New chat

Not seeing what you expected here? Don't worry, your conversation data is preserved! Check back soon.

Upgrade to Plus NEW

Dark mode

Updates & FAQ

Log out

CS2023 - Data Structures and
Algorithms
Take Home Assignment
Week 2 Complexity Analysis
March 09, 2023
You are required to answer the below questions and submit a PDF to the submission link provided under this week before the deadline (no extensions will be provided). You can either write / type your answers, but either way your answers should be readable.
Question 1
Study the Little Oh notation (o), Big Omega Notation(Ω) and Little Omega Notation(ω).
Note.: You should define. what these symbols mean and provide examples if possible.
Question 2
What are the relationships between the θ, O, o, and ω notations?
Note.: Please use a table to compare and contrast them
Question 3
Study the optimized Bubble sort algorithms given in the next page and
1. Analyze them for worst case time complexity.
2. Is there a difference in worst case time complexities?
3. Is there an easy method to analyze only the worst- c a s e time complexity?
1
Algorithm 1 Bubble Sort Optimizec

Regenerate response


```
if (A|i — 1|> A[i]) then
temp = A|i|
A|i| = A|i — 1|
A|i — 1| = temp
swapped = true
if (! swapped) then
break;
n = newLimit
Algorithm 2 Bubble Sort Optimized - Version II
n = A.length
do
swapped = false
for i = 2 to n do
if (A|i — 1| >A[i]) then
temp = A|i|
A|i| = A|i - 1|
A|i - 1| = temp
swapped = true
newLimit = i-1
```

Regenerate response

Question 1

Little Oh notation (o):
The little oh notation is used to describe the upper bound of the growth rate of a function. It is used to denote that a function f(n) grows strictly slower than g(n) as n approaches infinity. Mathematically, we say f(n) = o(g(n)) if for any positive constant c, there exists a value n0 such that f(n) < c*g(n) for all n > n0.

For example, if f(n) = n^2 and g(n) = n^3, then we can say f(n) = o(g(n)) because n^2 grows strictly slower than n^3 as n approaches infinity.

Big Omega Notation (Ω):
The big omega notation is used to describe the lower bound of the growth rate of a function. It is used to denote that a function f(n) grows at least as fast as g(n) as n approaches infinity. Mathematically, we say f(n) = Ω(g(n)) if for any positive constant c, there exists a value n0 such that f(n) > c

Regenerate response

The little omega notation is used to describe the lower bound of the growth rate of a function, but strictly faster than g(n). It is used to denote that a function f(n) grows strictly faster than g(n) as n approaches infinity. Mathematically, we say f(n) = ω(g(n)) if for any positive constant c, there exists a value n0 such that f(n) > c*g(n) for all n > n0.

For example, if f(n) = n^3 and g(n) = n^2, then we can say f(n) = ω(g(n)) because n^3 grows strictly faster than n^2 as n approaches infinity.

Question 2

The table below shows the relationships between the θ, O, o, and ω notations.

Notation	Definition	Upper Bound	Lower Bound	Tight Bound
O	f(n) is	Yes	No	No
	bounded			
Ω	f(n) is	[illegible]	[illegible]	No
ω	f(n) is	No	Yes	No
	tightly			
o	f(n) is	Yes	No	No
	tightly			

- The big O notation (O) is an upper bound on the growth rate of a function.
- The big omega notation (Ω) is a lower bound on the growth rate of a function.
- The theta notation (θ) is a tight bound on the growth rate of a function.
- The little omega notation (ω) is a lower bound on the growth rate of a function, but strictly faster than g(n).
- The little oh notation (o) is an upper bound on the growth rate of a function, but strictly slower than g

Regenerate response

ChatGPT Feb 13 Version. Free Research Preview. Our goal is to make AI systems more natural and safe to interact with. Your feedback will help us improve.