Question 8.

Algorithms Assignment 1

Faaiq Bilal
23100104

1 Part 1

This can be disproved with a counter-example. If we take $f(n) = n$ and $g(n) = n^2$
Then the first condition is satisfied as $n < c \cdot n^2$ for all $n > 0$ and $c \geq 1$
However, the converse is not necessarily true, as n^2 grows faster than $c \cdot n$.
Hence, in this case, $g(n) \notin f(n)$

2 Part 2

The proof of this statement can be expressed in two parts:

2.1 Proof part 1, Big O

For this, we can start off by supposing that $f(n) > g(n)$. This supposition doesn't actually matter, it only helps us in denoting one function as being larger than the other. The roles could very easily have been swapped.

In such a case, the following must also be true for any constant $c_0 \geq 2$:
$c \cdot f(n) \geq f(n) + g(n)$

This proves that $f(n) + g(n) \ \epsilon \ max(f(n), g(n))$

2.2 Proof part 2, Big Omega

This part is much simpler to prove, as function $f(n)$ and $g(n)$ grow with increasing n.
This would mean that $f(n) + g(n)$ must be larger than both $f(n)$ and $g(n)$ individually. It does not matter which of them is larger.
Hence, we can say that $f(n) + g(n) \geq max(f(n), g(n))$
$f(n) + g(n) \ \epsilon \ max(f(n), g(n))$
$f(n) + g(n) \ \epsilon \ \Omega(max(f(n), g(n)))$

2.3 Conclusion

As both statements above are true, then $f(n) + g(n) \ \epsilon \ \theta(max(f(n) + g(n)))$

3 Part 3

This statement is true as shown below:

$f(n) = O(g(n))$ can also be written as: $f(n) \leq M \cdot g(n)$, where M is some constant > 0.
This statement can be extended as follows: $lg(f(n)) \leq lg(M \cdot g(n))$
$lg(f(n)) \leq (lgM + lg(g(n)))$
As $lg(g(n)) > lgM$ for a sufficiently large n (as $n \rightarrow$ inf)
This would mean that $lg(f(n)) \leq 2 \cdot lg(g(n))$
Hence, $lg(f(n)) \leq k \cdot lg(g(n))$, where k ≥ 2.

4 Part 4

This statement can be disproved with a counter example
If $f(n) = 2n$ and $g(n) = n$, then $2n \ \epsilon \ O(n)$
And then we try proving for $2^{2n} \leq c \cdot 2^n$
This gives us $2^n \leq c$
This is not possible as $n \rightarrow$ inf
Hence, by contradiction we know that the statement is False

5 Part 5

This statement can be disproved with a counter example
When $f(n) = \frac{1}{n}$, then $\frac{1}{n} \leq c \cdot \frac{1}{n^2}$
This gives us $c \geq n$, which is not possible as n grows to be infinitely large.

It should be noted that this statement would be true if the function f was a monotonically increasing function. $f(n) = \frac{1}{n}$ is not a monotonically increasing function.

6 Part 6

$f(n) = O(g(n))$ implies that $f(n) \leq c \cdot g(n)$
This means that $g(n) \geq c \cdot f(n)$ which directly leads us to $g(n) \ \epsilon \ \Omega(f(n))$
Hence the statement is true

7 Part 7

This claim can be disproved with an example.
Consider the function $f(n) = 3^n$ and the fact that θ implies O and Ω to be true as well.
When trying to assert this case for the upper bound, we run into the following problem:
$3^n \leq c \cdot 3^{\frac{n}{2}}$
$3^{\frac{n}{2}} \leq c$
This cannot be true, as n grows to be infinitely large and will always overtake the constant term c.

8 Part 8

$\Omega(f(n))$ roughly represents a quantity that is larger than $f(n)$.
We will now consider the case for both the upper bound and the lower bound, as we have to prove that the function holds the tightest bound.

Proving for the lower bound: As $\Omega(f(n))$ is larger than $f(n)$, and $\Omega(f(n)) > 0$, we know that for $f(n) + \Omega(f(n)) \geq c \cdot f(n)$ where c is some constant > 0. This condition is fine.

Proving for the upper bound: as $\Omega(f(n))$ represents a function that is of a higher bound than $f(n)$, we cannot guarantee that there exists a constant c for which $f(n) + \Omega(f(n)) \leq c \cdot f(n)$. Hence, the condition is not satisfied.

Conclusion: the function does not maintain its tightest bound to be the same as the tightest bound of $f(n)$.