# CSPB 3104 - Park - Algorithms

<u>Dashboard</u> / My courses / <u>2241:CSPB 3104</u> / <u>22 January - 28 January</u> / <u>Quiz 2</u>

Started on	Monday, 22 January 2024, 9:38 PM
State	Finished
Completed on	Monday, 22 January 2024, 9:42 PM
Time taken	3 mins 12 secs
Marks	12.00/12.00
Grade	<b>10.00</b> out of 10.00 ( <b>100</b> %)

Correct

Mark 4.00 out of 4.00

Use case-1 of master theorem to find an upper bound on each of the recurrences.

#### Recall Case-1

Consider recurrences of the form  $T(n) = aT(\frac{n}{b}) + n^c$  where  $c < \log_b(a)$  then the master method shows that  $T(n) = \Theta(n^{\log_b(a)})$ .

- (A)  $T(n) = 4T(\frac{n}{2}) + 3n$ . Select from one of the options below:
- $\Theta(n)$
- $\bigcirc \Theta(n^3)$
- $\Theta(n^2)$   $\checkmark$  Correct
- $O(n \log(n))$
- $\bigcirc O(n\sqrt{n})$
- Case-1 of the theorem does not apply

Mark 1.00 out of 1.00

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

- (B)  $T(n) = T(\frac{n}{3}) + n$ . Select from one of the options below:
- $\Theta(n)$
- $\Theta(1)$
- Case-1 of the master theorem does not apply here

  ✓ Correct
- $\Theta(n\sqrt{n})$

Mark 1.00 out of 1.00

The correct answer is: Case-1 of the master theorem does not apply here

- (C)  $T(n) = T(n-3) + T(n-4) + \frac{n}{2}$ . Select from one of the options below:
- $\Theta(\sqrt{n})$
- $\Theta(n^5)$
- Master theorem does not apply here
   Correct
- $\bigcirc \Theta(n \log(n))$

Mark 1.00 out of 1.00

The correct answer is: Master theorem does not apply here

- (D)  $T(n) = 8T(\frac{n}{7}) + \sqrt{n}$  Let  $\alpha = \log_7 8$ .
- $\Theta(n)$
- $\Theta(\sqrt{n})$
- $\Theta(n^{\alpha})$   $\checkmark$  Correct

Mark 1.00 out of 1.00

The correct answer is:  $\Theta(n^{\alpha})$ 

Correct

Marks for this submission: 4.00/4.00.

Correct

Mark 2.00 out of 2.00

Recall Case-2 of the master method:

Consider recurrences of the form

$$T(n) = aT(\frac{n}{b}) + \Theta(n^c)$$
 where  $c = \log_b(a)$ .

then the master method shows that  $T(n) = \Theta(n^{\log_b(a)} \log n)$ .

In this assignment, we will consider applying case-2 of the master method to solve recurrences. Answer the questions below:

(a) Consider the recurrence

$$T(n) = 9T(\frac{n}{3}) + 3n^2 + 2n + 3\log(n).$$

Which of the following is true for its closed form?

- $\Theta(n^2)$
- Case 2 of master method cannot apply here
- $\Theta(n^2 \log(n))$   $\checkmark$  Correct
- $\Theta(n \log(n))$
- $\Theta(\log(n)^2)$

Mark 1.00 out of 1.00

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

(b) Consider the recurrence

$$T(n) = T(\frac{n}{3}) + 1.$$

Which of the following is true for its closed form?

- $\Theta(\log(n))$   $\checkmark$  Correct
- $\Theta(n \log(n))$
- $\Theta(n)$
- Case 2 of Master's theorem is inapplicable

Mark 1.00 out of 1.00

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

Correct

Marks for this submission: 2.00/2.00.

Correct

Mark 1.00 out of 1.00

Using the Master method, what is the solution for the below recurrence relation --

$$T(n) = 4 T(n/2) + 5 * (n^2)$$

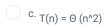
Select the "tightest" bound on T(n) from the available choices below.

Select one:



• a. 
$$T(n) = \Theta(n^2 \log n)$$





Your answer is correct.

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

Correct

Marks for this submission: 1.00/1.00.

Correct

Mark 1.00 out of 1.00

Using the Master method, what is the solution for the below recurrence relation -

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

Select one:

$$C. T(n) = \Theta (n^{(\log 2/\log 3)})$$

• d. 
$$T(n) = \Theta(n^{(\log 3/\log 2)})$$

Your answer is correct.

The correct answer is:  $T(n) = \Theta(n^{(\log 3/\log 2)})$ 

Correct

Marks for this submission: 1.00/1.00.

Correct

Mark 1.00 out of 1.00

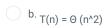
Using the Master method, what is the solution for the below recurrence relation –

$$T(n) = 3 T(n/3) + (n^3)$$

Select one:



• a. 
$$T(n) = \Theta(n^3)$$



C. T(n) = 0 (n^3)
-------------------

Your answer is correct.

The correct answer is:  $T(n) = \Theta(n^3)$ 



Marks for this submission: 1.00/1.00.

### Question 6

Correct

Mark 3.00 out of 3.00

Consider the divide-and-conquer approach -- MergeSort -- for sorting array with size m = 2^n. For each iteration, you

Divide: Divide the problem into 2 halves.

Conquer: Sort the two divided arrays using MergeSort recursively. If the arrays are of size 1,

Combine: Merge the sorted halves into a single array.

How many times will you perform the "divide" step in the procedure for the given array? (Each division into two sub-problems counts as 1.)

**Note:** This problem requires some precise counting. You should draw a "recursion tree" for merge sort and sum up the number of divisions carefully to arrive at the answer.

# Select one:

- a. 2<sup>(n-1)</sup>
- **b.** n
- c. 2<sup>n</sup>



Your answer is correct.

The correct answer is: (2^n)-1

Correct

Marks for this submission: 3.00/3.00.