Elijah Hubbard

2/11/2025

1. O(n^2)

2. O(n^4)

3. O(n^2)

4. O(n^3)

5. O(1)

6. O(1)

7. O(n)

8. O(n)

9. O(nlog(n))

10. O(n^3)

11. O(1)

12. O(n)

13. It is a linear time complexity relationship. When you remove from the middle, everything must shift to that position in the array.

14. O(1)

15. O(n) due to resizing. O(1) since you don't have to adjust the array

16. Getting and setting

17. The rudrata problem means to visit every item/object on a graph exactly once. The puzzle above shows a graph that has the possibility to be solved in the same manner since everything is connected to each other.

18. [O(n!))] Initially I thought O(n^2) but that pathway is to convient when it can be any number of stars each having any number of pathways that would need to be checked. So since they could vary intensely, it would make more since to take 2 to the power of n(number of stars), as each path way could easily have 2 paths(or more).

19. The worst case run time[O(indeterminate)

20. I thought of it like a stack of cards. If you have 56 cards, their is a possibility you would have to shuffle it 56 separate times until you get it in the order you wanted meaning it would be 56^56.

21. The average case run time is O(n! * n)

22. The though process is that since there are n! number of ways for the cards to be sorted(Every time you add a card, the factorial increases making the number of ways it can be sorted increases), and it takes n time to look through one by one to see if it is in the correct order, you then have to multiply them by each other to understand the run time.

23. Time complexities of Lab 2

1. Unique: O(n^2) since its a nested for lop meaning it would iterate through the list multiple times as it goes through the array.

2. AllMultiples: O(n) since it would iterate through the array one time

3. allStringsofSize: O(n) since it would iterate through the array one time

4. isPermutation: O(nlogn) since sorting splits everything into halves which runs on O(logn) time. But aswell it must resort everything that it havles into the correct order one by one which runs on n time. Meaning you have to multiply these together to get the total time which is nlogn.

5. stringsToListOfWords: O(n) since the for loop is n time.

6. removeAllInstances: O(n^2) since the for loop is working on n time, and remove also works on n time meaning n^2 time.