NO.

DATE

Data Structure and Algorithm Analysis (H) Lab 2.

Q2.1

a) $3n^2+5n-2 = \tilde{\Theta}(n^2)$.

since: $0 < 3n^2 \leq 3n^2+5n-2 \leq 10n^2$, $\forall n \geq 1$.

b) $42 = \tilde{\Theta}(1)$

since: $0 < 40 \leq 42 \leq 44$, $\forall n \geq 1$.

c) $4n^2(1+\log n)-2n^2 = \tilde{\Theta}(n^2 \log n)$.

since: $0 < 4n^2 \log n \leq 4n^2(1+\log n)-2n^2 \leq 6n^2 \log n$, $\forall n \geq 4$.

Q2.2

(a)

$f(n)$	$g(n)$	O	o	Ω	ω	$\tilde{\Theta}$
$\log(n)$	$\sqrt{n}$	yes	yes	no	no	no
n	$\sqrt{n}$	no	no	yes	yes	no
n	$n \log n$	yes	yes	no	no	no
n^2	$n^2+(\log n)^3$	yes	no	yes	no	yes
2^n	n^3	no	no	yes	yes	no
$2^{\frac{n}{2}}$	2^n	yes	yes	no	no	no
$\log_2 n$	$\log_{10} n$	yes	no	yes	no	yes.

Q2.3

Algorithm A:

1.	foo	1
2.	for $i=1$ to n do	
3.	for $j=1$ to $n-2$ do	
4.	foo	$n \times (n-2)$
5.	foo	$n \times (n-2)$
6	foo.	$n \times (n-2)$

# of foo operation: $1+n \times (n-2) \times 3 = 3n^2-6n+1 = \tilde{\Theta}(n^2)$

since $0 < 2n^2 \leq 3n^2-6n+1 \leq 3n^2$, $\forall n \geq 10$.

Algorithm B:

1.	foo	1
2.	for $i=1$ to n do	
3.	foo	n
4.	for $i=1$ to $\frac{n}{2}$ do	
5.	foo	$\frac{n}{2}$
6	foo.	$\frac{n}{2}$

NO.

DATE

# of foo operations: $1 + n + \frac{n}{2} + \frac{n}{2} = 2n + 1 = \Theta(n)$.

Since: $0 < 2n \leq 2n+1 \leq 3n, \ \forall n \geq 1$.

Algorithm C:

1.	foo	1
2.	for $i=1$ to n do	
3.	for $j=1$ to i do	
4.	foo	$1+2+\cdots+n = \frac{n(n+1)}{2}$
5.	foo	n
6.	foo	1

# of foo operations: $1 + \frac{1}{2}n^2 + \frac{1}{2}n + n + 1 = \frac{1}{2}n^2 + \frac{3}{2}n + 2 = \Theta(n^2)$

Since: $0 < \frac{1}{2}n^2 \leq \frac{1}{2}n^2 + \frac{3}{2}n + 2 \leq n^2, \ \forall n \geq 10$.

Q 2.9

1. True.

Proof. $\forall f(n) \in O(\sqrt{n})$, $\exists c > 0$, $n_0 \in \mathbb{N}^*$, s.t.

$0 < f(n) \leq c\sqrt{n}, \ \forall n \geq n_0$.

For such c and n_0:

$0 < f(n) \leq c\sqrt{n} \leq cn, \ \forall n \geq n_0$

$\Rightarrow$ by def. $f(n) = O(n)$. $\square$

2. False.

Counter Example: $\lim_{n \to \infty} \frac{n}{n^2} = 0. \Rightarrow n = o(n^2)$.

$\Rightarrow$ For LHS, choose n for $O(n^2)$.

$LHS = n + n = 2n$.

Suppose $2n = \omega(n) \Rightarrow n = o(2n)$.

i.e $\lim_{n \to \infty} \frac{n}{2n} = 0$. but in fact $\lim_{n \to \infty} \frac{n}{2n} = \frac{1}{2}$.

$\Rightarrow$ a contradiction $\lightning$.

3. True.

Proof: $\forall f(n) \in O(n)$, $\exists c > 0$, $n_0 \in \mathbb{N}^*$, s.t

$0 < f(n) \leq cn, \ \forall n \geq n_0$.

$\Rightarrow 0 < f(n) \leq cn \leq cn \log n, \ \forall n \geq \max\{n_0, 2\}$.

$\Rightarrow 0 < 3n \log n + f(n) \leq (c+3) n \log n, \ \forall n \geq \max\{n_0, 2\}$.

$\Rightarrow 3n \log n + f(n) = O(n \log n)$.

i.e. $3n\log n + O(n) = O(n\log n)$ ---- ①

On the other hand, $0 \le 3n\log n \le 3n\log n + f(n)$, $\forall n \ge 1$.

$\Rightarrow 3n\log n + f(n) = \Omega(n\log n)$

i.e. $3n\log n + O(n) = \Omega(n\log n)$ ---- ②

Combining ① and ②, $3n\log n + O(n) = \sim(n\log n)$. □

Explanation about the meaningless statement:

* "at least" implies a lower bound on the running time.
* "$O(n^2)$", however represents an upper bound.

⇒ the saying „at least $O(n^2)$" is illogical.
We should use „Ω" or „ω" for a lower bound.

Q2.5

```
Matrix-Multiply (A,B)
1. for i=1 to n do                                   ~(n).
2.     for j=1 to n do                               ~(n) · ~(n) = ~(n²)
3.         C[i,j] := 0                               ~(n²)
4.         for k=1 to n do                           ~(n²) · ~(n) = ~(n³)
5.             C[i,j] := C[i,j] + A[i,k] · B[k,j]    ~(n³)
6. return C.                                         ~(n²)
```

→ $\sim(n) + \sim(n^2) + \sim(n^2) + \sim(n^3) + \sim(n^3) + \sim(n^2)$

$= \sim(\max\{n, n^2, n^3\})$

$= \sim(n^3)$.

NO.

DATE

Q.2.6

1. Loop invariant: "At the start of each iteration of the for loop of lines 2~4, A[j] is the smallest element in the subarray A[j, ..., A.length]"

* Initialization: A[A.length] is the smallest element of A[A.length] trivially. (j = A.length).

* Maintenance: The inner loop "bubbles" a small element to the left hand side of the array. It makes sure A[j-1] ≤ A[j]. Then, A[j-1, ..., A.length] has its smallest element at A[j-1]

$$\begin{cases} A[j-1] \le A[j] : \text{from the inner loop.} \\ A[j] \le A[j+1] \\ A[j] \le A[A.length] \end{cases} \Big\} \text{ guaranteed before.}$$

* Termination: The inner loop terminates when j = i. Then the loop invariant for j = i says that the subarray A[i, ..., A.length] has its smallest element at A[i]. □

2. Loop invariant: "At the start of each iteration of the outer loop, A[1, ..., i] is already sorted in ascending order".

* Initialization: For i = 1. A[1] is sorted itself trivially.

* Maintenance: At the end of the inner loop for the i-th iteration of the outer loop, A[i+1, ..., A.length] has its smallest element at A[i+1], which is due to the loop invariant proved in 1. Then, A[1, ..., i+1] is sorted:

$$\underbrace{A[1] \le \cdots \le A[i]}_{\text{sorted before}} \le \underbrace{A[i+1]}_{\text{from the outer loop.}}$$

* Termination: The outer loop terminates when i = A.length. Then the loop invariant says that A[1, ..., A.length] is sorted in ascending order. □.

NO.

DATE

3. Bubble-Sort(A)

1.	for $i=1$ to A.length - 1 do		$\sim(n-1)$
2.	for $j=$ A.length down to $i+1$ do	$(n-1)+\cdots+1=\frac{n(n-1)}{2}=\sim(n^2)$	
3.	if $A[j] < A[j-1]$ then		$\sim(n^2)$
4.	exchange $A[j]$ with $A[j-1]$		$O(n^2)$

$\sim(n-1)+\ \sim(n^2)+\ \sim(n^2)+\ O(n^2)\ =\ \sim(n^2)$