SET - D

SRM Institute of Science and Technology
College of Engineering and Technology
School of Computing

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamil Nadu

Academic Year: 2023-24 (EVEN)

Test: CLA-T1 | **Date: 19.02.2024**
Course Code & Title: 21CSC204J Design and Analysis of Algorithms | **Duration:** 1 hour 40 min
Year & Sem: II Year / IV Sem | **Max. Marks:** 50

Course Articulation Matrix:

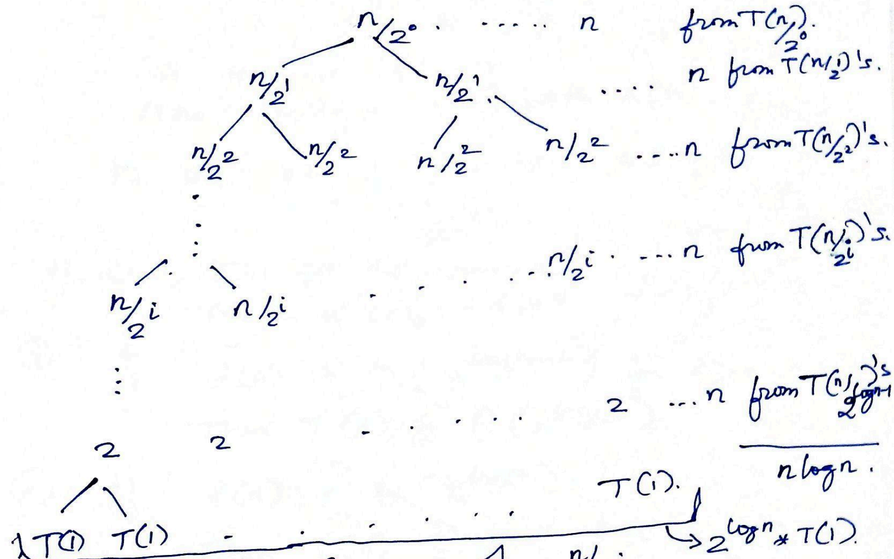
Course Outcome	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	Program Specific Outcomes		
													PSO-1	PSO-2	PSO-3
CO1	2	1	2	1	-	-	-	-		3	-	3	3	1	-
CO2	2	1	2	1	-	-	-	-		3	-	3	3	1	-
CO3	2	1	2	1	-	-	-	-		3	-	3	3	1	-
CO4	2	1	2	1	-	-	-	-		3	-	3	3	1	-
CO5	2	1	2	1	-	-	-	-		3	-	3	3	1	-

Part - A

(1 x 10 = 10 Marks) **Instructions: Answer all**

Q. No	Question	Marks	BL	CO	PO	PI Code
1	Correctness of algorithm means a) Algorithm works in linear time b) Algorithm uses linear space c) Algorithm is efficient d) **Algorithm behaves as expected**	1	1	1	1	1.6.1
2	The objective of time complexity analysis is a) To determine the development time of algorithm b) To determine the compile time of algorithm **c) To determine the running time of algorithm** d) To determine the complexity of conditions in an algorithm	1	1	1	1	1.6.1
3	Asymptotic notations deal with a) Running time for small values of input b) Running time for medium values of input **c) Running time for large values of input (tending to ∞)** d) Running time for inputs in the range 1-100	1	1	1	12	1.6.1
4	Which among these is both an asymptotically tight lower and upper bound ? **a)** O(n) b) o(n) c) ω (n) d) $\Theta(n)$ Answer: d) $\Theta(n)$	1	1	2	3	1.7.1

5	$f(n) = n^2$ is a) $O(n^2)$ b) $O(\log log\ \ n)$ c) $O(n)$ d) $O(1)$ Answer: a) $O(n^2)$	1	1	2	**3**	1.2.1
6	The worst case running time of binary search is a) $O(n)$ b) $O(n^2)$ c) $O(\log n)$ d) $O(1)$ Answer: c) $O(log\ n)$	1	1	2	**3**	1.7.1
7	Which among the following is NOT an application of divide and conquer strategy? **a) Bubble sort** b) Binary Search c) Merge sort d) Strassen's matrix multiplication	1	1	2	**3**	1.7.1
8	Partition algorithm on an array of size n takes running time a) $O(n)$ b) $O(n^2)$ c) $O(\log log\ \ n)$ d) $O(1)$ Answer: a) $O(n)$	1	1	2	**3**	1.7.1
9	In order to divide the closest pair problem into subproblems, we need to compute **a) Median x-coordinate** b) Min x-coordinate c) Mean x-coordinate d) Mode of x-coordinates	1	1	2	**3**	1.7.1
10	Which technique can be used to obtain the asymptotic running time for a recursive algorithm by direct application without solving recurrences? **a) Master theorem** b) Substitution method c) Recursion tree d) Mathematical Induction	1	1	2	**3**	1.7.1
Part – B **(5 x 4 = 20 Marks) Instructions: Answer All the Questions**						
11	Prove that $3n^2 + 2n + 5 \in \Theta(n^2)$ Solution $3n^2+2n+5 \leq 3n^2+2n^2+5n^2 \quad \forall n>0.$ $= 10n^2$ Also, $3n^2+2n+5 \geq 3n^2 \geq 0 \ \forall n>0.$ $\therefore 0 \leq 3n^2 \leq 3n^2+2n+5 \leq 10n^2 \ \forall n \geq 1 = 1$ $c_1=3, c_2=10, c_1, c_2>0.$ $\therefore 3n^2+2n+5 \in \Theta(n^2) \quad \square$	5	2	1	**1**	1.2.1
12	Solve $T(n) = 2T\left(\frac{n}{2}\right) + n$ using recursion tree method (Assume T(1) = 1).	5	2	1	**1**	1.2.1

<table>
<tr><td></td><td>Solution.


There are 2^i nodes of $n/2^i$
∴ There are $2^{\log n-1}$ nodes of $n/2^{\log n-1} = 2$.
∴ There are $2^{\log n}$ (= n) leaf nodes of T(1).
∴ Total = $n * \log n + 2^{\log n} * T(1) = n\log n + n$.
[Assumed that n is a power of 2 above] □</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>13</td><td>Illustrate Partition algorithm used in quicksort for the following array: 31, 43, 22, 15, 54, 17, 44, 29, 38 (use first element as pivot)

Soln.

31 43 22 15 54 17 44 29 38
pivot p ↑l >p ↑r <p
interchange.

31 29 22 15 54 17 44 43 38
↑l >p ↑r <p.
interchange.

31 29 22 15 17 54 45 43 38.
↑r <p. ↑l >p.

(No interchange l > r)
Now interchange A[0] with A[r].

17 29 22 15 31 54 45 43 38.
□</td><td>5</td><td>2</td><td>2</td><td>3</td><td>2.6.2</td></tr>
</table>

14	State Master theorem and demonstrate with an example. If $a \geq 1$, $b > 1$ for the recurrence $T(n) = aT(n/b) + f(n)$ ① If $f(n) = O(n^{\log_b a - \epsilon})$ for $\epsilon > 0$ Then $T(n) \in \Theta(n^{\log_b a})$. ② If $f(n) = \Theta(n^{\log_b a})$ Then $T(n) \in \Theta(n^{\log_b a} \log_2 n)$ ③ If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for $\epsilon > 0$ and $a f(n/b) \leq c f(n)$ for some $c > 0$ and sufficiently large n, Then $T(n) \in \Theta(f(n))$ Example. $T(n) = 2T(n/2) + n$. $a = 2, b = 2, f(n) = n$. $f(n) = n \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 2}) = \Theta(n^1)$. $\therefore$ Using case 2, we get $T(n) = \Theta(n^{\log_b a} \log n)$ $= \Theta(n \log n)$	5	2	2	1	1.2.1

Part – C

(2 x 10 = 20 Marks) **Instructions: Answer All the Questions**

15. A	Consider the algorithm below: **Algorithm F(A, lo, hi)** **Begin** **If hi = lo return A[lo]** **Mid = floor((hi + lo)/2)** **X1 = F(A, 1, mid)** **X2 = F(A, mid+1, hi)** **Return(X1 + X2)** **End** (i) Summarize what algorithm F is doing? (ii) Devise recurrence relations for running time of algorithm F (Assume inputs which are powers of 2). (iii) Compute the asymptotic time complexity using substitution method.	10	3	1	2	2.6.2

	15A (i) Computing the sum of array A from index lo to index hi. (2 marks) (ii) $T(n) = \begin{cases} c_1 & \text{if } n=1 \\ 2T(n/2) + c_2 & \text{for } n > 1. \end{cases}$ (2 marks) (iii) $T(n) = 2T(n/2) + c_2$ $= 2[2T(n/4) + c_2] + c_2.$ $= 2[2[2T(n/8) + c_2] + c_2] + c_2$ So, $T(n) = 2T(n/2^1) + c_2$ $= 2^2 T(n/2^2) + c_2[1+2]$ (6 marks) $= 2^3 T(n/2^3) + c_2[1+2+4]$ In general $T(n) = 2^i T(n/2^i) + c_2[1+2+\dots+2^{i-1}]$ — (1) Setting $i = \log n$, $n/2^i = n/2^{\log n} = n/n = 1$ and $2^{\log n} = n$. $\therefore T(n) = 2^{\log n} T(n/2^{\log n}) + c_2[1+2+\dots+2^{\log n - 1}]$ $= n T(1) + c_2(1)\left[\frac{2^{\log n}-1}{2-1}\right] = c_1 n + c_2(n-1) \in O(n)$ or $\Theta(n)$.					
	OR					
15. B	Vimal works with a detective agency and is need of processing suspect details in alphabetical order quickly. He has two processes running simultaneously (a) a sorting algorithm that will sort details of suspects and provide them in sorted order (b) a background process that will process the next suspect in sorted order without waiting for entire sorted list. He is in need of completing the task non-recursively and quickly. (i) Recommend the appropriate sorting algorithm for this task. (ii) Compute the time complexity of this algorithm. (iii) List the best case, average case, worst case time complexity of this algorithm.	10	3	1	2	2.6.4

<table>
<tr>
<td></td>
<td>
15 B.

(i) Bubble sort or Selection Sort because every successive iteration produces another (the next) element in the sorted list. (2 marks)

(ii)
BubbleSort(A)   Frequency
(1) for i = 1 to A.length − 1 n
(2)   for j = A.length downto i+1 $\sum_{i=1}^{n-1}(n-\overline{i+1}+2)$
(3)   if A[j] < A[j−1] $\sum_{i=1}^{n-1}(n-\overline{i+1}+1)$
(4)   exchange A[j] with A[j−1] $max = \sum_{i=1}^{n-1}(n-\overline{i+1}+1)$

If c_i is the time to execute step i, then
$T(n) = c_1 n + c_2\sum_{i=1}^{n-1}(n-\overline{i+1}+2) + c_3\sum_{i=1}^{n-1}(n-\overline{i+1}+1)$
$+ c_4\sum_{i=1}^{n-1}(n-\overline{i+1}+1)$
$= c_1 n + c_2\sum_{i=1}^{n-1}(n-i+1) + (c_3+c_4)\sum_{i=1}^{n-1}(n-i)$. (6 marks).
$= c_1 n + (c_2+c_3+c_4)\sum_{i=1}^{n-1} n - (c_2+c_3+c_4)\sum_{i=1}^{n-1} i$
$+ (c_2+c_3+c_4)\sum_{i=1}^{n-1} 1$
$= c_1 n + (c_2+c_3+c_4)n(n-1) - (c_2+c_3+c_4)\frac{n(n-1)}{2}$
$+(c_2+c_3+c_4)(n-1)$.
$= (c_2+c_3+c_4)\frac{n(n-1)}{2} + (c_1+c_2+c_3+c_4)n$
$- (c_2+c_3+c_4)$.

(iii) Best case, average case and worst case − $O(n^2)$.
Note: Best case for bubble sort is $O(n^2)$, not $O(n)$.

Partial Marks for insertion sort may be given.
</td>
<td></td><td></td><td></td><td></td><td></td>
</tr>
<tr>
<td>16. A</td>
<td>Ashiq provides an online sorting service for his customers. He is afraid that some malicious users will send sorting sequences that will take a long time to sort.
(i) Analyze which sorting algorithm should he use for best sorting performance? Justify your answer.
(ii) Explain the algorithm for the same.
(iii) Illustrate how the algorithm will work for the following array 37, 12, 8, 35, 27, 39, 18, 25.</td>
<td>10</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2.6.4</td>
</tr>
</table>

16A

(i) Bubble sort, insertion sort – bad sorting performance of $O(n^2)$

QuickSort – worst case $O(n^2)$, so malicious users can exploit this to cause bad performance

∴ Merge sort is best sorting algo – $O(n \log n)$ in worst case (3 marks)

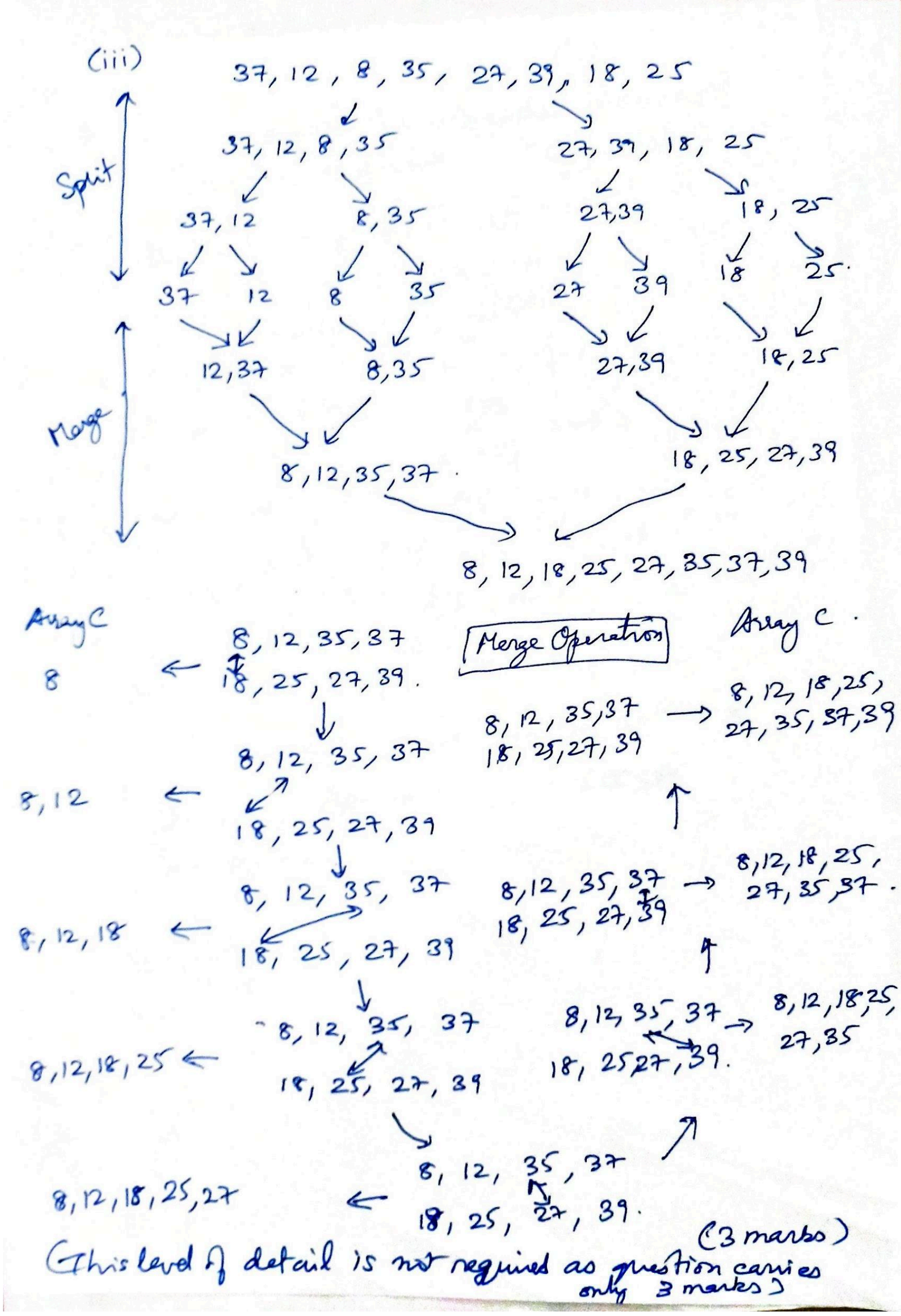
(ii) Mergesort Algorithm.

Algorithm Mergesort (A, low, high).
Begin
    if low < high
        mid = ⌊(low + high)/2⌋
        MergeSort (A, low, mid)
        MergeSort (A, mid+1, high)
        Merge (A, low, mid, high)
end.

Merge Algorithm:
Parameters : A, low, mid, high

1. i = low, j = mid+1.
2. if A[i] < A[j], copy A[i] into next position of C
   i = i+1
   else copy A[j] into next position of C.
   j = j+1
3. Repeat step 2 until i > mid or j > high.
4. if i > mid, copy A[j] to A[high] to next location in C
   else copy A[i] to A[mid] to next location in C
5. Copy C back to A[low...high] (4 marks)

(This level of detail is not required as this part carries only 4 marks)



Partial Marks for quicksort may be given

	OR					
16. B	Victory Supermarket has introduced an innovative incentive mechanism for its sales representatives. Each sales representative has to make Rs X sales in a day. If he or she makes Y sales, then she gets Y – X points for that day. The incentive for the month is determined by the largest continuous cumulative streak of points (LCCSP) bagged by the sales representative over the month obtained by summing up a sequence of consecutive values. (i) Devise an algorithm to compute the LCCSP for each sales representative in a month. (ii) Illustrate how the LCCSP will be computed for the following sequence: 13, 8, 14, 12, 17, 21, 13, 16, 5 the sales representative is required to make Rs.15/- each day.	**10**	**3**	**2**	2	**2.5.3**

	16B. (i) Largest Subarray Sum Algorithm. Summarized Algorithm is sufficient (5 marks). (ii) After subtracting 15, we get −2, −7, −1, −3, 2 | 6, −2, 1, −10. l=2 (5,5) | r=6 (6,6). −2, −7, −1 | −3, 2 a=8 (5,6) 6, −2, 1 | −10. l=−1 (3,3) | r=2 (5,5) l=6 | r=−10 −2, −7 | −1 a=−2 a=5 −10 (9,9). −3 | 2 6, −2 | 1 l=−2 | r=−1. l=−3 | r=2. l=6 | r=1 a=−8 −3 (4,4) 2 (5,5) a=5 −2 | −7 −1 (3,3) a=−1 6 | −2 1 (8,8) l=−2 | a=−7. l=6 | r=−2 (5 marks) −2 (1,1) −7 (2,2) 6 (6,6) a=4 −2 (7,7) a=−9. Answer: largest subarray sum = 8. l − left subarray = LCSP r − right subarray a − across.