CS32 Homework 4 Solutions

* Arsh Malik

**Ans2**.

We get compilation errors in Map<Coord, int>::insert because we haven’t defined comparison operators such as >, == for Coord. We don’t get compilation error for Map<Coord, int>::insert because the comparison operators are already defined for built-in data types.

**Ans3e.**

Whenever we add/delete an item in a vector, it may shuffle the memory around, and then the iterator may not point to the right place. Therefore, our iterator gets invalidated and when we try to follow the iterator, we get wrong results. This is the reason test 3 fails.

**Ans4b.**

Given the constraints in part a, i.e., not using global or static variables, and implementing the listAll function recursively, we would require a two-parameter listAll function which has one string parameter which is needed to keep track of the class hierarchy. It won’t be possible otherwise if we use a single parameter listAll function.

Every class knows about its sub classes, but it doesn’t know anything about its parent class. Therefore, we need a string parameter to keep track of class hierarchy.

**Ans5**

**a.)**

Time complexity: O(N3)

Reason: There are 3 nested for loops that perform N iterations each.

**b.)**

Time complexity: O(N3)

Reason: The outer for loop N times. The for loop for (int j = 0; j < i; j++) runs in the worst case, N/2 times. The innermost for loop for (int k = 0; k < N; k++) runs N times. Therefore we get time complexity of N\*(N/2)\*N which is O(N3).

**Ans 6a.**

To calculate the time complexity, we must examine the main functions performed in the reassign function. Let f(n) be the number of critical operations that reassign performs.

1. *m.get(0, prevKey, value0);* is a 1 step process.
2. The loop *for (int i = 1; i < m.size(); i++)* runs N times. Inside the for loop, m.get(i, k, v) performs its task in N steps worstcase and insert function performs the insertion in N steps as it calls doInsertOrUpdate which in turn calls findFirstAtLeast(), performing a linear search on the linked list.
3. The final insertion takes N steps.

Therefore our function f(n) = 1+N\*(N+N)+N = 2N2 + N + 1.

Time complexity of reassign is **O(N2).**

**Ans 6b.**

Since the nodes in the linked list are visited only once using the for loop, the time complexity for this algorithm is **O(N).** Therefore, this algorithm is better than the implementation in part a.