Jason Tran

CS32 Homework 4

3/7/23

**1e)** It fails at the 3rd test case because on the 2nd iteration it adds in more items into the container. This will break the loop(function) since the iterator doesn’t get reset/updated once it continues, which it needs to because adding stuff might shuffle memory around, so it is undefined behavior.

**3)** Since coord is not a primitive type, some operators aren’t defined for them like < or >. However this is needed to use the insert functions as that function calls firstAtLeast which uses the < operator to compare in its forloop. Also, after it uses the undefined == operator.

**4b)** If we were only given one parameter and restricted to not using any global variables, then there would be no way to track the parent string/strings before using recursion, which means you can’t concatenate correctly and get the desired output.

**5a) O(N^3)** because the outer loop runs N times, the middle loop would run N times and the inner loop would run N times. And since they are all nested you multiply them because the outer loop would make the inner loops run their loops. You can disregard the other operations which are o(1).

**5b)** It is still **O(N^3)** because even though the second loop looks simpler, O(N/2). You disregard the coefficients and multiply all the loops together and get the same general result.

**6a) O(N^2)** because the for loop loops N times (sp = size of set2/set, they are both N). And since the insert function is also called which is O(N), from the for loop in findfirstatleast, it does it N times (because of the for loop), which results to O(N\*N).

**6b)** **O(NlogN)** because it takes O(N) to do the beginning inserts, for a vector. Then the sort algo is O(NlogN). Erase would be O(N) and the for loop at the bottom would be O(N). Since we want to simplify and only take the most important terms the final result would just be O(NlogN).

**6c)**  **O(N)** because the first while loop loops through N elements and either uses the function insertBefore which is O(1) or not. Then the for loop at the bottom loops through N elements (O(N)) and uses that same constant function. So O(N+N) is O(N).