CPSC 327

Project 4

References:

1. Building and linking to a Static Library lectures and projects
2. Pointers Memory lectures and projects
3. Classes, Objects lectures and projects

Sample Code:
See 2 starter projects on course website projects folder

Topics covered by this project;

- Creating and using a static library
- Using pointers to manipulate objects
- Using vectors to hold objects and pointers
- Class Heiarchies
- Abstract Base Classes
- Polymorphism

Class Heiarchy
You are developing a class heiarchy for this project. An Abstract Base Class (ABC), ‘Smalltalk’ defines the heiarchy behaviour.
Classes derived from Smalltalk must implement populatePhrases(). A function that initializes the baseclass vector with phrases that are unique to that class type. For instance, Smalltalk_American will populate the vector with the american phrases found in constants.h.

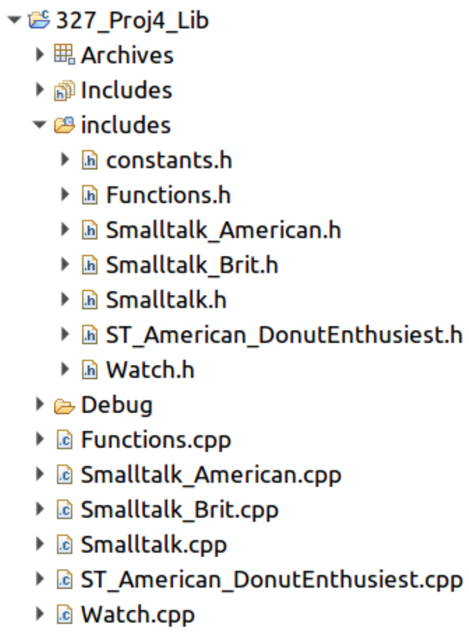
Additionally you are given a complete watch object. You may give or take a watch from any instance of Smalltalk_American, ST_American_DonutEnthusiest or Smalltalk_Brit. Note that watches cannot be created out of thin air, if you give one to an instance you no longer have that watch, the instance does. Because of that fact, this is one case where a shallow pointer copy is appropriate. See Smalltalk.h for further guidance.

Please also provide a function (as specifies in Functions.h and outlined in Functions.cpp) that generates a vector of unique pointers. Please pay attention to the hints I've left you in the implementation.

Please compile both projects using the C++11 language standard.

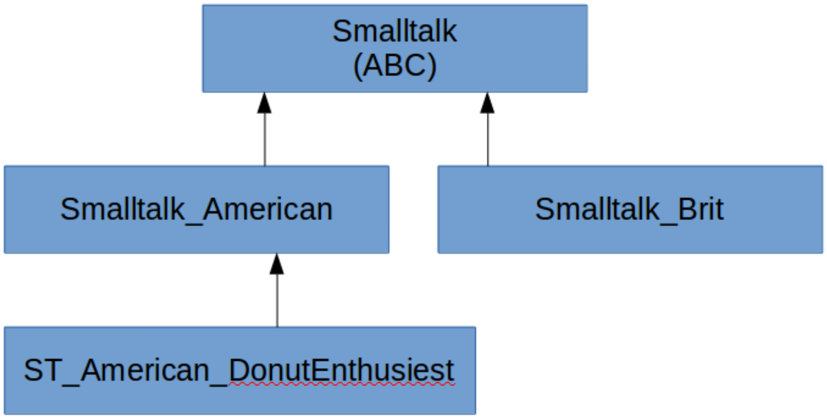
Library

I would like you to develop a static library with the following name and file structure.



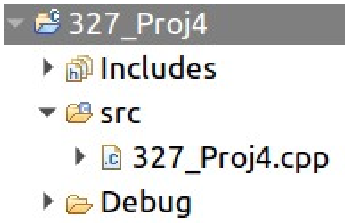
```
327_Proj4_Lib
  Archives
  Includes
  includes
    constants.h
    Functions.h
    Smalltalk_American.h
    Smalltalk_Brit.h
    Smalltalk.h
    ST_American_DonutEnthusiest.h
    Watch.h
  Debug
  Functions.cpp
  Smalltalk_American.cpp
  Smalltalk_Brit.cpp
  Smalltalk.cpp
  ST_American_DonutEnthusiest.cpp
  Watch.cpp
```

All classes inherit publicly. The class hierarchy is as follows;



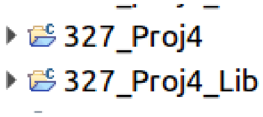
I have given you the header files and some of the implementation.

Testing
Please develop a test application that has the following name and file structure:



```
327_Proj4
    Includes
    src
        327_Proj4.cpp
    Debug
```

This application should link statically to the above library. The projects as they appear in the eclipse workspace.



```
327_Proj4
327_Proj4_Lib
```

Please be sure to test your library throughly. I will use my own test harness.

To Turn In
327_Proj4.cpp
Functions.cpp
Smalltalk_American.cpp
Smalltalk_Brit.cpp
Smalltalk.cpp
ST_American_DonutEnthusiest.cpp

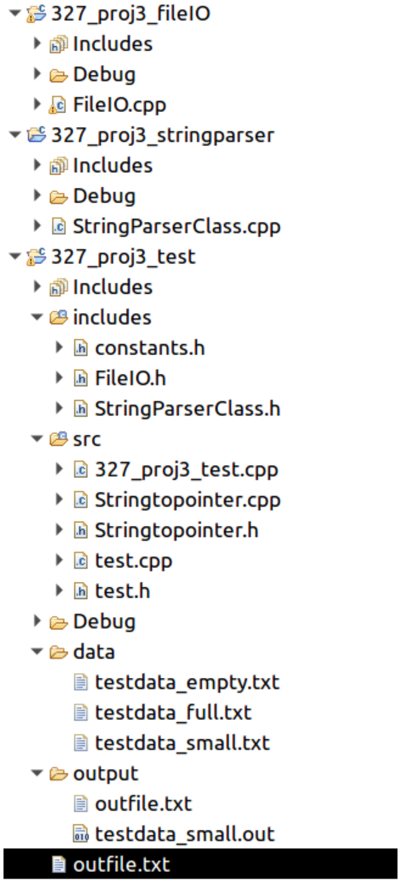
The string parser is so versatile that it should be packaged in a library so that it can be easily integrated into other applications. This approach has the following advantages;

1. Easy to integrate into multiple projects.
2. Easy to maintain. Any changes made to the parser library are made in a single codebase regardless of the number of applications using it. (There are not multiple copies of the parser source code embedded in client applications, each requiring individual attention when integrating parser changes.)
3. It makes it easier to divvy up projects in a team. The library is defined by its interface, in this case a header file that describes the functionality it provides. The header file is a contract describing library functions, how they are accessed, what is returned, and what is expected. It completely defines all communication possibilities between the library and its clients.

For these 3 reasons alone (there are more) much of the code produced in the world is provided as a library.

What I gave you

On the course website, projects section, you will find 3 incomplete projects. The following shows the solution as it appears in Eclipse. The includes directory should be in a top level stand alone folder, but we are running up against a limitation of eclipses' 'Workspace' concept. So it's located in the 327_proj3_test folder. Note that FileIO.h and StringParserClass.h as well as constants.h are located there.



```
327_proj3_fileIO
    Includes
    Debug
    FileIO.cpp
327_proj3_stringparser
    Includes
    Debug
    StringParserClass.cpp
327_proj3_test
    Includes
    includes
        constants.h
        FileIO.h
        StringParserClass.h
    src
        327_proj3_test.cpp
        Stringtopointer.cpp
        Stringtopointer.h
        test.cpp
        test.h
    Debug
    data
        testdata_empty.txt
        testdata_full.txt
        testdata_small.txt
    output
        outfile.txt
        testdata_small.out
    outfile.txt
```

Assignment

This project links 327_proj3_fileIO Library correctly to the 327_proj3_Test application. Please implement the 327_proj3_StringParser library and link it to the 327_proj3_Test application.

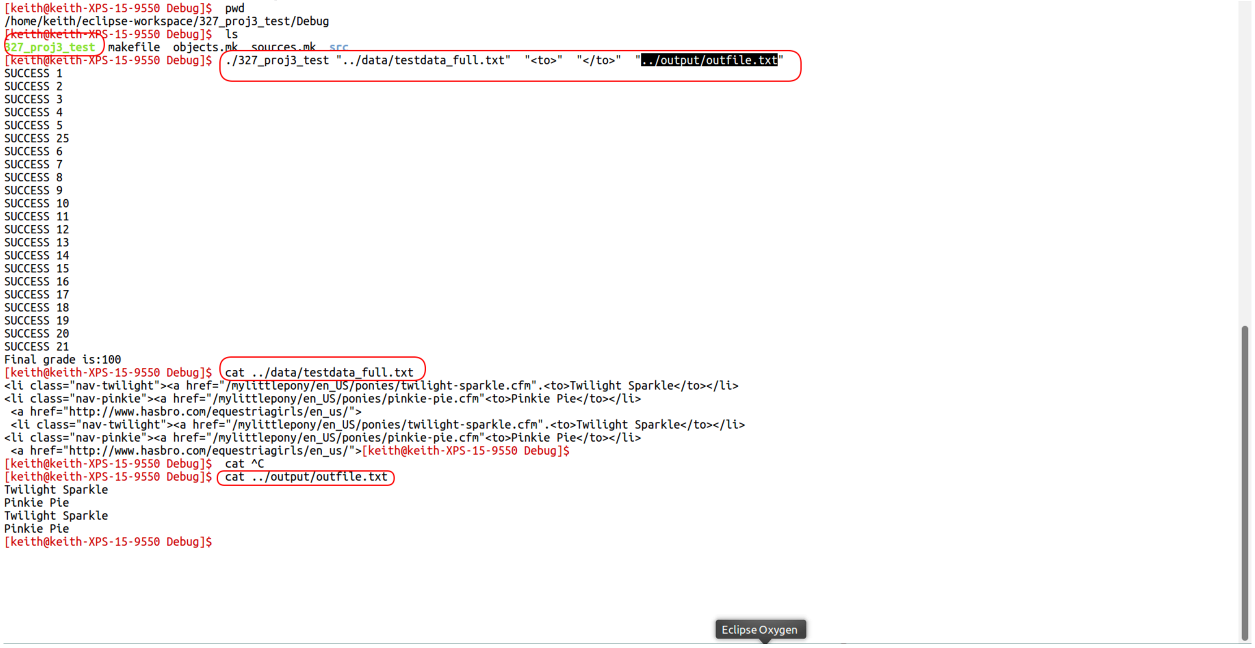
Please fill in all required content in;

- FileIO.cpp
- StringParserClass.cpp
- 327_proj3_test.cpp

Note that 327_proj3_test.cpp requires command line parameters to be passed in when the program is invoked, there should be 4 of them;

- the first is the filename to read data from
- the second is the first tag to search for
- the third is the second tag to search for
- and the fourth is the output file to write all the found data to

see below for sample run



```
[keith@keith-XPS-15-9550 Debug]$  pwd
/home/keith/eclipse-workspace/327_proj3_test/Debug
[keith@keith-XPS-15-9550 Debug]$  ls
327_proj3_test  makefile  objects.mk  sources.mk  src
[keith@keith-XPS-15-9550 Debug]$  ./327_proj3_test "../data/testdata_full.txt"  "<to>"  "</to>"  "../output/outfile.txt"
SUCCESS 1
SUCCESS 2
SUCCESS 3
SUCCESS 4
SUCCESS 5
SUCCESS 25
SUCCESS 6
SUCCESS 7
SUCCESS 8
SUCCESS 9
SUCCESS 10
SUCCESS 11
SUCCESS 12
SUCCESS 13
SUCCESS 14
SUCCESS 15
SUCCESS 16
SUCCESS 17
SUCCESS 18
SUCCESS 19
SUCCESS 20
SUCCESS 21
Final grade is:100
[keith@keith-XPS-15-9550 Debug]$  cat ../data/testdata_full.txt
<li class="nav-twilight"><a href="/mylittlepony/en_US/ponies/twilight-sparkle.cfm".<to>Twilight Sparkle</to></li>
<li class="nav-pinkie"><a href="/mylittlepony/en_US/ponies/pinkie-pie.cfm"<to>Pinkie Pie</to></li>
 <a href="http://www.hasbro.com/equestriagirls/en_us/">
 <li class="nav-twilight"><a href="/mylittlepony/en_US/ponies/twilight-sparkle.cfm".<to>Twilight Sparkle</to></li>
<li class="nav-pinkie"><a href="/mylittlepony/en_US/ponies/pinkie-pie.cfm"<to>Pinkie Pie</to></li>
 <a href="http://www.hasbro.com/equestriagirls/en_us/">[keith@keith-XPS-15-9550 Debug]$
[keith@keith-XPS-15-9550 Debug]$  cat ^C
[keith@keith-XPS-15-9550 Debug]$  cat ../output/outfile.txt
Twilight Sparkle
Pinkie Pie
Twilight Sparkle
Pinkie Pie
[keith@keith-XPS-15-9550 Debug]$
```

When finished StringParser_TEST will link 2 libraries StringParser.lib and FileReader.lib

Requirements

- Please submit just the following files:

  FileIO.cpp
  StringParserClass.cpp
  327_proj3_test.cpp

- Please ensure that you parse the text using char pointers, no algorithms or built in string parsing are allowed. Raw pointers only. You may save the parsed strings in a vector however.

- Please demonstrate a debug session, in person to me, of you stepping from 327_proj3_test.cpp to a function in StringParserClass.cpp. This is just to ensure that you have the libraries set up correctly.

Stuff to consider

- Follow my project outline
- **Read ref 1**
- I have provided the header files that serve as library contracts. You cannot change the header file at all! I will use my copy of this header file when linking to your library. Please implement all functions defined in the header file. You are of course able to implement other functions in the cpp file.
- Tags cannot be embedded within tags
- Read all the data from the file before you operate on it
- Tags are case sensitive <To> != <TO>
- What happens if no start or end tags are found?
- What if tags are malformed and right at the end of the file? Will your app read beyond the EOF and crash?
- If you change the header file that will break the contract between my testapp and your library, I may not be able to test it. Since I am using my copy of the library header files your application may also not compile either.
- See FileIO.h and StringParserClass.h (the contracts I've been going on about)
- See 327_proj3_Test project and examine its linkage to 327_proj3_fileIO

How I will Test your Solution

I will compile and link your solution.   I will probably use my own datasets to test regular and edge conditions. This is exactly the way that your code will be tested in industry BTW, with a primary eye to robust, stable operation under adverse conditions. I will not use your test application.