Jiawei Wu, collaborated with Jericho Dizon and Jose Idrovo

Q1 (20 points; 2 per part) DPV Problem 0.1 (parts a through j): In each of the following situations, indcate whether $f = O(g)$, or $f = \Omega(g)$, or both (in each case $f = \theta(g)$).

	$f(n)$	$g(n)$	**answer**
(a)	$n - 100$	$n - 200$	$f = \theta(g)$
(b)	$n^{\frac{1}{2}}$	$n^{\frac{2}{3}}$	$f = O(g)$
(c)	$100n + log\ n$	$n + (log\ n)^2$	$f = \theta(g)$
(d)	$n\ log\ n$	$10n\ log\ 10n$	$f = \theta(g)$
(e)	$log\ 2n$	$log\ 3n$	$f = \theta(g)$
(f)	$10\ log\ n$	$log(n^2)$	$f = \theta(g)$
(g)	$n^{1.01}$	$nlog^2\ n$	$f = \Omega(g)$
(h)	$\frac{n^2}{log\ n}$	$n(log\ n)^2$	$f = \Omega(g)$
(i)	$n^{0.1}$	$(log\ n)^{10}$	$f = \Omega(g)$
(j)	$(log\ n)^{log\ n}$	$\frac{n}{log\ n}$	$f = \Omega(g)$

Q2 (10 points) Consider the following pseudo-code which takes the integer $n \geq 0n \geq 0$ as input:

```
Function bar(n)
    Print '*';
    if n == 0 then
        Return;
    end
    for i = 0 to n - 1 do
        bar(i);
    end
```

Let $T(n)$ be the number of times the above function prints a star $(*)$ when called with input $n \geq 0$. What is $T(n)$ exactly, in terms of only n (and not values like $T(n-1)$ or $T(n-2)$)? Prove your statement

Solution $= \mathbf{2^n}$

n	calls	$T(n)$
0	print	$T(0) = 1$
1	print, $T(0)$	$T(1) = 2$
2	print, $T(0)$, $T(1)$	$T(2) = 4$
3	print, $T(0)$, $T(1)$, $T(2)$	$T(3) = 8$

We deduct that $T(n) = 2^n$

We can prove the claim by induction. Our recurrence formula given by the function is $T(n) = 2T(n-1)$ for $T(0) = 1$.

Base Case: $n = 0$

$$T(0) = 1$$
$$1 = 2^0$$
$$1 = 1$$

Induction step: prove $2^n = 2T(n-1)$ for $n = k+1$

$$\boxed{2^{k+1}} = 2T(k+1-1)$$

Work on RHS

$$2T(k) = 2 \times 2^k$$
$$2 \times 2^k = \boxed{2^{k+1}}$$

Thus proving that our recurrence formula generates the given solution 2^n. ■

Q3 (30 points) Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of θ-notation, prove that $max(f(n), g(n)) = \theta(f(n) + g(n))$

Basic definition of the θ notation: If $max(f(n), g(n)) = \theta(f(n) + g(n))$, then it must satisfy $max(f(n), g(n)) = O(f(n) + g(n))$ and $max(f(n), g(n)) = \Omega(f(n) + g(n))$.

We must prove $max(f(n), g(n)) = O(f(n) + g(n))$ and $max(f(n), g(n)) = \Omega(f(n) + g(n))$.

Case 1: Prove $max(f(n), g(n)) = \Omega(f(n) + g(n))$

We know the following:

$$max(f(n), g(n)) \geq f(n) \text{ if max = g(n)}$$
$$max(f(n), g(n)) \geq g(n) \text{ if max = f(n)}$$

The outcome must of one of them.

Therefore, we can derive the following equation by adding both sides:

$$2 \times max(f(n), g(n)) \geq f(n) + g(n)$$
$$max(f(n), g(n)) \geq \frac{1}{2}(f(n) + g(n))$$

where $C = \frac{1}{2}$. Since $max(f(n), g(n)) \geq C(f(n) + g(n))$ for some constant C, we prove that $max(f(n), g(n)) = \Omega(f(n) + g(n))$

Case 2: Prove $max(f(n), g(n)) = O(f(n) + g(n))$

We know that $max(f(n), g(n)) \leq f(n) + g(n)$. If max is $f(n)$, $f(n) + g(n)$ could be the same value (in the case that $g(n) = 0$), or larger than the max. The same applies if max is $g(n)$. Since this is true, a constant $C > 0$ can be placed on the RHS, rendering the following equation:

$$max(f(n), g(n)) \leq C(f(n) + g(n))$$

By assuming that $C = 1$, which is a value in $C > 0$, the function remains the same. Our function remains the same as our first statement, which we know to be true, therefore we prove that $max(f(n), g(n)) = O(f(n) + g(n))$.

With the basic definition of θ notation, we have proved that $max(f(n), g(n)) = \theta(f(n)+g(n))$■

Q4 (10 points; 5 per part)

(a) is $2^{2n} = O(2^n)$?

No

(b) why?

n^a dominates n^b if $a > b$. In this instance, no matter the value for n, $2n$ will always dominate n (unless negative, but that doesn't apply here). In addition, 2^{2n} can be rewritten as $(2^2)^n$, therefore it is in $O(4^n)$.