Spondon Ganguli
An Educator, Artist & Author

Important questions on Computational Complexity

Q1. With the help of an example, explain the Constant factor in time complexity.

Answer: Constant factor refers to the idea that different operations with the same complexity take slightly different amounts of time to run. For example, three addition operations take longer than a single addition operation. Another example, 6n, 6 and n are factors. A constant factor is anything that doesn't depend on the input parameter(s) (n in this case). Regardless of what we make n, 6 will always stay 6, so it's constant.

Q2. What is Constant Time in Complexity?

Answer: When your algorithm is not dependent on the input size n, it is said to have a constant time complexity with order O(1). This means the run time will always be the same regardless of the input size.

Q3. What is Big 'O' notation? State its significance. [ISC 2010]

Answer: Big O Notation is the measurement of the complexity of an algorithm with respect to the time factor and space(memory) factor. Big O Notation allows programmers to classify algorithms depending on how their run time or space requirements vary as the input size varies. For example, runtime complexity for Linear Search is O(n) whereas for Binary Search is O(log n)

Q4. Distinguish between the worst-case and best-case complexity of an algorithm. [ISC 2011 a]

Answer: The best case is the function which performs the minimum number of steps on input data of n elements whereas the worst case is the function which performs the maximum number of steps on input data of size n.

Q5. What are the two major factors that influence the efficiency of an algorithm? [ISC 2011 b]
Answer: Time and space complexity are the two main measures for calculating algorithm efficiency. Where time measures how long it takes to process the algorithm, space measures how much memory is used.

Q6. What is the role of constant factors in computational complexity? [ISC 2012]

Answer: When Big 'O' notation is used, constant and low-order terms are dropped. However, if 2 algorithms have the same Big 'O' time complexity then also one may be faster than the other. In such cases, constants do matter in terms of which algorithm is faster.
For example, suppose algorithm 1 requires N^2 time and algorithm 2 requires $10*N^2+N$ time. For both the algorithms, the time is $O(N^2)$, but algorithm 1 will be faster than algorithm 2.

Q7. Define the terms complexity and big 'O' notation. [ISC 2014]

Answer: Complexity is the measurement of an algorithm with respect to time, space and complexity. Big 'O' notation is the unit of measurement of the complexity of an algorithm.

Important questions on Computational Complexity

Q8. Give the meaning of the following common expression in Big O notation: [ISC 2015 a]
O(N) and O(N^2)

Answer: O(N) is the complexity of a single loop or condition/expression. O(N^2) is the complexity of a nested loop with each loop going up to N.

Q9. List any two cases to analyse algorithm complexities. [ISC 2015 b]

Answer: Best-case and worst-case complexities

Q10. Define the dominant term. Explain with an example how it affects the complexity. [ISC 2016]

Answer: Dominant term: While determining the performance of an algorithm, we have to also consider the term, which plays a vital role and affects the performance of the function. Such a term is known as a dominant term.
Let an algorithm have both a nested loop running it N^2 times and a single loop running N times. In this case, N^2 is the dominant term and hence, the complexity is O(N^2).

Q11. Distinguish between time complexity and space complexity.

Answer: Time complexity is a function that describes how long an algorithm takes in terms of the quantity of input it receives. Space complexity is a function that describes how much memory (space) an algorithm requires for the quantity of input to the method.

Q12. Distinguish between the worst-case and average-case complexity of an algorithm.

Answer: The worst case is the function which performs the maximum number of steps on input data of size n whereas the average case is the function which performs an average number of steps on input data of n elements.

~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~