Time complexity notations and master theorem

Due No due date **Point Allowed attempts** Unlimited

Points 5 **Questions** 5

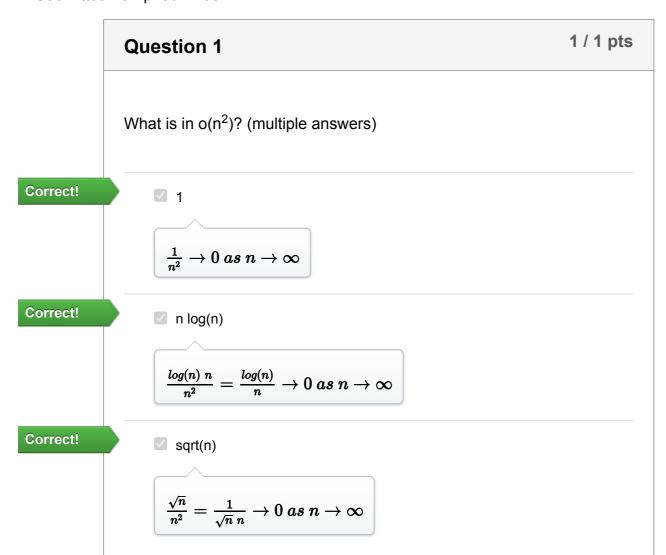
Time limit None

Take the quiz again

Attempt history

	Attempt	Time	Score	
KEPT	Attempt 2	3 minutes	5 out of 5	
LATEST	Attempt 2	3 minutes	5 out of 5	
	Attempt 1	27 minutes	2 out of 5	

Submitted 19 Apr at 22:56



	□ log(n) n^2		
	□ n^3		
	□ n^2		
Correct!	✓ n		
	$rac{n}{n^2}=rac{1}{n} o 0\ as\ n o \infty$		
	Question 2	1 / 1 pts	
	Which of the following is true? (multiple answers)		
Correct!	if $f(n)=o(n)$ and $g(n)=o(n)$ then $f(n)+g(n)=o(n)$		
	$rac{f(n)}{n} ightarrow 0 \ as \ n ightarrow \infty \ and \ rac{g(n)}{n} ightarrow 0 \ as \ n ightarrow \infty,$	$hence \; rac{f(n)+g(n)}{n} o 0 \; a$	is $n o \infty$
	O(n) is the complexity in the worst case		
	if $f(n)=\Omega(g(n))$ and $f(n)=\Omega(h(n))$, then $g(n)<=h(n)$	n)) for all n>0	
	if $f(n)=O(n)$ then $f(n)=o(n)$		
Correct!	if f(n)=\Theta(g(n)) then g(n)=\Theta(f(n))		
	Yes, by the definition of \Theta.		
	Question 3	1 / 1 pts	

Prove that functions $T_1(n) = T_1(n/2) + n$, and $T_2(n) = 2T_2(n/2) + 2$ are of

the same order.

	Select this order.	
Correct!		
	O log(n)	
	n log(n)	
	○ n^2	
	O 1	
	Use master theorem.	
	Question 4	1 / 1 pts
	Quodion 4	
	We considered Binary search which halves our sold recursively searches in one of the halves. Now imagine we want instead split our sorted array then continue searching in one of 3 parts. Let's call algorithm Ternary search. What is the computational complexity of Ternary search.	rted arrays and then y into 3 parts and I this recursive
	We considered Binary search which halves our sor recursively searches in one of the halves. Now imagine we want instead split our sorted array then continue searching in one of 3 parts. Let's call algorithm Ternary search.	rted arrays and then y into 3 parts and I this recursive
	We considered Binary search which halves our sor recursively searches in one of the halves. Now imagine we want instead split our sorted array then continue searching in one of 3 parts. Let's call algorithm Ternary search. What is the computational complexity of Ternary search.	rted arrays and then y into 3 parts and I this recursive
	We considered Binary search which halves our sor recursively searches in one of the halves. Now imagine we want instead split our sorted array then continue searching in one of 3 parts. Let's call algorithm Ternary search. What is the computational complexity of Ternary search.	rted arrays and then y into 3 parts and I this recursive
	We considered Binary search which halves our sol recursively searches in one of the halves. Now imagine we want instead split our sorted array then continue searching in one of 3 parts. Let's call algorithm Ternary search. What is the computational complexity of Ternary search.	rted arrays and then y into 3 parts and I this recursive

Question 5 1 / 1 pts

What is the time complexity of this function?

```
// Input: array and its size.
int f (vector<int> inputArray, int n) {
  if (n < 2) return 1;
// resize n/2 leaves only n/2 first elements.
return f(inputArray.resize(n/2), n/2) + 7*n + n*n + 11;
}</pre>
```

Correct!

- Theta(log(n))
- \Theta(n^2)
- \Theta(n)
- \Theta(n log(n))

Apply master theorem. a=1, b=2, d=0 (because of the constant number of operations in each function call).