Ali Mirabzadeh

305179067

HW4

2. The reason why the call to Sequence<Coord>::insert causes a compilation error is because that member function relies on the comparison operator > in order to determine where to insert the indicated value. Since the comparison operator > was not defined for the class Coord, the compiler is unable to perform the comparison and throws a compilation error.

3b. It is not possible to implement the one-parameter listAll as recursive because with only one parameter it is not possible to pass in a simplifying case to the recursive call each time. Every time we call listAll, we want to pass a simpler/shorter argument till get to the breakpoint, by having one parameter to be passed such a functionality would be ignored

4a. The time complexity is O(N^3). The reason is that there are three for loops that each runs three times and they are nested. So the outermost would run N times, then the inner and the innermost one would run N times as well.

4b. The time complexity is still O(N^3). This is because we still have a for loop nested inside a for loop nested inside a for loop, and even though the second loop might not run N times each time, in the worst case it will. Since time complexity is supposed to be based off the worst-case scenario which would be O(N^3).

5a. The worst-case would be O(N^2) since each get and insert has a for loop that runs N times. They are both inside another for loop that runs N times. So the worst-case would be O(N^2)

5b. The time complexity is O(N logN). The first two loops each has O(N), then there is the sort function in which as specified has O(N logN), the while loop has a O(N). Finally, the for loop would run 2N times since v.size() has the size of both set1 and set2

5c. The time complexity is O(N). Since the while loop would run N in the worst case, N for each p1 and p2. The for loop would run N times, but it is outside of the while loop, so the time complexity is O(N).