Contents

Lab 9

STL Stacks

Topics: Algorithm Efficiency, Searching

Group work: Group work is allowed for the labs, but each person must do their own coding and each person must turn in an assignment. Copying other peoples' code / code files is not allowed. Copied assignments will receive 0% for everybody involved.

Assignments must build: Assignments that don't build will automatically just be given a flat 50% score. I may still give feedback on what went wrong, and 50% is better than 0%, so turn in something - but ideally, make sure it builds.

For this lab, you will be implementing a simple course catalog program. This program will be able to tell you which prerequisite classes you need, in order from first to last.

9.1 Input file

The courses.txt file looks like the following. For this project, we are assuming each class only has one direct prerequisite.

```
COURSE      CS250       Basic_Data_Structures_Using_C++
PREREQ      CS235


COURSE      FL165       Elementary_Chinese_I

COURSE      FL166       Elementary_Chinese_II
PREREQ      FL165
```

9.2 Program output

The program output will look like this:

Main Menu

```
------------------------------------------------------------
 | LOADING COURSES |
 -------------------

 * 17 courses loaded

------------------------------------------------------------
 | MAIN MENU |
 -------------

 1. View all courses
 2. Get course prerequisites
 3. Exit

 >>
```

View all courses

```
------------------------------------------------------------------------
 | VIEW COURSES |
 ----------------

#    CODE      TITLE                                          PREREQS
------------------------------------------------------------------------
0    MATH111   Fundamentals_of_Mathematics
1    MATH115   Elementary_Algebra                             MATH111
2    MATH116   Intermediate_Algebra                          MATH115
3    MATH173   Precalculus                                   MATH116
4    MATH241   Calculus_I                                     MATH173
5    MATH242   Calculus_II                                    MATH241
6    MATH243   Calculus_III                                   MATH242
7    MATH254   Differential_Equations                         MATH243
8    CS134     Programming_Fundamentals                       NOEXIST
9    CS200     Concepts_of_Programming_Algorithms_Using_C++   CS134
```

```
10   CS210     Discrete_Structures_I                          MATH171
11   CS211     Discrete_Structures_II                         CS210
12   CS235     Object_Oriented_Programming_Using_C++          CS200
13   CS250     Basic_Data_Structures_Using_C++                CS235
14   FL165     Elementary_Chinese_I
15   FL166     Elementary_Chinese_II                          FL165
16   FL192     Intermediate_Chinese_I                         FL166

 Press ENTER to continue...
```

Get course prerequisites

```
------------------------------------------------------------
 | GET PREREQS |
 ---------------

 Enter class code

 >> CS250

Error! Unable to find course NOEXIST!

Classes to take:
1. CS134 Programming_Fundamentals
2. CS200 Concepts_of_Programming_Algorithms_Using_C++
3. CS235 Object_Oriented_Programming_Using_C++
4. CS250 Basic_Data_Structures_Using_C++
```

9.3 Program code

The following classes are already pre-written for you:

CourseNotFound exception

```
1  class CourseNotFound : public runtime_error
2  {
3      public:
4      CourseNotFound( const string& error );
5  };
```

Course struct

```
1  struct Course
2  {
3      Course() { Clear(); }
4
5      void Clear()
6      {
7          name = code = prereq = "";
8      }
9
10     string name;
11     string code;
12     string prereq;
13 };
```

The `Course` object contains a single course's information: The code ("CS250"), the class name ("Basic_Data_Structures_Using_C++"), and one prereq, if any ("CS235").

CourseCatalog class

```
1  class CourseCatalog
2  {
3      public:
4      //! Initializes the course catalog
5      CourseCatalog();
6
7      //! Runs the course catalog program
8      void Run();
9
10     private:
11
12     //! Loads all courses from a text file
13     void LoadCourses();
14
15     //! Displays list of all courses
16     void ViewCourses();
17
18     //! User enters course, list of prereqs are
       displayed
19     void ViewPrereqs();
20
21     //! Finds a course by its code and returns the
       course
22     Course FindCourse( const string& code );
23
24     //! The vector of courses
25     vector<Course> m_courses;
26 };
```

You will be implementing the `ViewCourses()`, `FindCourse( code )`, and `ViewPrereqs()` functions.

9.4 Functions to implement

9.4.1 ViewCourses

Use a for-loop to display all courses currently stored, including the index, their code, title, and prereqs

9.4.2 FindCourse

Input Parameter	`const string &`	`code`	The course code (e.g., "CS250") to search for.
Output Return	`Course`		The course whose 'code' variable matches the parameter.

Use a for-loop to search through all the courses in the `vector<Course>` `m_courses` member structure. If the current course's code is the same as the one passed in as a parameter, return this course.

OTHERWISE, if you've searched all courses and the function hasn't returned yet, that means the course isn't found - throw the CourseNotFound exception.

9.4.3 ViewPrereqs

This menu will ask the user to enter a class code. From that, you will try to search for the class we want prereqs for. Use a try/catch to find this course. (If there's an exception, display an error message and return).

Next, create a stack of Course objects, called prereqs. First, you will push the current course to the list.

Then, while the current course has a prereq, find the prereq of the current course, and store it. Push this prereq to the stack.

Use a try/catch within your while loop to detect exceptions. If a course is not found, just break out of the while loop.

Finally, use a second while loop to go through all the items in the stack, displaying the top-most item (first prerequisite) and popping the item until the stack of prereqs is empty.

9.5 Turn in

Once completed, turn in your `CourseCatalog.cpp` file into Canvas.