Ojas Bardiya

UID:505145284

CS 32 – Homework #4

2) In the single-argument form of insert, it determines the position at which the item should be placed by comparing the value at the current position in order to place the item before the next biggest value, i.e., in ascending order. However, when trying to use it in the case of Coord, it must have the ‘>’ operator overloaded in order to successfully compare two different items. Since the Coord class does not have this operator overloaded we get an error.

4) b) For a solution to be recursive in nature, it should be able to be broken down into simpler sub-problems which can be solved independently. In the case of this function **listAll**, it requires the Domain\* d parameter to access the subdomains of the domain, while also editing the labels of domain which requires a separate string parameter. Thus, solving it recursively requires two parameters and not one.

5) a) The time complexity of this algorithm is O(N^3).

This can be seen as we get three levels deep, i.e., one for each for loop. Though the if statements are of constant runtime, each for loop runs until we hit N -> each for loop has a runtime of O(N). As we move towards the innermost loops, the time complexity of each loop is factored in to make the overall time complexity O(N\*N\*N) = O(N^3).

b) Even in this case, the time complexity of the algorithm is O(N^3).

This is because in the worst-case scenario, i will be N and the problem is essentially the same as 5) a). In the average scenario, the time complexity of the inner two for loops is O(N^2). Combining this with the outer for loop we get the total time complexity as O(N^3).

6) a) The time complexity for visiting each item in the first interleave is O(N^2).

Observing the first for loop, we see that k varies from 0 to N. The get function then goes again through the linked list k to N – 1 times, which is approximately equal to N in the worst-case scenario. This gives it a time complexity of O(N) (in reality O(2N) as it does so for both sequences). The next part of the program essentially copies the remaining values from the bigger sequence into result, which has a time complexity of O(N^2). Thus, the time complexity of this particular interleave function is O(N^2) (as we ignore coefficients).

b) The time complexity for this particular implementation of interleave is O(N), making it better than the implementation in 6) a). The first for loop goes through both sequences for about N times, while insertBefore considers only the value of the current Node in the linked list and inserts it there. This makes it a constant time function O(c). The next for loop enters the remaining items from the larger sequence, making it have a time complexity of O(N). The swap function has a time complexity of O(N) as well. Thus the overall time complexity is O(N) (as we ignore coefficients).