Problem 1

(a) By applying the limit test:

$$\lim_{n\to\infty} \frac{3^{n+1}}{3^n} = 3 \to 3^{n+1} = \Theta(3^n)$$
$$\lim_{n\to\infty} \frac{3^{3n}}{3^n} = 3^{2n} = \infty \to 3^{3n} = \Omega(3^n) \text{ and } 3^{3n} \neq O(3^n)$$

(b) Notice that $1 \leq 2 + \cos n \leq 3$ for all n. Hence, for all $n \geq 1$, we have $n^3 \geq n^{2+\cos n}$, which implies that $n^3 = \Omega(n^{2+\cos n})$ by definition.

To show that $n^3 \neq O(n^{2+\cos n})$, we will proceed by contradiction. Assume, for the sake of contradiction, that there exist constants $C > 0$ and $n_0 \geq 0$ such that $n^3 \leq Cn^{2+\cos n}$ for all $n \geq n_0$. This inequality simplifies to:

$$n^{3-(2+\cos n)} = n^{1-\cos n} \leq C \quad \text{for all } n \geq n_0.$$

However, this leads to a contradiction. The left-hand side, $n^{1-\cos n}$, can be made arbitrarily large for large values of n where $\cos n < 0$. Specifically, since $\cos n$ oscillates between -1 and 1, there are infinitely many values of n where $\cos n < 0$, and thus $1 - \cos n$ becomes larger than 1. For such n, the expression $n^{1-\cos n}$ grows without bound as n increases, which contradicts the assumption that it is bounded by a constant C.

Therefore, the assumption that $n^3 = O(n^{2+\cos n})$ must be false.

(c) Notice that

$$g(n) = \frac{n^5 + n^{3+\sin n} + 1}{n^2 + n + 1} = n^3 \times \frac{1 + n^{\sin n - 2} + n^{-5}}{1 + n^{-1} + n^{-2}} \to \lim_{n\to\infty} \frac{g(n)}{n^3} = 1.$$

Thus, $g(n) = \Theta(n^3)$, meaning that $g(n) = O(n^3)$ and $g(n) = \Omega(n^3)$.

Since $n^3 = \Omega(n^2)$ and $n^3 = O(n^4)$, by transitivity, we also have $g(n) = \Omega(n^2)$ and $g(n) = O(n^4)$.

Thus, the correct choice is D. The same conclusion can be reached using the limit property.

Problem 2

(a)

Data: Array *arr* of length n
Result: Scalar *val*
$val \leftarrow 0$;
for $i \leftarrow 1$ **to** n^2 **do**
  $j \leftarrow 1$;
  while $j \leq i^2$ **do**
    $val \leftarrow val + arr[i] \times j^2$;
    $j \leftarrow j + 1$;
  end
end

Assignment to variables in this algorithm is basic, and operations are minimal, with $\Theta(1)$ complexity.

Consider the outer loop of the algorithm running from 1 to n^2. This will be the basis of our summation, as everything inside the outer loop repeats n^2 times.

Now consider the inner loop, which runs from 1 to i^2. This will be the content of the summation, as i^2 iterations of the inner loop are repeated for every iteration of the outer loop.

This gives us the summation:

$$\sum_{i=1}^{n^2} \theta(i^2)$$

Written out, the summation is as follows:

$$1^2 + 2^2 + \ldots + {n^2}^2 = \frac{n^2(n^2+1)(2n^2+1)}{6}$$

This is found by using the Summation of Squares Formula.

From here, combining the terms in the numerator gives the largest degree of n^6, meaning that the runtime complexity of the algorithm is $\Theta(n^6)$.

(b)

Data: Array arr of length n
Result: Scalar val

```
val ← 0;
for i ← 1 to n do
    j ← 1;
    for j ← 1 to ln(i) do
        val ← val + j × (arr[i])^n;
    end
end
```

For this algorithm, it is important to notice that the assignment includes exponentiation by n, as shown below:

$$(arr[i])^n$$

Since exponentiation by n inherently contains n multiplication operations of $\Theta(1)$, the total time complexity for the variable assignment is $\Theta(n)$, as the algorithm must spend time calculating the value.

Now consider the outer loop, which goes from 1 to n. This will be the basis of the summation.

The inner loop iterates from 1 to $\ln(i)$, which is then repeated per iteration of the outer loop.

Therefore, the summation representing this algorithm will be:

$$\sum_{i=1}^{n} \theta(n \ln(i))$$

Written out, the summation is as follows:

$$n\ln(1) + n\ln(2) + \ldots + n\ln(n) = n\ln(n!)$$

Note that n is a constant in this case, and doesn't change per iteration.

Also remember addition of logarithms, where $\log(x) + \log(y) = \log(xy)$, which is why $\ln(1) + \ldots + \ln(n)$ is equal to $n!$.

Using the hint as substitution for $\ln(n!)$ inside of $n\ln(n!)$:

$$\ln(n!) = n\ln(n) - n + O(\ln(n))$$

Since the largest term from the hint is $n\ln(n)$, runtime complexity is found by multiplying $n\ln(n)$ by the existing n, meaning the runtime complexity of the algorithm is $\Theta(n^2\ln(n))$.

(c)

Data: None
Result: Scalar *val*
val $\leftarrow 0$;
for $i \leftarrow 1$ **to** n **do**
  $j \leftarrow 1$;
  while $j \leq i$ **do**
    $val \leftarrow val + j \times i$;
    $j \leftarrow 3 \times j$
  end
end

For this algorithm, all assignments are basic and mathematical operations are simple, leading to a constant runtime of $\theta(1)$.

The outer loop of this algorithm iterates from 1 to n, so this will be the basis of the summation.

The inner loop iterates from 1 to i, but it is indexed by multiplying the previous index by 3 each time, as seen below:

$$j \leftarrow 3 \times j$$

Therefore, the complexity of the inner loop is $\theta(\log_3(i))$, as it progresses arithmetically, rather than linearly (like if it was indexed by a constant +3 instead).

It is important to note that in runtime notation, the base of the logarithm does not matter, so $\theta(\log_3(n))$ is equivalent to $\theta(\log(n))$.

The resulting summation is as follows:

$$\sum_{i=1}^{n} \theta(\log(i))$$

Written out, the summation will be:

$$\log(1) + \log(2) + \ldots + \log(n) = \log(n!)$$

Also remember addition of logarithms, where $\log(x)+\log(y) = \log(xy)$, which is why $\log(1)+\ldots+\log(n)$ is equal to $\log(n!)$.

Substituting for $\log(n!)$:

$$\log(n!) = n\log(n) - n + O(\log(n))$$

The largest term is $n\log(n)$, meaning the runtime complexity of the algorithm is $\Theta(n\log(n))$.

Problem 3

(a) $\lim_{n\to\infty} \frac{\frac{n!}{k!\times(n-k)!}}{n^k} = \frac{\frac{n\times(n-1)\times\ldots\times(n-k+1)}{k!}}{n^k} = \frac{1}{k!} \to \binom{n}{k} = \Theta(n^k)$

(b) $2 + \sin n \in [1, 3]$. Following a similar argument as in Problem 1(b), $n^{2+\sin n} = \text{None}(n^2)$.

Problem 3 continued on next page. . .

(c) Notice that $5 + 4\sin n \in [1, 9]$, hence for all $n \geq 1$, we have $\log n \leq \log(n^{5+4\sin n}) \leq 9 \log n$. Hence, $\log(n^{5+4\sin n}) = \Theta(\log n)$.

(d) $\lim_{n\to\infty} \frac{n^3+2n^2+5}{n^3} = 1 \to n^3 + 2n^2 + 5 = \Theta(n^3)$

(e) $\lim_{n\to\infty} \frac{2^n}{n^3} = \infty \to 2^n = \Omega(n^3)$ and $2^n \neq O(n^3)$

(f) $\lim_{n\to\infty} \frac{n\log n+n}{n\log n} = \lim_{n\to\infty} \frac{\log n+1}{\log n} = 1 \to n\log n + n = \Theta(n\log n)$

(g) $\lim_{n\to\infty} \frac{\log n}{n^2} = 0 \to \frac{\log n}{n^2} = O(1)$ and $\frac{\log n}{n^2} \neq \Omega(1)$

(h) $\lim_{n\to\infty} \frac{k^n}{2^n} = \lim_{n\to\infty} \left(\frac{k}{2}\right)^n = \infty \to k^n = \Omega(2^n)$ and $k^n = O(2^n)$

(i) $\lim_{n\to\infty} \frac{\sqrt{n}}{\log n} = \infty \to \sqrt{n} = \Omega(\log n)$ and $\sqrt{n} \neq O(\log n)$

(j) $\lim_{n\to\infty} \frac{\log^2 n}{n} = 0 \to \log^2 n = O(n)$ and $\log^2 n \neq \Omega(n)$