Project 2 Submissions

Daniel Loranger

CS-300, Week 7

Discussions:

Initially the code was developed following the pseudo code from week 6 and for the most part everything was working with the program except I could not get the alpha sorting to work properly. About 12 hours was spent attempting to find a solution.

Finally feeling like I had hit a dead end, I started over using the BST architecture previously developed in week 6 and adjusted it to utilize courses instead of bids. Further enhancements for error handling and special conditions were added to make a more robust user interface and more robust file loading experience.

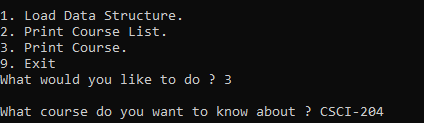
As the project did not require removal of courses based on the input requirements, and some other behavioral code, these were commented out, but left for future reference should they become required at a later iteration. This would allow quickly uncommenting the code and checking for intended operations without needing to relocate or regenerate the behavioral code from other sources.

Conclusions – Given the limited requirements provided, and a small sample input file, I feel the code is working as intended based on the testing and enhancements provided in the following pages. While this did not follow the planned pseudo-code, the architecture is considered stable and generally vetted based on previous usages, and will provide the customer with a responsive application that delivers the information and behavioral inputs.

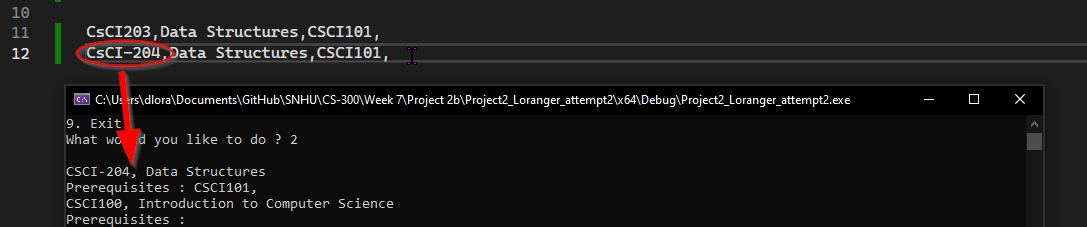
Development of BST and testing

The BST functionally came together in just a few hours and features appear to be working to specifications. The main issue I struggled

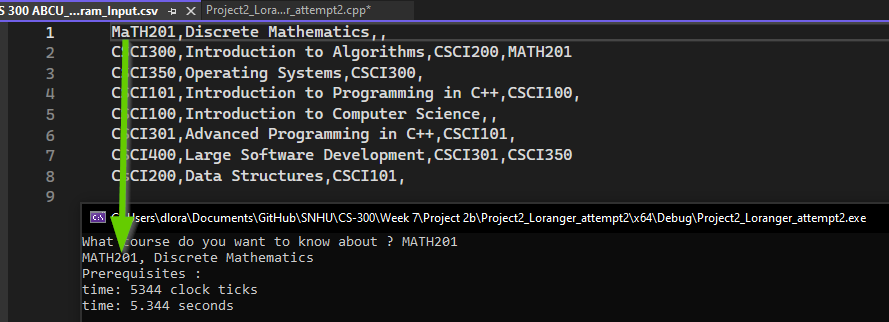
* User Menu
  + Development – The menu options are simple cout statements, and were bundled into a helper function displayMenu() to help keep the main loop readable. This format was implemented based on the sample inputs provided by the customer requirements input.
  + Testing
    - Intended behavior - The valid menu options are 1,2,3,9. Each of these were tested in the sequence 1,2,3 <input course name>, 9. This worked as expected, no faults were found. Menu item 3 requires a second input, which is shown in the following image. All text was presented in the customer requirements input.



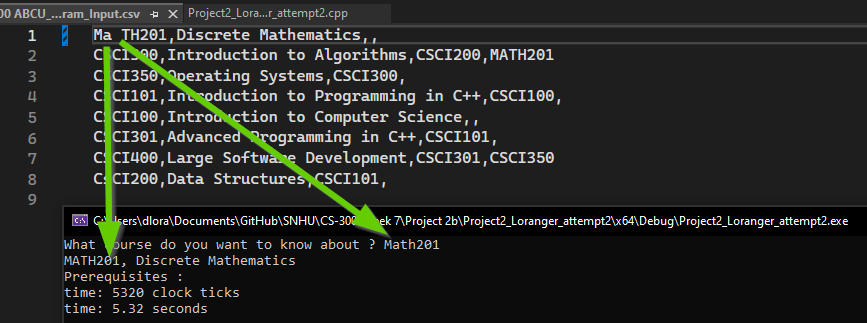
* + - Unintended behavior
      * Print list or course search before loading courses
        + helper function created to check for a non-initialized root rootNullCheck() supports a pre-test to alert the user that no courses are loaded yet.
      * Non-valid option selection
        + Alpha character – default case option is triggered which clears the invalid input and alerts the user of the invalid selection
        + Numeric character – default case option is triggered which clears the invalid input and alerts the user of the invalid selection
        + Random mixed alpha/white/numerics – default case option is triggered which clears the invalid input and alerts the user of the invalid selection
    - Enhanced behaviors
      * + While not specified, the user input to option 3 will accept a string of mixed case including white space and will trim the white space (math 123 becomes math123) and also capitalizes the course name (math123 becomes MATH123). This is done to ensure the user entry will match the course listing as described in the file loading behavior.
* File Read
  + Sample File provided
    - File is read as expected – the file is first parsed to get a list of all the valid course IDs contained in the file for consideration. Once this is completed, the courses are read in 1 at a time and the pre-requisites are all compare to the valid course IDs to ensure they are contained somewhere within the file. <design requirement fulfillment>.
  + Enhanced Behaviors
    - * While not presented as a consideration, the input file course id field could potentially come as mixed case. This field is always capitalized and white space is removed to ensure consistent behavior with the above formatting of the users input.
  + Testing
    - Extended course name
      * Some courses could contain additional characters such as a hyphen between the major and course sequence number such as MATH-123. This is considered something that would be common to enhance readability, and was tested for specifically. This is handled without issue



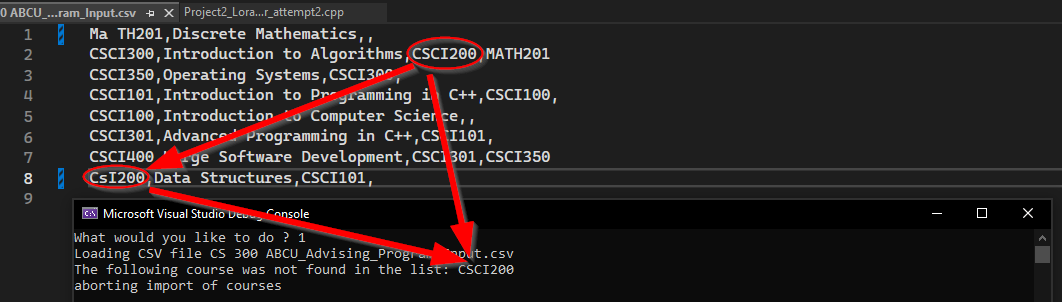
* + - Mixed case course name. MATH201 is changed to Math201 – all behaviors work as expected



* + - Mixed case course name with space. MATH201 is changed to “Ma th201” – all behaviors work as expected



* + - Missing prerequisite on course – “CSCI200” was changed to CsI200 – The program detects the missing prerequisite and aborts the import with alert provided to the user interface



* + - Mixed presentation of prerequisits
      * It was noted upon inspection that the presentation of the prerequisits is not consistent, some rows end with a comma, while others do not. 

This presented a behavioral issue as the comma is used as the delimitator. To account for this, a check is done on each row, and if the final comma is missing, it is added to ensure consistent behavior

* + - Empty row
      * While not presented as a requirement, it’s possible the input file could contain an empty row. This would cause an out of bounds issue. A check was put in 2 places to handle this condition by skipping that row



* + - Incomplete row
      * While not presented as a requirement, it’s possible the input file could contain an incomplete row (containing less than 3 commas). This would cause an out of bounds issue. A check was put in 2 places to handle this condition by skipping that row

