Section 1: Time & Space Complexity

Q1. Which notation is commonly used to describe the *worst-case* time complexity of an algorithm?
A. Ω (Omega)
B. Θ (Theta)
C. O (Big O)
D. o (Small o)

Q2. If an algorithm has a time complexity of $O(n^2)$, doubling the input size approximately increases the running time by what factor?
A. 2 times
B. 4 times
C. n times
D. Exponential increase

Q3. Consider an algorithm that has a constant time complexity, regardless of the input size. Which of the following best describes its complexity?
A. O(1)
B. O(n)
C. O(log n)
D. O(n!)

Q4. Which statement best describes *space complexity*?
A. It only counts the number of CPU cycles required.
B. It measures the amount of memory used by an algorithm as a function of input size.
C. It determines how quickly an algorithm executes.
D. It is the maximum number of loops in the algorithm.

Q5. What is the space complexity of an algorithm that uses an extra array equal in size to the input?
A. O(1)
B. O(log n)
C. O(n)
D. $O(n^2)$

Q6. Which complexity class denotes a linear increase in running time relative to the input size?
A. O(1)
B. O(log n)
C. O(n)
D. O(n log n)

Q7. If an algorithm performs nested iterations over the input, each iterating n times, what is its time complexity?

A. O(n)

B. O(n log n)

C. $O(n^2)$

D. O(2^n)

Q8. Which scenario likely uses a trade-off between time complexity and space complexity?

A. Sorting an already sorted array

B. Using caching (memoization) to avoid repeated computations

C. Recursion without a base case

D. Comparing two numbers

Q9. What does Big O notation primarily describe?

A. The best-case scenario performance

B. The average-case performance

C. The worst-case upper bound on running time

D. The exact number of operations performed

Q10. An algorithm that divides the problem in half at every step usually has which time complexity?

A. O(1)

B. O(log n)

C. O(n)

D. O(n!)

Section 2: Arrays Basics (Theoretical)

Q1. What is an array?
A. A collection of items stored at non-contiguous memory locations
B. A list-like data structure that stores elements sequentially in contiguous memory
C. A dynamically resizable list used only in high-level programming languages
D. A type of linked list with bidirectional pointers

Q2. Which of the following is a primary advantage of using arrays?
A. Dynamic memory allocation without a fixed size
B. Quick access to elements using index positions
C. Built-in support for complex data structures
D. Efficient insertion and deletion at arbitrary positions

Q3. Which operation on an array has a time complexity of O(1)?
A. Appending an element at the end (if capacity is available)
B. Inserting an element at the beginning
C. Removing an element from the middle
D. Shifting elements after deletion

Q4. In most programming languages, how is the first element of an array typically indexed?
A. 0
B. 1
C. It depends on the language
D. -1

Q5. What is the drawback of a static array compared to a dynamic array?
A. Faster element access
B. Fixed size, requiring allocation of memory up front
C. Lower memory overhead
D. Better cache performance

Q6. How does a dynamic array handle the insertion of elements when it is full?
A. It raises an error immediately
B. It creates a new, larger array and copies the old elements over
C. It discards the new element
D. It automatically compresses the existing array

Q7. Which of the following operations is typically most costly in terms of time complexity on an array?
A. Accessing an element by index
B. Updating an element by index
C. Inserting an element at the start
D. Iterating over the array

Q8. What is the average-case time complexity for searching an element in an unsorted array?
A. O(1)
B. O(log n)
C. O(n)
D. O(n log n)

Q9. Which of the following is true regarding arrays in most programming languages?
A. Arrays are immutable by default
B. Arrays have a predetermined fixed size (unless implemented dynamically)
C. Arrays automatically sort their elements upon insertion
D. Arrays provide built-in methods for multi-dimensional lookup without additional libraries

Q10. Multi-dimensional arrays are often used to represent:
A. A single list of items
B. Complex data structures like trees
C. Matrices or grids
D. Linked lists

Section 3: Binary Search

Q1. What is the basic precondition for performing a binary search on an array?
A. The array must be unsorted
B. The array must be sorted
C. The array must contain only unique elements
D. The array must be of even length

Q2. What is the average-case time complexity of binary search?
A. O(1)
B. O(log n)
C. O(n)
D. O(n log n)

Q3. In a binary search, how is the middle element of an array determined?
A. It is always the first element
B. It is always the last element
C. It is the element at index (low + high) / 2
D. It is selected randomly

Q4. What happens if the target value is smaller than the middle element during a binary search?
A. The search continues in the right subarray
B. The algorithm terminates immediately
C. The search continues in the left subarray
D. The search moves to the next adjacent element

Q5. If a binary search fails to find the target value, what is typically returned?
A. The target value
B. The position of the nearest element
C. A special indicator (e.g., -1 or null)
D. The size of the array

Q6. Which best describes binary search's method of dividing the search interval?
A. Linear division
B. Iterative halving
C. Exponential separation
D. Quadratic reduction

Q7. When implementing binary search recursively, what is the primary condition that stops the recursion?
A. When the target is found
B. When the lower index exceeds the upper index
C. Both A and B
D. When the array is completely traversed

Q8. What type of search is binary search considered as?
A. A brute-force search
B. A divide-and-conquer algorithm
C. A dynamic programming approach
D. A greedy algorithm

Q9. Which scenario might require modifications to the standard binary search algorithm?
A. Searching for an exact match in a sorted array
B. Searching in a circularly sorted array
C. Accessing an array element by index
D. Iterating through all elements in a sorted array

Q10. Which of the following is a benefit of using binary search over linear search in sorted data?
A. Binary search is easier to implement
B. Binary search requires no preconditions
C. Binary search significantly reduces the number of comparisons
D. Binary search works on both sorted and unsorted arrays equally well