1. What is an algorithm?
    - A) A hardware component
    - B) A software application
    - C) A finite set of instructions that accomplishes a particular task
    - D) A type of data structure
2. Which of the following is NOT a characteristic of an algorithm?
    - A) Input
    - B) Output
    - C) Complexity
    - D) Definiteness
3. What is the first step in designing an algorithm?
    - A) Decision making on data structures
    - B) Understand the problem
    - C) Algorithmic verification
    - D) Analysis of the algorithm
4. What is a data structure?
    - A) A way to store data on a hard drive
    - B) A particular way of storing and organizing data in a computer so that it can be used efficiently
    - C) A type of algorithm
    - D) A programming language
5. Which factor is NOT considered when selecting a data structure?
    - A) Resource constraints
    - B) Basic operations support
    - C) Aesthetic design
    - D) Performance requirements
6. What do the order of growth for varying input sizes indicate in algorithm analysis?
    - A) How the running time of an algorithm scales with the input size
    - B) The maximum memory usage of an algorithm

- C) The programming language used for implementation
- D) The ease of implementation

7. What does the Big O notation represent?
    - A) The lower bound of an algorithm’s running time
    - B) The exact running time of an algorithm
    - C) The upper bound of an algorithm’s running time
    - D) The average running time of an algorithm
8. In a max-heap, which of the following is true?
    - A) The root is the minimum element
    - B) Every parent node is greater than or equal to its children
    - C) Every parent node is smaller than or equal to its children
    - D) The tree is always complete and balanced
9. Which method is used to build a max-heap from an unordered array?
    - A) MAX-HEAPIFY
    - B) BUILD-MAX-HEAP
    - C) INSERT
    - D) DELETE
10. What is the goal of Heapsort?
    - A) To build a min-heap from an array
    - B) To build a max-heap and then sort the array
    - C) To implement a linked list
    - D) To optimize memory usage
11. What is hashing primarily used for?
    - A) To sort data efficiently
    - B) To perform insertion, deletion, and search operations in constant average time
    - C) To create visual representations of data
    - D) To compile programming languages
12. Which method is NOT a type of hash function?
    - A) Truncation Method

- B) Folding Method
- C) Sorting Method
- D) Division Method

13. What is a primary characteristic of a good hash function?
    - A) It should be easy to compute
    - B) It should use a lot of memory
    - C) It should produce many collisions
    - D) It should depend on the operating system
14. What technique is used in open addressing to resolve collisions?
    - A) Separate Chaining
    - B) Linear Probing
    - C) Linked Lists
    - D) Binary Search Trees
15. What does double hashing use to eliminate clustering problems?
    - A) A secondary hash function
    - B) A larger hash table
    - C) Linear search
    - D) Tree structures
16. **What is an algorithm?**
    - A) A hardware component
    - B) A software application
    - C) A finite set of instructions that accomplishes a particular task
    - D) A type of data structure
17. **Which of the following is NOT a characteristic of an algorithm?**
    - A) Input
    - B) Output
    - C) Complexity
    - D) Definiteness
18. **What is the first step in designing an algorithm?**

- A) Decision making on data structures
- B) Understand the problem
- C) Algorithmic verification
- D) Analysis of the algorithm

19. **What is a data structure?**
    - A) A way to store data on a hard drive
    - B) A particular way of storing and organizing data in a computer so that it can be used efficiently
    - C) A type of algorithm
    - D) A programming language
20. **Which factor is NOT considered when selecting a data structure?**
    - A) Resource constraints
    - B) Basic operations support
    - C) Aesthetic design
    - D) Performance requirements
21. **What do the order of growth for varying input sizes indicate in algorithm analysis?**
    - A) How the running time of an algorithm scales with the input size
    - B) The maximum memory usage of an algorithm
    - C) The programming language used for implementation
    - D) The ease of implementation
22. **What does the Big O notation represent?**
    - A) The lower bound of an algorithm's running time
    - B) The exact running time of an algorithm
    - C) The upper bound of an algorithm's running time
    - D) The average running time of an algorithm
23. **In a max-heap, which of the following is true?**
    - A) The root is the minimum element
    - B) Every parent node is greater than or equal to its children
    - C) Every parent node is smaller than or equal to its children

- D) The tree is always complete and balanced

24. **Which method is used to build a max-heap from an unordered array?**
    - A) MAX-HEAPIFY
    - B) BUILD-MAX-HEAP
    - C) INSERT
    - D) DELETE
25. **What is the goal of Heapsort?**
    - A) To build a min-heap from an array
    - B) To build a max-heap and then sort the array
    - C) To implement a linked list
    - D) To optimize memory usage
26. **What is hashing primarily used for?**
    - A) To sort data efficiently
    - B) To perform insertion, deletion, and search operations in constant average time
    - C) To create visual representations of data
    - D) To compile programming languages
27. **Which method is NOT a type of hash function?**
    - A) Truncation Method
    - B) Folding Method
    - C) Sorting Method
    - D) Division Method
28. **What is a primary characteristic of a good hash function?**
    - A) It should be easy to compute
    - B) It should use a lot of memory
    - C) It should produce many collisions
    - D) It should depend on the operating system
29. **What technique is used in open addressing to resolve collisions?**
    - A) Separate Chaining
    - B) Linear Probing

- C) Linked Lists
- D) Binary Search Trees

30. **What does double hashing use to eliminate clustering problems?**
    - A) A secondary hash function
    - B) A larger hash table
    - C) Linear search
    - D) Tree structures
31. **Analyze the following pseudocode using the Master Theorem:**

T(n) = 2T(n/2) + n

According to the Master Theorem, which case does this recurrence fall under, and what is the asymptotic complexity of T(n)?

- A) Case 1: $O(n\log n)$
- B) Case 2: $O(n)O(n)$
- C) Case 3: $O(n^2)$
- D) None of the above

32. **Which of the following is NOT a common operation on data structures?**
    - A) Traversal
    - B) Compilation
    - C) Insertion
    - D) Deletion
33. **What is the worst-case time complexity of QuickSort?**
    - A) $O(n\log n)$
    - B) $O(n^2)$
    - C) $O(n)$
    - D) $O(\log n)$
34. **Which sorting algorithm is stable and works in $O(n\log n)O(n\log n)$ time on average?**
    - A) QuickSort
    - B) MergeSort
    - C) HeapSort

- D) InsertionSort

35. **What is the height of a balanced binary search tree (BST) with nn nodes in the worst case?**

    - A) $O(\log n)$
    - B) $O(n)$
    - C) $O(n\log n)$
    - D) $O(1)$

36. **What is the main principle behind the divide and conquer approach?**

    - A) Merge smaller parts to form the final solution
    - B) Divide the problem into smaller subproblems, solve them independently, and combine their solutions
    - C) Use greedy methods to find an optimal solution
    - D) Solve the problem iteratively to reduce complexity

37. **Which of the following algorithms uses the divide and conquer strategy?**

    - A) Bubble Sort
    - B) Insertion Sort
    - C) Merge Sort
    - D) Radix Sort

38. **What is the time complexity of the merge operation in Merge Sort?**

    - A) $O(1)$
    - B) $O(\log n)$
    - C) $O(n)$
    - D) $O(n^2)$

39. **What is the worst-case time complexity of Merge Sort?**

    - A) $O(n\log n)$
    - B) $O(n^2)$
    - C) $O(n)$
    - D) $O(\log n)$

40. **What is the best-case time complexity of Quick Sort?**

- A) $O(n\log n)$
- B) $O(n^2)$
- C) $O(n)$
- D) $O(\log n)$

41. **In Quick Sort, what is the purpose of the partitioning step?**
    - A) To merge two sorted arrays
    - B) To divide the array into two subarrays based on a pivot element
    - C) To sort the elements in a single step
    - D) To find the median of the array
42. **What is the average-case time complexity of Quick Sort?**
    - A) $O(n\log n)$
    - B) $O(n^2)$
    - C) $O(n)$
    - D) $O(\log n)$
43. **Analyze the following recurrence relation using the Master Theorem:**

T(n) = 3T(n/2) + n

According to the Master Theorem, which case does this recurrence fall under, and what is the asymptotic complexity of T(n)?

- A) Case 1: $O(n^{\log 3})$
- B) Case 2: $O(n\log n)$
- C) Case 3: $O(n^2)$
- D) None of the above

44. **Consider the following pseudocode:**

```
Algorithm Sort(A, low, high)
  if low < high then
    pi = Partition(A, low, high)
    Sort(A, low, pi - 1)
    Sort(A, pi + 1, high)
```

What sorting algorithm does this pseudocode represent, and what is the worst-case time complexity?

- A) Merge Sort, $O(n\log n)$
- B) Quick Sort, $O(n^2)$
- C) Heap Sort, $O(n\log n)$
- D) Insertion Sort, $O(n^2)$

45. **Using the Master Theorem, analyze the recurrence relation:**

T(n) = 2T(n/2) + n \log n

Which case does this fall under, and what is the asymptotic complexity?

- A) Case 1: $O(n^2\log n)$
- B) Case 2: $O(n\log n)$
- C) Case 3: $O(n^2)$
- D) None of the above

46. **What is the key difference between Merge Sort and Quick Sort?**

- A) Merge Sort is a divide and conquer algorithm, while Quick Sort is not
- B) Merge Sort requires additional memory space, while Quick Sort does not
- C) Quick Sort guarantees $O(n\log n)$ time complexity, while Merge Sort does not
- D) Merge Sort works in-place, while Quick Sort does not

47. **What is the space complexity of Merge Sort?**

- A) $O(n)$
- B) $O(\log n)$
- C) $O(n\log n)$
- D) $O(1)$

48. **Which of the following best describes the partitioning strategy used in Quick Sort?**

- A) Divide the array into two equal halves
- B) Randomly select an element as pivot and rearrange the array so that elements less than pivot are on the left and elements greater than pivot are on the right
- C) Recursively merge sorted subarrays
- D) Use a linked list to partition elements

49. **Analyze the following recurrence relation using the Master Theorem:**

T(n) = T(n/2) + n

According to the Master Theorem, which case does this recurrence fall under, and what is the asymptotic complexity of T(n)?

- A) Case 1: $O(n\log n)$
- B) Case 2: $O(n)$
- C) Case 3: $O(n^2)$
- D) None of the above

50. **Which of the following scenarios would make Quick Sort perform poorly?**

- A) The pivot element is always the middle element
- B) The pivot element is always the largest or smallest element
- C) The array is already sorted
- D) The array contains all distinct elements

51. **Consider the recurrence relation for a divide and conquer algorithm:**

T(n) = 4T(n/2) + n^2

Using the Master Theorem, determine the asymptotic complexity of T(n).

- A) $O(n^2)$
- B) $O(n^3)$
- C) $O(n\log n)$
- D) $O(n\log n)$

52. **Which of the following is a drawback of using Merge Sort over Quick Sort?**

- A) Merge Sort is not a stable sort
- B) Merge Sort has a higher average-case time complexity
- C) Merge Sort requires additional memory space proportional to the input size
- D) Merge Sort is in-place

53. **What is the primary advantage of Quick Sort over Merge Sort?**

- A) Quick Sort is always faster
- B) Quick Sort has a guaranteed worst-case time complexity of $O(n\log n)$
- C) Quick Sort typically requires less additional memory space
- D) Quick Sort is more stable

54. **Using the Master Theorem, analyze the following recurrence relation:**

T(n) = 2T(n/2) + n

Which case does this fall under, and what is the asymptotic complexity?

- A) Case 1: $O(n)$
- B) Case 2: $O(n\log n)$
- C) Case 3: $O(n^2)$
- D) None of the above

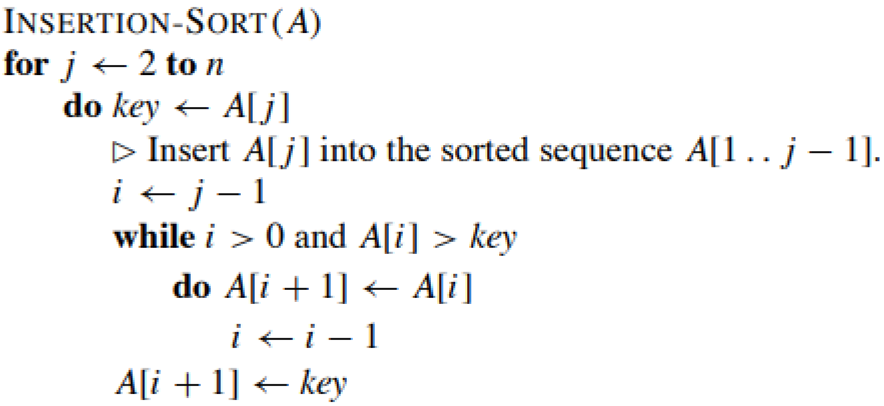
55. **Which sorting algorithm is most efficient for large datasets with random elements?**

- A) Quick Sort
- B) Bubble Sort
- C) Selection Sort
- D) Insertion Sort

56. Explain separate chaining collision resolution techniques.

57. Write the pseudo code Quick sort and what is the best and worst case analysis?

58. Find the analysis of insertion sort algorithm below



```
INSERTION-SORT(A)
for $j \leftarrow 2$ to $n$
    do $key \leftarrow A[j]$
       ▷ Insert $A[j]$ into the sorted sequence $A[1 \,..\, j-1]$.
       $i \leftarrow j - 1$
       while $i > 0$ and $A[i] > key$
           do $A[i+1] \leftarrow A[i]$
              $i \leftarrow i - 1$
       $A[i+1] \leftarrow key$
```

59. Sort the following elements using heap sort. Show each step.

2,5,-5,0 15, 6, 3, 9, 1, 8, 7, 8