Analysis of Algorithms

Refer the Following Links - Link 01, Link 02

01. **Consider the following code snippet and write the time complexity (big O) of each code.**

> *A.* *FOR i FROM 0 TO 2n-1*
> *PRINT i*

> *B.* *FOR i FROM 0 TO n-1*
> *FOR j FROM 0 TO i*
> *PRINT i, j*

> *C.* *FOR i FROM 0 TO n-1*
> *PRINT i*
> *FOR j FROM 0 TO 2n-1*
> *PRINT j*

> *D.* *i = n*
> *WHILE i > 0*
> *PRINT i*
> *i = i // 2*

> *E.* *FOR i FROM 1 TO n*
> *FOR j FROM 1 TO n*
> *FOR k FROM 1 TO 10*
> *PRINT i, j, k*

02. Given the following list of 25 student names:

["Adam", "Bella", "Carl", "Dina", "Ella", "Finn", "Gina", "Harry", "Ivy", "John", "Kim", "Leo", "Mia", "Nate", "Omar", "Pia", "Quin", "Ria", "Sam", "Tess", "Uma", "Vik", "Will", "Xena", "Zoe"]

to find **"Nate"**

I.   Given a function to find a specific student's name and explain the time complexity of the function.

FUNCTION LinearSearch(student_names, target_name):
*   FOR each index i from 0 to length of student_names - 1:*
*   IF student_names[i] == target_name:*
*   RETURN i*
*   RETURN -1*

II.   Explain the time complexity of the linear search function

2.   Given a sorted list of 2000 integers, the following function helps to find a specific number using binary search and explain the time complexity of the function.

FUNCTION BinarySearch(sorted_numbers, target_number):
*   left = 0*
*   right = length of sorted_numbers - 1*

*   WHILE left <= right:*
*   mid = (left + right) // 2*

*   IF sorted_numbers[mid] == target_number:*
*   RETURN mid*
*   ELSE IF sorted_numbers[mid] < target_number:*
*   left = mid + 1*
*   ELSE:*
*   right = mid - 1*

*   RETURN -1*

I.   Test your function with the array GENERATE (From 1 TO 2000) to find the number 1450
II.   Explain the time complexity of the binary search function.

3.   You are given an sorted array of 150 numbers, would it be more efficient to use linear search or binary search at this point? Justify your answer.

Quizzes

1. What is the best-case time complexity of a linear search?

   a) O(1)

   b) O(n)

   c) O(log n)

   d) O(n^2)

2. How is time complexity affected by nested loops?

   a) It always becomes linear

   b) It increases multiplicatively

   c) It stays constant

   d) It depends on recursion

3. What is the primary goal of time complexity analysis?

   a) To determine the efficiency of an algorithm.

   b) To determine the hardware requirements of an algorithm.

   c) To analyze the programming language used.

   d) To evaluate the user interface of a program.

4. Which metric is used to represent the number of basic operations an algorithm performs?

   a) Input size

   b) Output size

   c) Basic operation

   d) Time complexity

5. What is the best-case scenario for binary search?

       a) Target is the smallest value

       b) Target is the largest value

       c) Target is the middle value

       d) Target is not in the list

6. In the context of time complexity, what does 'n' typically represent?

       a) Number of algorithms

       b) Input size

       c) Output size

       d) Number of basic operations

7. What type of analysis considers the most challenging situation for an algorithm?

       a) Best-case analysis

       b) Average-case analysis

       c) Worst-case analysis

       d) Every-case analysis

8. What is the worst-case time complexity of a linear search when the element is not present in the array?

       a) $O(1)$ b) $O(n)$

       c) $O(\log n)$ d) $O(n^2)$

9. Which factor does **not** influence time complexity?

  a) Size of input

  b) Speed of CPU

  c) Number of iterations

  d) Nature of algorithm

10. Why is worst-case analysis often considered the most crucial aspect of algorithm design?

  a) It provides an optimistic view.

  b) It ensures reliability and performance guarantees.

  c) It is easier to compute.

  d) It is rarely used in practical scenarios.

11. What does Big-O notation represent?

  a) Best-case time complexity

  b) Average-case time complexity

  c) Worst-case time complexity

  d) Space complexity

12. Given an array of 10 elements, what would be the best-case time complexity for a linear search to find a specific element?

  a) O(10)

  b) O(1)

  c) O(log 10)

  d) O(n)

13. If an algorithm has a time complexity of O(n^2), how will the time required change if the input size doubles?

    a) It will remain the same.

    b) It will double.

    c) It will quadruple.

    d) It will increase exponentially.

14. What does time complexity O(log n) usually imply?

    a) Constant time

    b) Recursive halving

    c) Linear growth

    d) Exponential growth

15. What is the average-case time complexity for a linear search in an array with 'n' elements?

    a) O(1)

    b) O(n)

    c) O(log n)

    d) O(n^2)

16. Why is average-case analysis rarely done in practical cases?

    a) It is too simplistic.

    b) It requires understanding the mathematical distribution of all possible inputs.

    c) It is less important than worst-case analysis.

    d) It is usually misleading.

17. For a binary search algorithm, what is the best-case scenario?

  a) The target element is the middle element of the array.

  b) The target element is the first element of the array.

  c) The target element is the last element of the array.

  d) The target element is not in the array.

18. What is the significance of knowing the worst-case time complexity of an algorithm in real-time systems?

  a) It helps in designing user interfaces.

  b) It ensures system reliability and stability.

  c) It reduces the development cost.

  d) It improves the average performance of the algorithm.

19. Given an array of size 'n', what will be the best, average, and worst-case time complexities for a linear search?

  a) O(1), O(log n), O(n)

  b) O(1), O(n/2), O(n)

  c) O(1), O(n), O(n^2)

  d) O(log n), O(n), O(n^2)

20. Which notation is used to represent the average-case time complexity of an algorithm?

  a) Big-O notation

  b) Omega notation

  c) Theta notation

  d) Delta notation

21. In the context of algorithm analysis, what does the term 'basic operation' refer to?

a) The most frequently executed instruction of the algorithm.

b) The least frequently executed instruction of the algorithm.

c) The initialization step of the algorithm.

d) The termination step of the algorithm.

22. Why is best-case analysis considered misleading for practical applications?

a) It assumes all inputs are identical.

b) It does not account for average performance.

c) It focuses only on the most favorable conditions.

d) It provides an upper bound on the time complexity.

23. For the linear search algorithm, under what condition does the worst-case scenario occur?

a) The element is found at the first index.

b) The element is found at the middle index.

c) The element is found at the last index or not found at all.

d) The element is found multiple times.

24. How does worst-case analysis help in resource allocation for algorithm implementation?

a) By identifying the average resource requirements.

b) By determining the minimum resources needed.

c) By ensuring adequate provisioning for the most resource-intensive scenarios.

d) By reducing the computational complexity.