C++ Project Step 1

Please read the project description and refer to it while reading this.

Our first step will be to create an XML parser for the dungeon files. You will need to be able to read in all of the files in sample.xml and samplepack.xml.zip in the Overview.zip directory.

We will soon have an example parser that parses a different XML file than we have provided. This should help you in writing your parser. Look at the file DungeonExplained.pdf in the Overview.zip file to see an explanation of what the different elements in the XML file mean.

In addition to parsing the file without crashing, you should create classes for the different elements of a game dungeon. As you parse the XML file, you will call the corresponding functions to create new instances of objects or to add elements to objects.

As an example, looking at the sample.xml file, we'll need

1. A room object for a room that will contain methods to add the room name, the description, an item name (e.g., "torch" in the first room), a container name (e.g., "chest"), a creature name, a trigger object or pointer to a trigger object, a border object or pointer to a border object;
2. A trigger object that contains methods to add a type, a command name, a condition object or pointer to a condition object, and a string to print.
3. A condition object that contains a *has* field, an object name (e.g., a *torch* for the trigger in the first room), as status for an object, and an owner of the possessed object, an
4. Other objects and fields corresponding to the different dungeon items.

You should print out the parsing steps as your output. A relatively easy way to do this is to have each constructor for the objects created to print out the object created and what it is, and for each of he functions corresponding to adding information to an object (e.g., an item name) print out that information and the function being called.

What you should turn in:

Your code, turned into the code part of Step1.

A video showing your parser in action on sample.xml and 2 other files of your choosing from samplepack.xml that are different from sample.xml. The video should show you downloading your code from Brightspace, building it, and running your program on the xml files. You should show the output from the runs in your video. This video, not surprisingly, is turned into the video part of Step1. Try to keep videos brief as the graders will have approximately 250 projects to grade. This will be easier to do if you practice downloading and building your project and videoing a run before you do the actual video.