3. What will be the output if quickselect algorithm is applied to the array arr={1,5,4,3,7} with k given as 4?
a) 1
b) 3
c) 4
d) 5

Answer: d
Explanation: Quickselect algorithm finds the kth smallest element from the given list. So as here the given value of k is 4 so we need to find the fourth smallest element which is 5 in the given array.

4. What is the auxiliary space requirement of the quickselect algorithm?
a) $O(n^2)$
b) O(n)
c) O(n log n)
d) O(1)

Answer: d
Explanation: Quickselect algorithm requires no extra space in order to calculate the desired result. It performs manipulations in the given array itself so its auxiliary space requirement will be O(1).

5. Quickselect is an in-place algorithm?
a) true
b) false

Answer: a
Explanation: Quickselect's auxiliary space requirement is O(1). So quickselect qualifies as an in-place algorithm.

9. What is the average case time complexity of quickselect?
a) O(n log n)
b) $O(n^2)$
c) O(n)
d) O(log n)

Answer: c
Explanation: In quickselect, we don't recur for both portions of the array. Only that portion is considered where the smallest element lies. So this causes the average time complexity to be O(n).

11. Which of the following correctly represent the algorithm of quickselect?

a)

```
function quickSelect(list, left, right, k)
   if left = right
      return list[left]
   Select a pivotIndex between left and right
   pivotIndex := partition(list, left, right,  pivotIndex)
   if k = pivotIndex
      return list[k]
   else if k < pivotIndex
      right := pivotIndex - 1
   else
      left := pivotIndex + 1
```

Answer: a

Explanation: In quickselect algorithm if index of partitioned element is more than k, then we recur for left part. If index is same as k, we have found the kth smallest element and we return. If index is less than k, then we recur for right part.

Question 7 CORRECT

Which of the following is not true about comparison based sorting algorithms?

A. The minimum possible time complexity of a comparison based sorting algorithm is O(nLogn) for a random input array

B. Any comparison based sorting algorithm can be made stable by using position as a criteria when two elements are compared

C. Counting Sort is not a comparison based sorting algorithm

✓ Heap Sort is not a comparison based sorting algorithm.

Sorting **Analysis of Algorithms** **CountingSort** **HeapSort** **50 Algorithms MCQs with Answers**
Discuss it

Question 7 Explanation:
See http://www.geeksforgeeks.org/lower-bound-on-comparison-based-sorting-algorithms/ for point A. See http://www.geeksforgeeks.org/stability-in-sorting-algorithms/ for B. C is true, count sort is an Integer Sorting algorithm.

Question 11 CORRECT

You have to sort 1 GB of data with only 100 MB of available main memory. Which sorting technique will be most appropriate?

A. Heap sort

✓ Merge sort

C. Quick sort

D. Insertion sort

Sorting **QuickSort** **MergeSort** **HeapSort**
Discuss it

Question 11 Explanation:
The data can be sorted using external sorting which uses merging technique. This can be done as follows: 1. Divide the data into 10 groups each of size 100. 2. Sort each group and write them to disk. 3. Load 10 items from each group into main memory. 4. Output the smallest item from the main memory to disk. Load the next item from the group whose item was chosen. 5. Loop step #4 until all items are not outputted. The step 3-5 is called as merging technique.

Question 19 CORRECT

Which sorting algorithms is most efficient to sort string consisting of ASCII characters?

A Quick sort

B Heap sort

C Merge sort

✓ Counting sort

Sorting **QuickSort** **CountingSort** **HeapSort**
Discuss it

Question 19 Explanation:
Counting sort algorithm is efficient when range of data to be sorted is fixed. In the above question, the range is from 0 to 255(ASCII range). Counting sort uses an extra constant space proportional to range of data.

Question 22 WRONG

Given an array where numbers are in range from 1 to n^6, which sorting algorithm can be used to sort these number in linear time?

A Not possible to sort in linear time

✓ Radix Sort

✗ Counting Sort

D Quick Sort

Sorting **QuickSort** **RadixSort** **CountingSort**
Discuss it

Question 22 Explanation:
See Radix Sort for explanation.

Question 35 WRONG

If we use Radix Sort to sort n integers in the range $(n^{k/2},n^k]$, for some k>0 which is independent of n, the time taken would be?

A Θ(n)

X Θ(kn)

✓ Θ(nlogn)

D $\Theta(n^2)$

Gate IT 2008 RadixSort Sorting Analysis of Algorithms 50 Algorithms MCQs with Answers
Discuss it

Question 35 Explanation:
Radix sort time complexity = O(wn)
for n keys of word size= w
=>w = $\log(n^k)$
O(wn)=O(klogn.n)
=> kO(nlogn)

Question 49 CORRECT

You are given a sequence of n elements to sort. The input sequence consists of n/k subsequences,each containing k elements. The elements in a given subsequence are all smaller than the elements in the succeeding subsequence and larger than the elements in the preceding subsequence. Thus, all that is needed to sort the whole sequence of length n is to sort the k elements in each of the n/k subsequences. The lower bound on the number of comparisons needed to solve this variant of the sorting problem is

A Ω (n)

B Ω (n/k)

✓ Ω (nlogk)

D Ω (n/klogn/k)

UGC-NET CS 2017 Nov - III Sorting
Discuss it

Question 49 Explanation:
We have n/k subsequences each containing k elements. If we use the quick sort technique to sort k elements in a subsequence then the complexity of sorting k elements of a subsequence is klogk. And we have n/k such sequences. Then the time complexity of n/k subsequences having k elements will be: = (n/k)* klogk = nlogk

Question 52 CORRECT

Which of the following algorithms sort n integers, having the range 0 to (n^2 - 1), in ascending order in O(n) time ?

A Selection sort

B Bubble sort

✓ Radix sort

D Insertion sort

UGC NET CS 2015 Jun - II **Sorting** **50 Algorithms MCQs with Answers**
Discuss it

Question 52 Explanation:

- Selection sort takes $O(n^2)$ time.
- Bubble sort takes $O(n^2)$ time.
- Radix sort takes $O(n)$ time.
- Insertion sort takes $O(n^2)$ time.

So, option (C) is correct.

1. Which of the following is not a stable sorting algorithm?

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

7. The time complexity of heap sort in worst case is

a) O(logn)
b) O(n)
c) O(nlogn)
d) $O(n^2)$

11. Time complexity of bubble sort in best case is

a) θ (n)
b) θ (nlogn)
c) θ (n^2)
d) θ $(n(\log n)^2)$

14. Which of the following sorting algorithms is/are stable

a) Counting sort
b) Bucket sort
c) Radix sort
d) All of the above

14. Which of the following sorting algorithms is/are stable

a) Counting sort
b) Bucket sort
c) Radix sort
d) All of the above

Answer: c
Explanation: Time complexity of counting sort is given as O(n+k) where n is the number of input elements and k is the range of input. So if range of input is not significantly larger than number of elements in the array then it proves to be very efficient.

6. Which of the following sorting techniques is stable?
a) quick sort
b) counting sort
c) heap sort
d) selection sort

Answer: b
Explanation: Counting sort is an example of stable sorting algorithm as the elements with identical values appear in the same order in the output array as they were in the input array.

7. Which of the following uses the largest amount of auxiliary space for sorting?
a) Bubble sort
b) Counting sort
c) Quick sort
d) Heap sort

Answer: b
Explanation: Counting sort requires auxiliary space of O(n+k) whereas quick sort, bubble sort and heap sort are in place sorting techniques. Thus counting sort requires most auxiliary space.

1. How many comparisons will be made to sort the array arr={1,5,3,8,2} using counting sort?
a) 5
b) 7
c) 9
d) 0

Answer: d
Explanation: As counting sort is an example of non comparison sort so it is able to sort an array without making any comparison.

11. Counting sort is often used as a sub routine for radix sort.
a) true
b) false

Answer: a
Explanation: Counting sort is used as a sub routine for radix sort as it is a stable and non comparison based sorting algorithm.

14. What is the disadvantage of counting sort?
a) counting sort has large time complexity
b) counting sort has large space complexity
c) counting sort is not a comparison based sorting technique
d) counting sort cannot be used for array with non integer elements

Answer: d
Explanation: Counting sort can only be used for arrays with integer elements because otherwise array of frequencies cannot be constructed.

3. LSD radix sort requires _____ passes to sort N elements.
a) (w/logR)
b) N(w/logR)
c) (w/log(RN))
d) (wN/log(N))

Answer: a
Explanation: LSD radix sort sorts the N elements in (w/logR) passes where w is the number of digits in largest number and R(radix) is extra space required for performing the sorting operation.

6. LSD radix sort is faster than comparison sorts.
a) True
b) False

Answer: b
Explanation: LSD radix sort is faster than comparison sorts when the word size is less than logn. But LSD radix sort runs slowly for elements with larger word size and smaller radix.

7. Which of the following should be used to sort a huge database on a fixed-length key field?
a) Insertion sort
b) Merge sort
c) LSD radix sort
d) Quick sort

Answer: c
Explanation: LSD radix requires only w passes to sort a fixed-length string, where w is a length of the strings. So, LSD radix sort is best suited to sort a huge database on a fixed-length key field.

Which of the following method is taken different operations different charges in amortized analysis?

- Aggregate method
- accounting method ✓
- potential method
- both (A) and (B) ✗

NEXT QUESTION

Which of the following method is computing total cost of an algorithm in amortized analysis?

- Aggregate method
- accounting method
- potential method ✓
- None

NEXT QUESTION

which of the following method is credit as the potential energy to pay for future operations?

- Aggregate method
- accounting method ✗
- potential method
- both (A) and (B) ✓

NEXT QUESTION

6. Which of the following is not a noncomparison sort?

a) Counting sort
b) Bucket sort
c) Radix sort
d) Shell sort

21. Time complexity to sort elements of binary search tree is

a) O(n)
b) O(nlogn)
c) $O(n^2)$
d) $O(n^2 \log n)$

22. The lower bound on the number of comparisons performed by comparison-based sorting algorithm is

a) Ω (1)
b) Ω (n)
c) Ω (nlogn)
d) Ω (n^2)

26. For merging two sorted lists of size m and n into sorted list of size m+n, we require comparisons of

a) O(m)
b) O(n)
c) O(m+n)
d) O(logm + logn)