30/08/2017 SUMMER 2017

CS 202 EFFIECIENCY WRITE-UP FOR PROGRAM 5

BY TEJAS MENON; ASSIGNMENT BY PROFESSOR KARLA FANT

Program 5 for me was all about building familiarity with returning by value and not being able to set right and left pointers simply by calling the function. Although the code for that quickly became unreadable and complicated, it provided a simple one case assumption- you are always working with the value of the reference. In many ways, this improved code maintainability in that there was little need to look back at prototypes and refer back to code in other packages that might have been unavailable. But this additionally called for unnecessary functions to set and get node information. To avoid this, I would have implemented a BST within cars itself had I more time and make them automatically add nodes by themselves without involving an external higher level class.

A successfully implemented function this program was my search function, that sorted on the basis of a priority score added to each node and then altered with every search. The dealership BST managing the searchable cars list allowed for a quick creation and deletion, and since each car in the company was sorted by this priority score, I could choose to display all results in a prioritized fashion up to a certain integer, and the user could choose up to how many results they wanted displayed. A further improvement that could have been made to this design would have been to involve an interface BST (the tree classlist in java) and have it manage all the BST's in the program- since each implementation was very self similar - the dealerships managed cars and the company managed the dealerships, while the search BST was simply a slightly altered cars BST.

Prioritizing specific features to be higher up in the search results was another aspect of the program that worked well, as I had the search score increased by 3 when an engine matched, increased by 2 when the color matched and 1 when wheels matched. This realized in results that showed up on an equal level cars that had non matching wheels and color but a matching engine with cars that had a non matching engine but matching color and wheels- as their scores would be identical (3). A slight modification I had to make while displaying the BST was having a display that showed up from largest to smallest (based on priority score) rather than the conventional display.

Finally, having the option list be generated as a static function was very beneficial, as I could run the function once to populate the array, and subsequently have each car read from it for their choices rather than have a new option list be generated for each car that would have been wasteful of performance. The alternative of this would have also been more cluttering, as having the option list delivered as an argument would have required several function changes throughout the program.