Question one

Write the pseudocode for selection sort. What loop invariant does this algorithm maintain? Why does it need to run for only the first \( n-1 \) elements, rather than for all \( n \) elements? Give the best-case and worst-case running times of selection sort in theta notation.

Pseudocode

\[
\begin{align*}
\text{for firstUnsorted } &= 1 \text{ to } n-1 \\
\quad \text{lowestValue } &= -\infty \\
\quad \text{for } i &= \text{firstUnsorted to } n \\
\quad &\quad \text{if } A[i] < \text{lowestValue} \\
\quad &\quad \quad \text{lowestIndex } = i \\
\quad &\quad \quad \text{swap}(A[\text{firstUnsorted}-1],A[\text{lowestIndex}])
\end{align*}
\]

Loop invariant

At the start of each iteration of the outer for loop, the subarray \( A[1..\text{firstUnsorted}-1] \) consists of the first \( (\text{firstUnsorted}-1) \) lowest elements in the entire array.

Why only \( n-1 \)?

If the algorithm were to run for the last element, it would have only one choice for the element to swap, the last element. It would put this at the end of the sorted array. But it’s already there, so processing the final element never changes anything.
Best/worst case

\[ T(n) = \sum_{\text{firstUnsorted}=1}^{n-1} \left( c_1 + \sum_{i=\text{firstUnsorted}}^{n} c_2 + c_3 \right) \]

\[ = \sum_{\text{firstUnsorted}=1}^{n-1} (c_1 + c_2(n - \text{firstUnsorted} + 1) + c_3) \]

\[ = \sum_{\text{firstUnsorted}=1}^{n-1} (c_1 + c_2n - c_2 \cdot \text{firstUnsorted} + c_2 + c_3) \]

\[ = \sum_{\text{firstUnsorted}=1}^{n-1} (c_2n - c_2 \cdot \text{firstUnsorted} + (c_1 + c_2 + c_3)) \]

\[ = \sum_{\text{firstUnsorted}=1}^{n-1} (c_2n - c_2 \cdot \text{firstUnsorted} + c_4) \]

\[ = c_2n(n - 1) - c_2 \frac{(n - 1)(n)}{2} + c_4(n - 1) \]

\[ = c_2n^2 - c_2^2n - c_5n^2 - c_5n + c_4n - c_4 \]

\[ = (c_2 - c_5)n^2 - (c_2 - c_5 + c_4)n - c_4 \]

\[ = c_6n^2 - c_7n - c_4 \]

\( T(n) \) does not rely on the input, so the best and worst cases are the same. \( T(n) \in \Theta(n^2) \).
**Question two**

**Theorem**

If \( f(n) \in \Omega(g(n)) \) and \( g(n) \in \Omega(h(n)) \) then \( f(n) \in \Omega(h(n)) \)

**Proof**

By hypothesis:

\[
\exists c_1, n_1 > 0 \text{ such that } 0 \leq c_1 g(n) \leq f(n) \quad \forall \ n > n_1
\]

\[
\exists c_2, n_2 > 0 \text{ such that } 0 \leq c_2 h(n) \leq g(n) \quad \forall \ n > n_2
\]

Therefore

\[
\begin{align*}
  f(n) & \geq c_1 g(n) \geq 0 \quad \forall \ n > n_1 \\
 & \geq c_1 (c_2 h(n)) \geq 0 \quad \forall \ n > \max\{n_1, n_2\}
\end{align*}
\]

So for \( c_0 = c_1 c_2 \) and \( n_0 = \max\{n_1, n_2\} \), \( f(n) \in \Omega(h(n)) \).

**Theorem**

If \( f(n) \in \Theta(g(n)) \) and \( g(n) \in \Theta(h(n)) \) then \( f(n) \in \Theta(h(n)) \)

**Proof**

By hypothesis:

\[
\exists c_1, c_2, n_1 > 0 \text{ such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \quad \forall \ n > n_1
\]

\[
\exists c_3, c_4, n_2 > 0 \text{ such that } 0 \leq c_3 h(n) \leq g(n) \leq c_4 h(n) \quad \forall \ n > n_2
\]

Therefore

\[
\begin{align*}
  0 & \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \quad \forall \ n > n_1 \\
  0 & \leq c_1 (c_3 h(n)) \leq f(n) \leq c_2 (c_4 h(n)) \quad \forall \ n > \max(n_1, n_2)
\end{align*}
\]

Setting \( c_1 = c_1 c_3 \), \( c_2 = c_2 c_4 \), and \( n_0 = \max\{n_1, n_2\} \), \( f(n) \in \Theta(h(n)) \).
Given that $f(n) = 4^n$ and $g(n) = 2^n$:

a) Is $f(n) \in O(g(n))$?

$4^n = 2^{2n} = (2^n)(2^n) \neq c \cdot 2^n$

No. $4^n$ cannot be asymptotically upper bounded by $2^n$ times any constant, it needs to be multiplied by another exponential expression to equal $4^n$.

b) Is $\log f(n) \in O(\log g(n))$?

$\log 4^n = 2n$
$\log 2^n = n$
$2n \leq cn \forall n, c \geq 2$

Yes. $n_0 = 0$ and $c = 2$ would work.

c) Is $f(n) \in \Omega(g(n))$?

$4^n \geq c \cdot 2^n \geq 0 \forall n \geq 1$
$(2^n)(2^n) \geq c \cdot 2^n \geq 0 \forall n \geq 1$
$2^n \geq c \geq 0 \forall n \geq 1$

Yes. $n_0 = 1$ and $c = 1$ works.

d) Is $\log f(n) \in \Omega(\log g(n))$?

Yes. Since $\log f(n) = 2n$ and $\log g(n) = n$, the question is if $2n \in \Omega(n)$? For $c = 2$, the functions are identical and therefore $n_0 = 0$ works.
Question four

Prove that $\log(n!) \in O(n \log n)$.

By hypothesis:

\[
\begin{align*}
\log(n!) & \leq c \cdot n \cdot \log(n) & \forall n > n_0 \\
\log(n!) & \leq \log(n^n) & \forall n > n_0 \\
n! & \leq n^n & \forall n > n_0, c = 1 \\
1 \cdot 2 \cdot \ldots \cdot n & \leq n \cdot n \cdot \ldots \cdot n & \forall n > n_0, c = 1
\end{align*}
\]

Each element in $n!$ is less than or equal to the corresponding element in $n^n$, so the inequality holds for $c = 1$ and all positive $n$. 
1) Let $f(n) = 1 + 2 + 3 + \ldots + n$ and $g(n) = n^3$. Is $f(n) \in O(g(n))$?

$$f(n) = \frac{n(n+1)}{2} = \frac{1}{2}(n^2 + n)$$

For $n > n_0$, by hypothesis and the line above, $\frac{1}{2}(n^2 + n) \leq c_1 n^3 \forall n > n_0$.

Let $c_2 = 2c_1$, so $n^2 + n \leq c_2 n^3 \forall n > n_0$.

Divide by $n^2$ to get $1 + \frac{1}{n} < c_2 n \forall n > n_0$.

As $n$ increases, the left decreases and the right increases, so for high $n$ the inequality holds.

2) Let $f(n) = 2 \cdot 2^{\left(\frac{1}{n}\right)}$ and $g(n) = 2^{\left(\frac{1}{n}\right)}$. Is $\log f(n) \in O(\log(g(n)))$?

By hypothesis:

$$\log\left(2 \cdot 2^{\left(\frac{1}{n}\right)}\right) \leq c \cdot \log\left(2^{\left(\frac{1}{n}\right)}\right) \forall n > n_0$$

$$\log\left(2 \cdot 2^{\left(\frac{1}{n}\right)}\right) \leq \log\left(2^{\left(\frac{1}{n}\right)}\right) \forall n > n_0$$

$$2 \cdot 2^{\left(\frac{1}{n}\right)} \leq 2^{\left(\frac{1}{n}\right)} \forall n > n_0$$

No. Whatever $c$ is, the $2^{\left(\frac{1-c}{n}\right)}$ term will go to 1 as $n$ goes to infinity, so the left side will approach 2.

3) Let $f(n) = n^2$. Is $f(n) \in \Theta(f(2n))$?

By hypothesis:

$$0 \leq c_1 4n^2 \leq n^2 \leq c_2 4n^2 \forall n > n_0$$

Yes. Let $c_1 = c_2 = \frac{1}{4}$, and all expressions are equal for all $n$, so $n_0 = 0$ works.