the specific operating system and the compiler/IDE you used.

   Please note that **Windows** is not an operating system; **Windows 7**, **Windows 8**, and **Windows 8.1** are operating systems. **Visual Studio** is not a compiler/IDE. **Visual C++ 2010 Express** and **Visual Studio Ultimate 2013** are compiler/IDEs. You should be specific about which OS and compiler/IDE you used, otherwise you may lose points in grading because I graded using the wrong compiler.

   NOTE: You assignment will not be regraded if you do not put in the correct comment or are not specific enough and, as a result, your code is tested with the wrong compiler causing your code to crash.

Suppose the user wants to read from **data.txt** and write to **data.xml**. If data.txt contained these 2 records:

```
45912 BARNICLE GAIL BRIGHTON TN
CONOCO INC
UST INC
LANDS END INC
--END_MANAGER_DATA--
19283 ANDERSON ANGELINE NORRIS TN
LEVEL 3 COMMUNICATIONS INC
ADC TELECOMMUNICATIONS
--END_MANAGER_DATA--
```

The output in data.xml would be:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Processed by John Nicholson's converter -->
<accounts>
    <account>
        <manager employeeId="45912">
            <lastName>BARNICLE</lastName>
            <firstName>GAIL</firstName>
        </manager>
        <location>
            <city>BRIGHTON</city>
            <state>TN</state>
        </location>
        <companies count="3">
            <company>CONOCO INC</company>
            <company>UST INC</company>
            <company>LANDS END INC</company>
        </companies>
    </account>
    <account>
        <manager employeeId="19283">
            <lastName>ANDERSON</lastName>
            <firstName>ANGELINE</firstName>
        </manager>
        <location>
            <city>NORRIS</city>
            <state>TN</state>
        </location>
        <companies count="2">
            <company>LEVEL 3 COMMUNICATIONS INC</company>
            <company>ADC TELECOMMUNICATIONS</company>
        </companies>
    </account>
</accounts>
```

SUBMITTING

Upload the following three files to the Assignment 1 Dropbox in D2L:

- Your .cpp file
- The XML file generated from processing sample1.txt
- The XML file generated from processing sample2.txt

Do not upload your entire Visual Studio project (or project of whatever you are using). Just upload the files listed above.