Started on	Saturday, 17 February 2024, 12:16 PM
State	Finished
Completed on	Saturday, 17 February 2024, 12:35 PM
Time taken	18 mins 23 secs
Marks	7.00/7.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1

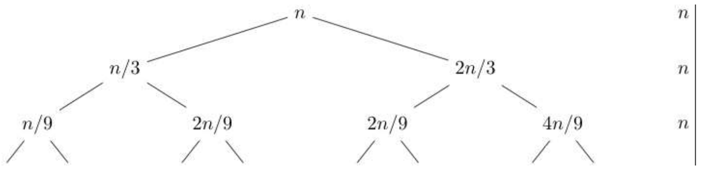
Correct

Mark 1.00 out of 1.00

Recursion Tree is one way to analyze recursive functions. Consider a function with following time complexity.

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

Following figure shows the first 3 levels of the recursion tree.



What is/are the number(s) which can not be appear in the next (4th) level in this recursion tree?

- a. 8n/27
- b. 16n/27 ✔
- c. 2n/27
- d. n/27

The correct answer is: 16n/27

Question 2

Correct

Mark 1.00 out of 1.00

Which is not a method for analyzing time complexity of recurrences?

- a. Substitution Method
- b. Recurrence Tree Method
- c. Amortized Method ✔
- d. Master Method

The correct answer is: Amortized Method

Question 3

Correct

Mark 1.00 out of 1.00

Given a set 'S' of n integers and another integer x, an algorithm should determine whether or not there exists two elements in S whose sum is exactly x . A possible algorithm for this task is described below.

1) Sort the elements in S using any efficient sorting algorithm.

2) Remove the last element from S. Let y be the value of the removed element.

3) If S is non-empty, look whether an element z exist in S where z=x-y

4) If S contains such an element z, then stop, since we have found y and z such that x=y+z; otherwise repeat Step 2.

5) If S is empty, then no two elements in S sum to x.

Select the correct statement(s) regarding above approach.

- [x] a. Time complexity of this algorithm is Θ(nlg n). ✔
- [ ] b. Best time complexity to do Step 3 is Θ(n) .
- [x] c. Step 1 can be acheived through merge sort with Θ(nlg n) time complexity. ✔
- [ ] d. There are algorithms which can solve this task with better time complexity than above described algorithm

The correct answers are: Step 1 can be acheived through merge sort with Θ(nlg n) time complexity., Time complexity of this algorithm is Θ(nlg n).

Question 4

Correct

Mark 1.00 out of 1.00

Select the asymptotic upper and lower bounds for T(n) in the following recurrence. Assume that T(n) is constant for $n \leq 3$. Make your bounds as tight as possible.

$T(n) = T(n-2) + \log(n)$

Select one:

- a. $T(n) = \Theta(n)$
- b. $T(n) = \Theta(\log(n))$
- c. $T(n) = \Theta(n^2)$
- d. $T(n) = \Theta(n\log(n))$ ✔

Your answer is correct.

The correct answer is:
$T(n) = \Theta(n\log(n))$

Question 5

Correct

Mark 1.00 out of 1.00

The solution to the recurrence $T(n) = 3T(n/3) + O(\lg n)$ is $T(n) = \Theta(n \lg n)$.

Select one:

- True
- False ✔

False.

Case 3 of the master theorem applies:

$f(n) = O(n^{\log 3 \text{ (base3)}}) = O(n)$ for $f(n) = O(\lg n)$, hence, $T(n) = O(n)$.

The correct answer is 'False'.

Question 6

Correct

Mark 1.00 out of 1.00

For the following recurrence, select the correct expression for runtime T(n) if the recurrence can be solved using Master Theorem, Otherwise, indicate that the Master Theorem does not apply.

T(n) = 16T(n/4) +n

- a. T(n) = $\Theta(n^2 \log(n))$
- b. Master Theorem does not apply.
- c. T(n)=$\Theta(n^2)$ ✔
- d. T(n) =nlog(n)

The correct answer is: T(n)=$\Theta(n^2)$

Question 7

Correct

Mark 1.00 out of 1.00

Solve the following Recursive Algorithm:

$T(n) = \{^{1 \; if \; n=1}_{2T(\frac{n}{2})+F'(n) \;\; if \; n>1}$

Note: $F'(n)$ function is in the order of $O(n)$

- a. $T(n) = O(n)$
- b. $T(n) = O(nlog(n))$ ✔
- c. $T(n) = O(n^2)$
- d. $T(n) = O(log(n))$

The correct answer is: $T(n) = O(nlog(n))$