CSE687 - Object Oriented Design **Spring 2015**

Project #1 – File Catalogue

due Tuesday, February 10

version 1.4

Purpose:

This project requires you to analyze the directory structure on the local machine, looking for duplicate file names and searching for text in specified files. The C++ standard libraries provide a rich set of containers to support structuring data for this analysis. However, there is a surprising omission in the standard libraries – there is no support for managing directory information. For that you are encouraged to use a FileSystem facility you will find on the college server[1].

This project is only moderately challenging. That should give you time to analyze the support code, cited above, and to look carefully at the standard template library containers provided by the C++ standard libraries.

Requirements:

Your Catalogue project:

1. **shall** use standard C++[2] and the standard library, compile and link from the command line, using Visual Studio 2013, as provided in the ECS clusters and operate in the environment provided there.

2. **shall** use services of the C++ std::iostream library for all input and output to and from the user's console and C++ operator new and delete for all dynamic memory management.

3. **(3) shall** identify a set of files for analysis by supplying, on the command line, a path, one or more file patterns, and a switch /s which, if present, indicates that the entire directory tree rooted at the path is searched for matching files. If the switch is not present on the command line only the directory at that path is searched.

4. **(5) shall** construct a catalog of all files in the file set, saving each file name only once and saving each path only once, while preserving all of the containment relationships between directories and their files. That implies that each file storage will have to save a list of references into a set of paths where they are found. You will find the STL containers and iterators very useful for this. Please provide a storage class for this.
5. **(4) shall** support the use of a command line option /d that, when present, causes your program to emit a list of duplicate file names along with their paths.

6. **(3) shall** provide a command line option, /f<search text> which, when present, causes your program to list all the files stored in the catalog that contain the search text[3].

7. **(1) Shall,** if no options are provided on the command line, emit a brief summary, e.g., N Files found in M directories.

8. **(2) shall**, after construction of the catalog and emitting any specified results, accept from the console new text specifications for text searches in the catalog by providing text and file pattern(s). No other commands are to be accepted.

9. **(2) shall** provide a test executive package and a display package that, combined with the analysis facility, demonstrates you meet all the requirements of this specification. It is important that your demonstration is accurate, complete, and clear. Your score for meeting requirements will be based on this display. You will get no credit for requirements met but not accurately demonstrated.

10. **Shall** provide a compile.bat that builds your project and a run.bat that demonstrates you meet all the functional requirements stated above. You will need to run your project several times with different command lines in order to do that. Please test your compile and run.bat files by unzipping your submission in an empty directory on a different path than you used to develop your submission. Then type compile (return) and NOTHING ELSE to build your project. Then type run (return) and NOTHING ELSE to demonstrate you meet all requirements.

---

[1] http://ecs.syr.edu/faculty/fawcett/handouts/Coretechnologies/Cpp/Code/FileSystem-Windows/
[2] This means, for example that you may not use the .Net managed extensions to C++.
[3] You are not required to support regular expressions but will find the standard regular expression library useful if you wish to do that.