Komal Prakash

304980756

HW 4 Responses

Problem 2)

The call to Map<Coord, int>::insert causes an error because insert calls doInsertOrUpdate which calls the find function at the beginning. This find function is causing the error because it is trying to compare Coord with a != operator. There needs to be a function that defines the != operator when a struct/class like Coord is created and used as a type in Map.

Problem 3)

1. This has a time complexity of O(N^3). This is because it has three nested for loops with each loop having N indices to go through. So, N\*N\*N.
2. This also has a time complexity of O(N^3). This is because, the outer for loop would loop through N items. The second nested for loop only goes till i, but it would still go through N indices when i is equal to N and we only keep the highest order terms so we can still treat the j loop as in the first case. For each j loop, there is another N items that the nested k loop goes through, so the Big O is still N\*N\*N which is N^3.

Problem 4)

1. This has a time complexity of O(N^2). This is because the for loop in reassign goes through N indices because N is the size of the m. Within this loop, the get function is called which also contains a for loop that goes through the N items in order to find the value at the specified index. Thus, this is like 2 nested for loops so the Big O is N\*N. Other functions called like erase, insert, and swap have smaller Big Os than N^2, that when added on, would not significantly add to the overall time complexity.
2. This has a time complexity of O(N). It only has one for loop that visits all N items of the list. Even though it also visits the next item in each iteration that would just make it 2N, which overall maintains a time complexity of O(N) for this algorithm. This is a better implementation than a, because it uses much simpler logic, is quite clear, and uses a smaller Big O.