 UNIT 4 – SEARCHING AND SORTING

(Questions 1–20)

Q1. Which of the following algorithms is NOT a searching algorithm?
a) Linear search
b) Binary search
c) Selection sort
d) Interpolation search

Q2. The time complexity of linear search in the worst case is _____.
a) O(1)
b) O(log n)
c) O(n)
d) O(n²)

Q3. Fill in the blank: Binary search can only be applied on _____ data.
a) Random
b) Unsorted
c) Sorted
d) Partially sorted

Q4. The best case time complexity of binary search is _____.
a) O(1)
b) O(n)
c) O(log n)
d) O(n log n)

Q5. Which of the following searching techniques works based on the position formula:
`pos = low + [(key - arr[low]) * (high - low) / (arr[high] - arr[low])]`?
a) Linear search
b) Binary search
c) Interpolation search

d) Fibonacci search

Q6. What is the space complexity of binary search (recursive version)?
a) O(1)
b) O(n)
c) O(log n)
d) O(n log n)

Q7. Which sorting algorithm repeatedly compares adjacent elements and swaps them if they are in wrong order?
a) Bubble sort
b) Insertion sort
c) Selection sort
d) Merge sort

Q8. Fill in the blank: In selection sort, the smallest element is placed at the _____ of the list in each pass.
a) Beginning
b) End
c) Middle
d) Random position

Q9. What is the time complexity of bubble sort in the best case (already sorted)?
a) $O(n^2)$
b) O(n log n)
c) O(n)
d) O(1)

Q10. Which sorting algorithm works by inserting each element into its correct position among previously sorted elements?
a) Bubble sort
b) Insertion sort
c) Selection sort
d) Merge sort

Q11. Fill in the blank: The merge sort algorithm is based on the _____ principle.

a) Divide and conquer
b) Decrease and conquer
c) Greedy method
d) Dynamic programming

Q12. Which of the following has the best average case time complexity?
a) Bubble sort
b) Selection sort
c) Insertion sort
d) Quick sort

Q13. What is the worst-case time complexity of quick sort?
a) O(n log n)
b) $O(n^2)$
c) O(log n)
d) O(1)

Q14. Fill in the blank: In quick sort, the element used to divide the array is called _____.
a) Pivot
b) Splitter
c) Divider
d) Marker

Q15. Which sorting algorithm is considered stable?
a) Quick sort
b) Merge sort
c) Selection sort
d) Heap sort

Q16. What is the time complexity of merge sort in the worst case?
a) O(n)
b) O(log n)
c) $O(n^2)$

d) O(n log n)

Q17. Fill in the blank: The number of passes required to sort n elements using bubble sort is _____.
a) n
b) n – 1
c) log n
d) n²

Q18. In insertion sort, if the input is already sorted, the number of swaps performed is _____.
a) n – 1
b) 0
c) n
d) log n

Q19. Which sorting algorithm is **non-comparative**?
a) Merge sort
b) Quick sort
c) Radix sort
d) Selection sort

Q20. What is the time complexity of radix sort if there are `n` elements and `d` digits per element?
a) O(n)
b) O(n log n)
c) O(dn)
d) O(n²)

 ANSWER KEY – UNIT 4 (Q1–Q20)

---

1. c — Selection sort is a sorting algorithm, not searching.
2. c — Linear search checks all n elements O(n).
3. c — Binary search requires sorted input.

4. a — Best case: middle element found at first step.
5. c — Interpolation search uses position formula.
6. c — Recursive version adds O(log n) stack space.
7. a — Bubble sort compares adjacent elements.
8. a — Selection sort places smallest at beginning.
9. c — Optimized bubble sort O(n) if sorted.
10. b — Insertion sort inserts in correct order.
11. a — Merge sort uses divide and conquer.
12. d — Quick sort usually fastest (O(n log n) avg).
13. b — Worst case when pivot always smallest/largest O(n²).
14. a — Pivot divides array into subarrays.
15. b — Merge sort maintains order stable.
16. d — Merge sort always O(n log n).
17. b — Bubble sort requires n–1 passes.
18. b — No swaps if already sorted.
19. c — Radix sort works on digits, not comparisons.
20. c — Radix sort = O(dn).

 END OF SET 1 (UNIT 4: QUESTIONS 1–20)

 UNIT 4 – SEARCHING AND SORTING

(Questions 21–40)

Q21. What will be the output of the following code?

```
int arr[] = {5, 2, 8, 3};
for (int i = 0; i < 3; i++) {
for (int j = 0; j < 3 - i; j++) {
if (arr[j] > arr[j + 1]) {
```

```
swap(arr[j], arr[j + 1]);
}
}
}
for (int i = 0; i < 4; i++) cout << arr[i] << " ";
```

a) 8 5 3 2
b) 2 3 5 8
c) 5 2 3 8
d) 3 2 5 8

Q22. The number of comparisons in the best case for binary search on an array of 16 elements is _____.
a) 1
b) 2
c) 4
d) log 16

Q23. Fill in the blank: The key characteristic of selection sort is that it performs _____ swaps in total.
a) n
b) n–1
c) log n
d) n^2

Q24. What will be the output of linear search for value 7 in the array [3, 5, 7, 9, 11]?
a) Index 2
b) Index 3
c) Index 4
d) Not found

Q25. Which searching algorithm is preferred when data is continuously changing?
a) Linear search
b) Binary search
c) Interpolation search

d) Fibonacci search

Q26. Fill in the blank: In bubble sort, after k passes, the last k elements are guaranteed to be _____.
a) Unsorted
b) In their correct positions
c) Randomly placed
d) Duplicated

Q27. The best case time complexity of insertion sort occurs when _____.
a) Array is in reverse order
b) Array is already sorted
c) All elements are same
d) Pivot chosen at random

Q28. What is the auxiliary space of merge sort?
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q29. Which sorting algorithm can be easily implemented using recursion?
a) Bubble sort
b) Quick sort
c) Insertion sort
d) Selection sort

Q30. Fill in the blank: The number of comparisons in selection sort for n elements is _____.
a) (n – 1)
b) n(n – 1)/2
c) n log n
d) n^2

Q31. Which sorting algorithm is considered the fastest for large datasets (average case)?

a) Quick sort
b) Bubble sort
c) Selection sort
d) Insertion sort

Q32. What is the time complexity of searching in a hash table (on average)?
a) O(1)
b) O(n)
c) O(log n)
d) O(n²)

Q33. Fill in the blank: In binary search, if the key is greater than mid element, the search continues in the _____ half.
a) Left
b) Right
c) Middle
d) Random

Q34. Which sorting algorithm divides the list into two halves and merges them after sorting?
a) Merge sort
b) Quick sort
c) Heap sort
d) Selection sort

Q35. What is the output of this pseudo-code?

```
arr = [4, 1, 3, 9];
selectionSort(arr);
```

Output: _____

a) 1 3 4 9
b) 4 3 1 9
c) 1 4 3 9

d) 3 1 4 9

Q36. Which sorting technique is called **in-place** because it requires no extra memory?
a) Merge sort
b) Quick sort
c) Counting sort
d) Bucket sort

Q37. Fill in the blank: The maximum number of comparisons in a binary search for n elements is _____.
a) log n
b) n
c) n log n
d) n^2

Q38. The partition step in quick sort divides the array into _____.
a) Two sorted halves
b) Two unsorted halves
c) Sorted and unsorted halves
d) Random halves

Q39. What is the average case time complexity of linear search?
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q40. Which of the following sorting algorithms is NOT stable?
a) Bubble sort
b) Insertion sort
c) Quick sort
d) Merge sort

✅ ANSWER KEY – UNIT 4 (Q21–Q40)
---

21. b — After bubble sort ➡ 2 3 5 8.
22. d — log₂ 16 = 4 comparisons (best case 1).
23. b — Selection sort performs n–1 swaps.
24. a — Value 7 found at index 2 (0-based).
25. a — Linear search works for changing data.
26. b — Each pass places largest at end.
27. b — Best when array already sorted ➡ O(n).
28. b — Merge sort uses O(n) auxiliary space.
29. b — Quick sort naturally recursive.
30. b — Selection sort makes n(n–1)/2 comparisons.
31. a — Quick sort average O(n log n).
32. a — Hashing average search = O(1).
33. b — If key > mid, search right half.
34. a — Merge sort divides and merges.
35. a — Selection sort output ➡ 1 3 4 9.
36. b — Quick sort modifies array in-place.
37. a — At most log₂ n comparisons.
38. b — Quick sort partitions into two unsorted subarrays.
39. b — Linear search scans half on average ➡ O(n).
40. c — Quick sort not stable by default.

✅ END OF SET 2 (UNIT 4: QUESTIONS 21–40)

✅ UNIT 4 – SEARCHING AND SORTING

(Questions 41–60)

Q41. What will be the array after the first pass of bubble sort on [5, 3, 8, 4, 2]?
a) [3, 5, 8, 4, 2]
b) [3, 5, 4, 2, 8]
c) [5, 3, 4, 2, 8]
d) [2, 3, 4, 5, 8]

Q42. Fill in the blank: Binary search reduces the search interval by a factor of _____ in each step.
a) 1/2
b) 1/3
c) 2
d) n

Q43. The number of comparisons required for linear search in the best case is _____.
a) n
b) 1
c) log n
d) n – 1

Q44. What is the time complexity of insertion sort in the worst case?
a) O(n)
b) O(n log n)
c) $O(n^2)$
d) O(1)

Q45. Which sorting algorithm divides the array into partitions based on a pivot element?
a) Merge sort
b) Quick sort
c) Insertion sort
d) Selection sort

Q46. Fill in the blank: The time complexity of bubble sort and selection sort in the worst case is _____.
a) O(n)
b) $O(n^2)$

c) O(log n)
d) O(n log n)

Q47. What will be the sorted output of insertion sort for the list [9, 5, 1, 4, 3]?
a) [1, 3, 4, 5, 9]
b) [9, 5, 1, 3, 4]
c) [5, 9, 1, 4, 3]
d) [1, 4, 3, 5, 9]

Q48. The stability of a sorting algorithm means _____.
a) Equal elements remain in same relative order
b) It uses no extra space
c) It runs in O(n log n)
d) It cannot handle duplicates

Q49. Which sorting algorithm does not perform well on linked lists?
a) Merge sort
b) Quick sort
c) Insertion sort
d) Bubble sort

Q50. Fill in the blank: In merge sort, the number of comparisons in merging two sorted lists of sizes m and n is at most _____.
a) $m \times n$
b) $m + n - 1$
c) $\log(m + n)$
d) $(m + n)^2$

Q51. What will be the array after the second pass of selection sort on [6, 3, 8, 5]?
a) [3, 6, 5, 8]
b) [3, 5, 6, 8]
c) [3, 8, 6, 5]
d) [3, 6, 8, 5]

Q52. Which searching algorithm does NOT require sequential access?
a) Linear search
b) Binary search
c) Sequential search
d) Sentinel search

Q53. Fill in the blank: The total number of comparisons in bubble sort for n elements is approximately _____.
a) $n^2 / 2$
b) n log n
c) n
d) n^2

Q54. What is the auxiliary space used in quick sort (iterative implementation)?
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q55. Which sorting algorithm is based on counting the occurrences of each element?
a) Bucket sort
b) Counting sort
c) Quick sort
d) Radix sort

Q56. Fill in the blank: In linear search, if the element is not found, the number of comparisons is _____.
a) n – 1
b) n
c) log n
d) 2n

Q57. Which of the following has the same time complexity for best, average, and worst cases?
a) Merge sort
b) Quick sort

c) Insertion sort
d) Selection sort

Q58. What is the time complexity of sorting an array using quick sort when all elements are identical?
a) O(1)
b) O(n)
c) O(n log n)
d) O(n²)

Q59. Fill in the blank: The minimum number of passes required to sort 6 elements using bubble sort is _____.
a) 6
b) 5
c) log n
d) 4

Q60. Which sorting algorithm is the most suitable for small datasets and nearly sorted data?
a) Merge sort
b) Quick sort
c) Insertion sort
d) Heap sort

 ANSWER KEY – UNIT 4 (Q41–Q60)
---

41. a — First pass moves largest to end [3,5,8,4,2].
42. a — Binary search halves the interval each step.
43. b — Best case: found at first position 1.
44. c — In reverse order O(n²).
45. b — Quick sort uses pivot-based partitioning.
46. b — Both have O(n²) worst case.
47. a — Sorted list after insertion sort: [1,3,4,5,9].
48. a — Stable = equal elements maintain order.

49. b — Quick sort not efficient for linked lists.
50. b — Merging takes ≤ m + n − 1 comparisons.
51. a — After 2 passes: [3,6,5,8].
52. b — Binary search can directly access mid element.
53. a — $(n^2 - n)/2 \approx n^2 / 2$.
54. c — Stack depth = O(log n).
55. b — Counting sort counts frequency of elements.
56. b — Must check all n elements if not found.
57. a — Merge sort = O(n log n) in all cases.
58. d — Identical elements worst partitioning $O(n^2)$.
59. b — Bubble sort n − 1 passes = 5.
60. c — Insertion sort good for small/nearly sorted data.

 END OF SET 3 (UNIT 4: QUESTIONS 41–60)

 UNIT 4 – SEARCHING AND SORTING

(Questions 61–80)

Q61. Which of the following algorithms is a **Divide and Conquer** algorithm?
a) Merge sort
b) Bubble sort
c) Selection sort
d) Insertion sort

Q62. Fill in the blank: Binary search is efficient because it eliminates _____ of elements in each step.
a) 1
b) 2
c) Half
d) All

Q63. Which sorting algorithm uses the concept of “minimum selection” in each pass?
a) Bubble sort
b) Selection sort
c) Merge sort
d) Insertion sort

Q64. What will be the array after the first partition step of quick sort with pivot = 6 for [8, 4, 7, 2, 6, 5]?
a) [4, 2, 5, 6, 8, 7]
b) [6, 4, 2, 5, 8, 7]
c) [2, 4, 5, 6, 7, 8]
d) [4, 2, 6, 8, 5, 7]

Q65. Fill in the blank: The time complexity of merge sort and quick sort (average case) is _____.
a) $O(n^2)$
b) O(n log n)
c) O(log n)
d) O(n)

Q66. Which sorting algorithm works by comparing and inserting elements into a partially sorted list?
a) Selection sort
b) Insertion sort
c) Bubble sort
d) Quick sort

Q67. The number of comparisons made in binary search for an array of 128 elements (worst case) is _____.
a) 7
b) 8
c) log 128
d) Both b and c

Q68. Fill in the blank: In bubble sort, if no swapping occurs in a pass, the array is _____.
a) Partially sorted

b) Fully sorted
c) Reversed
d) Random

Q69. Which searching algorithm has an average time complexity of O(log log n)?
a) Binary search
b) Linear search
c) Interpolation search
d) Fibonacci search

Q70. What is the main disadvantage of bubble sort?
a) Complex to implement
b) Requires extra memory
c) Slow for large data
d) Unstable algorithm

Q71. Fill in the blank: Quick sort performs poorly when pivot element is always the _____.
a) Middle element
b) Random element
c) Smallest or largest element
d) None

Q72. Which sorting algorithm can be easily adapted for singly linked lists?
a) Bubble sort
b) Merge sort
c) Quick sort
d) Selection sort

Q73. Which of the following is TRUE about the stability of sorting algorithms?
a) Merge sort is stable
b) Quick sort is stable
c) Selection sort is stable
d) Heap sort is stable

Q74. Fill in the blank: Counting sort and radix sort are examples of _____ sorting algorithms.
a) Comparison-based
b) Non-comparison-based
c) Recursive
d) Divide-and-conquer

Q75. What is the auxiliary space required for counting sort?
a) O(1)
b) O(n)
c) O(k) where k = range of input
d) O(n log n)

Q76. Which of the following best describes the working of radix sort?
a) Sorts elements based on digits or bits
b) Uses comparison of elements
c) Requires recursion
d) Works only on floats

Q77. Fill in the blank: The best case time complexity of quick sort occurs when the pivot divides the array into _____.
a) Unequal parts
b) Two equal halves
c) Random halves
d) Single elements

Q78. Which searching technique can be used efficiently on linked lists?
a) Binary search
b) Linear search
c) Interpolation search
d) Jump search

Q79. The number of swaps in selection sort for n elements is always _____.
a) n

b) n – 1
c) n log n
d) n²

Q80. Fill in the blank: Merge sort is preferred for large datasets because it has _____ time complexity in all cases.
a) O(n)
b) O(log n)
c) O(n²)
d) O(n log n)

 ANSWER KEY – UNIT 4 (Q61–Q80)
---

61. a — Merge sort follows Divide and Conquer.
62. c — Binary search removes half of remaining elements per step.
63. b — Selection sort picks minimum each pass.
64. a — First partition with pivot=6 [4,2,5,6,8,7].
65. b — Both merge and quick sort average at O(n log n).
66. b — Insertion sort inserts into partially sorted section.
67. d — log 128= 7 about 7-8 comparisons.
68. b — No swaps = already sorted.
69. c — Interpolation search has O(log log n) average time.
70. c — Bubble sort is inefficient for large datasets.
71. c — Poor pivot choice (min/max) worst O(n²).
72. b — Merge sort ideal for linked lists.
73. a — Merge sort is stable; others are not by default.
74. b — Counting & radix are non-comparison sorts.
75. c — Counting sort requires auxiliary array of size k.
76. a — Radix sort sorts digits one by one.
77. b — Best case = pivot divides array equally.
78. b — Linear search only practical for linked lists.
79. b — Selection sort always swaps n–1 times.

80. d — Merge sort consistently O(n log n).

 END OF SET 4 (UNIT 4: QUESTIONS 61–80)

 UNIT 4 – SEARCHING AND SORTING

(Questions 81–100)

Q81. Which sorting algorithm is the best choice for **large datasets with random order**?
a) Insertion sort
b) Merge sort
c) Quick sort
d) Bubble sort

Q82. Fill in the blank: Linear search is also called _____.
a) Sequential search
b) Indexed search
c) Hash search
d) Binary search

Q83. Which of the following sorting algorithms does **not** require comparisons between elements?
a) Counting sort
b) Selection sort
c) Quick sort
d) Merge sort

Q84. The space complexity of quick sort (recursive version) is _____.
a) O(1)
b) O(n)
c) O(log n)

d) $O(n^2)$

Q85. What is the maximum number of comparisons required in binary search on a sorted list of 1000 elements?
a) 10
b) 11
c) $\log_2(1000) \approx 10$
d) Both b and c

Q86. Fill in the blank: The main advantage of merge sort is its _____ performance on large datasets.
a) Inconsistent
b) Predictable
c) Slow
d) Unstable

Q87. Which of the following algorithms is **unstable** but **in-place**?
a) Merge sort
b) Quick sort
c) Bubble sort
d) Insertion sort

Q88. What will be the array after one pass of insertion sort on [5, 3, 4, 1, 2]?
a) [3, 5, 4, 1, 2]
b) [4, 5, 3, 1, 2]
c) [1, 3, 4, 5, 2]
d) [5, 4, 3, 2, 1]

Q89. Which searching technique is efficient for data stored in **linked lists**?
a) Linear search
b) Binary search
c) Jump search
d) Hashing

Q90. Fill in the blank: The process of arranging data items in logical order is called _____.

a) Searching
b) Sorting
c) Insertion
d) Deletion

Q91. The best case time complexity of bubble sort occurs when the array is _____.
a) Randomly ordered
b) Reverse ordered
c) Already sorted
d) Containing duplicates

Q92. Which algorithm is **recursive and non-stable**, but usually faster than merge sort for practical cases?
a) Bubble sort
b) Quick sort
c) Insertion sort
d) Selection sort

Q93. Fill in the blank: The time complexity of linear search in average case is _____.
a) O(1)
b) O(n)
c) O(log n)
d) $O(n^2)$

Q94. What is the primary advantage of counting sort over comparison-based algorithms?
a) Handles any type of data
b) Linear time for limited range inputs
c) Uses less memory
d) Simple recursive structure

Q95. Which of the following algorithms would be most efficient for sorting **student roll numbers (0–999)**?
a) Bubble sort
b) Radix sort
c) Quick sort

d) Selection sort

Q96. Fill in the blank: The number of levels in a binary search on n elements is approximately _____.
a) n
b) log n
c) √n
d) n log n

Q97. Which of the following statements about quick sort and merge sort is TRUE?
a) Quick sort uses extra space; merge sort is in-place
b) Merge sort uses extra space; quick sort is in-place
c) Both are in-place
d) Both require extra space

Q98. The average time complexity of bubble sort and selection sort are both _____.
a) $O(n^2)$
b) O(n log n)
c) O(n)
d) O(log n)

Q99. Which of the following is a real-life application of sorting algorithms?
a) Ranking students by marks
b) Searching files alphabetically
c) Database indexing
d) All of the above

Q100. Fill in the blank: The most suitable sorting algorithm for **external sorting (large files)** is _____.
a) Merge sort
b) Bubble sort
c) Insertion sort
d) Selection sort

 ANSWER KEY – UNIT 4 (Q81–Q100)
---

81. c — Quick sort performs best on large, random data.
82. a — Linear search = sequential search.
83. a — Counting sort compares no elements, uses counting logic.
84. c — Recursive quick sort uses O(log n) stack space.
85. d — log 1000 ≈ 10, so about 10–11 comparisons.
86. b — Merge sort gives consistent O(n log n) performance.
87. b — Quick sort is in-place but not stable.
88. a — After one pass [3, 5, 4, 1, 2].
89. a — Linked lists allow only sequential access linear search.
90. b — Sorting arranges data in order.
91. c — Already sorted array = best case for bubble sort.
92. b — Quick sort faster in practice but non-stable.
93. b — Average O(n) for linear search.
94. b — Counting sort = O(n + k) for small range values.
95. b — Radix sort ideal for fixed-digit integers.
96. b — Binary search depth ≈ log n.
97. b — Merge sort needs extra arrays; quick sort is in-place.
98. a — Both bubble & selection sort take O(n²).
99. d — Sorting used in all ranking and search systems.
100. a — Merge sort handles large external data efficiently.

 END OF SET 5 (UNIT 4: QUESTIONS 81–100)

 UNIT 4 COMPLETE — SEARCHING AND SORTING (100 QUESTIONS)

Topics Covered:
- Searching: Linear, Binary, Interpolation
- Sorting: Bubble, Selection, Insertion, Quick, Merge, Radix, Counting

- Complexity (Best, Average, Worst)
- Stability & Space trade-offs
- Real-world & Implementation-based questions