2. The call to Map<Coord, int>::Insert causes a compilation because the insert function calls the doInsertOrUpdate() function, which in turn calls the find() function. The find function relies on comparison operators, for which the Coord class does not have a custom version of. Therefore, the insert function fails because there is no comparison.

3. b. We can’t solve this problem with only one-parameter because we kind off need a string that holds the previous outputs in order to write the correct outputs. Without the string path, we would not be able to solve this problem unless we change the function’s return type into something like string.

4. a. The time complexity of this algorithm is O(N^3) because there are three for loops that all

run N times, thus N\*N\*N or N^3.

4. b. The time complexity of this algorithm is still O(N^3) because in the worst case scenario, it is still possible that you run the second loop almost N times. Therefore, Big O is not truly affected.

5. The time complexity of this algorithm is O(N^2) because there is a for loop that runs roughly N times. Then, the functions that are called (get(), insert(), and erase()) all end up calling the find(), which does a linear search of the Linked List. The time complexity of find() would be O(N). Therefore, we can conclude that the time complexity of the algorithm would be O(N\*N) or O(N^2).