ARICENT: First Mile Foundation Program

Quiz 4 Solutions

For questions, refer to the Quiz page. Only the solutions are given below

1. **In a party of N people, a celebrity walks in. Everyone knows the celebrity but the celebrity knows no one. Non-celebrities may/may not know anyone in the room. Assuming we can only ask each person A questions like "do you know B?", what is the minimum number of questions to find the celebrity?**
   A. N
   B. 2N
   C. N^2
   D. (N+1)*(N+1)

   Answer : A
   Steps to find the celebrity:
   - Make them stand all in a row,
   - Compare first two individuals say (*a*,*b*)
     - If *a* knows *b* => *a* is certainly not the celebrity, then probable celebrity is *b* otherwise the probable celebrity is *a*.
   - Eliminate the improbable one, repeat above steps till a individual remains.

   Since in each comparison an individual is eliminated and have others as the probable celebrity, we will have N question to be asked.

2. **Consider the following code segment. Assume n to be a positive integer.**

```
for (i = 1; i <= n; i++) {
  for (j = 1; j <= n; j = j+i) {
     printf("Hi");
  }
}
```

   Let T(n) represent the total running time of the above code segment. Specify which of the following statements are correct.
   A. T(n)=O(n^2)
   B. T(n)=O(n*log n)
   C. T(n)≠O(n^4)
   D. T(n)=O(log n)

Answer : B

In the i-th iteration, the inner loop runs O(n/i) times, and therefore, the overall complexity is O(n+n/1+n/2+...+1), and this is O(nlogn).

3. **In the mergesort algorithm, what is the asymptotic running time of the step of merging sorted subarrays?**
   A. O(log n)
   B. O(n)
   C. O(n log n)
   D. O(n^2)

   Answer : B

   Merging two sorted arrays will require linear scan on each sub-array.

4. **What is the efficient asymptotic running time to find the median of a sorted array of size N?**
   A. O(n)
   B. O(log n)
   C. O(1)
   D. O(n logn)

   Answer : C

   If array is *odd* sized pick the middlemost. If *even*, pick the two middlemost and average. Either way, this can be done in constant number of steps, so O(1).

5. **A list of n strings, each of length at most n, is sorted in lexicographic order using the merge-sort algorithm. Which of the following gives the worst-case complexity of this? Hint: Account for the cost of comparing two strings also.**
   A. $O(n^2)$
   B. O(n log n)
   C. $O\ (n^2 + \log n)$
   D. $O(n^2 \log n)$

   Answer : D

   Merge sort requires O(n log n) comparisons. So for string of length k, a string comparison at most requires O(k). Therefore sorting k length string will take $O(k^2 \log k)$.

6. **What is the degree of the polynomial defined by**
   R(x) = P(x) * Q(x) where P(x) = x^3+x^2+1 and Q(x) = x^2+ x + 1?

   Answer : 5

The degree of a polynomial is the highest degree of its terms. In R(x), the term with the highest degree will we obtained by multiplying x^3 of P(x) and x^2 of Q(x). So, the degree is 5.

7. **Which among the following sorting algorithm will take least time when all elements of input are identical?**
   A. Insertion Sort
   B. Merge Sort
   C. Both would take same time.

   Answer : A
   The insertion sort will take O(n) while merge sort takes O(n log n) when input array is already sorted.

8. **Given a sorted array of integers, what can be the worst case complexity of efficient algorithm to find is the smallest element present in array which is greater than or equal to x. ?**
   For example, if the given array is {10, 30, 50, 100, 200, 299} and x = 105, then output should be 200.
   A. O(nlogn)
   B. O(logn*logn)
   C. O(n)
   D. O(log n)

   Answer : D
   Algorithm for finding the smallest element in array which is greater than or equal to x can be done using modified binary search approach which takes O(log n) as shown below.

```
FindCeil(A[],low,high,x)
1. mid = index of middle element of A[low..high]
2. If x ==A[mid], then return mid
3. If x > A[mid], then either A[mid + 1] is ceil of x or ceil lies in A[mid+1…high]
4. If x < A[mid], then either A[mid] is ceil of x or ceil lies in A[low .. mid-1]
```

9. **For the following array and the specified search key, what will be the number of iterations performed by binary search?**
   array = {-6, 3, 4, 7, 9, 10, 11, 15, 16, 18, 19, 22, 25 , 33, 47}, search key = 12

   Answer : 4
   1st comparison : {-6, 3, 4, 7, 9, 10, 11, **15**, 16, 18, 19, 22, 25, 33, 47}
   2nd comparison : {-6, 3, 4, **7**, 9, 10, 11}
   3rd comparison : {9, **10**, 11}
   4th comparison : {**11**}

   Binary search would exit failing to find '12' as there are no more elements to search. The answer therefore is 4.

10. **Can linear search take lesser number of comparisons than binary search to find an element in an array?**
    A. Yes
    B. No

    Answer : A
    A linear search takes only one comparison to search for the **first** element whereas with binary search it is at least 1 (1 when the size of an array is 1, > 1 otherwise).

11. **Given an array of n elements that is sorted in ascending order, what is the worst case time complexity to find the maximum element in the array?**
    A. O(n)
    B. B. O(n^2)
    C. C. O(1)
    D. D. O(log n)

    Answer : C
    As array is already sorted, maximum/minimum will be first/last element of the list which can be computed in linear time.

12. **The tightest lower bound on the number of comparisons, in the worst case, for any comparison-based sorting algorithm is of the order of:**
    A. O(n)
    B. B. O(nlogn)
    C. C. O(1)
    D. D. O(n^2)

    Answer : B
    The number of comparisons that a comparison based sort algorithm requires is given by O(nlogn), where n is the number of elements to sort.

13. **Which of the following recurrence relationships is applicable for the runtime of merge sort on an array of n elements?**
    A. T(n)=T(n-2)+O(n)
    B. T(n)=T(n-1)+O(n)
    C. T(n)=2T(n/2)+O(n)
    D. T(n)=3T(n/3)+O(n)

    Answer : C

In merge sort every problem is subdivided into two problems of equal size which is later merged at cost of O(n).

$$T(n) = 2\,T(n/2) + O(n)$$

14. **Assume that a mergesort algorithm in the worst case takes 30 seconds for an input of size 64. Which of the following most closely approximates the maximum input size of a problem that can be solved in 6 minutes?**
    - **A. 256**
    - **B. 512**
    - **C. 1024**
    - **D. 2048**

Answer : B

Time complexity of merge sort is Θ(n log n).

➔ c*64 log 64 = 30

➔ c*64*6 = 30

➔ c = 5/64

For 6 minutes,

➔ 5/64*n logn = 6*60

➔ n logn = 72*64 = 512 * 9

➔ n = 512.

15. **Let the numbers 794, 332, 561, 342, 200, 607, and 893 be sorted using radix sort. What will be the sixth number in the sequence of numbers after sorting the second digit?**

Answer : 893

Given numbers are 794, 332, 561, 342, 200, 607, and 893.

Sorting on the least significant digit :

200, 561, 332, 342, 893, 794, 607

Sorting on the next significant digit :

200, 607, 332, 342, 561, **893**, 794

Hence the answer is **893**.