Question 1

ⓐ $f(n) = (n^2 - 3n)^2$, $g(n) = 5n^3 + n$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = \lim_{n \to \infty} \frac{n^4 - 6n^3 + 9n^2}{5n^3 + n} = \frac{\infty}{\infty}$$

i) L'Hospital: $\lim_{n \to \infty} \frac{4n^3 - 18n^2 + 18n}{15n^2 + 1} = \frac{\infty}{\infty}$

ii) L'Hospital: $\lim_{n \to \infty} \frac{12n^2 - 36n + 18}{30n} = \frac{\infty}{\infty}$

iii) L'Hospital: $\lim_{n \to \infty} \frac{24n - 36}{30} = \infty$

$\boxed{\text{Therefore } f(n) \in \Omega(g(n))}$

ⓑ

$f(n) = n^3$, $g(n) = \log_2 n^4$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = \lim_{n \to \infty} \frac{n^3}{\log_2 n^4} = \frac{\infty}{\infty}$$

i) L'Hospital: $\lim_{n \to \infty} \frac{3n^2}{\frac{4}{n.\ln 2}} = \lim_{n \to \infty} \frac{3n^2 . n . \ln 2}{4} = \infty$

$\boxed{\text{Therefore } f(n) \in \Omega(g(n))}$

ⓒ $f(n) = 5n . \log_2(4n)$, $g(n) = n . \log_2(5^n)$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = \lim_{n \to \infty} \frac{5n . \log_2(4n)}{n . \log_2(5^n)} = \frac{\infty}{\infty}$$

i) L'Hospital: $\lim_{n \to \infty} \frac{\frac{5}{\ln 2 . n}}{\log_2 5} = \frac{5}{\log_2 5 . \ln 2 . n} = \frac{5}{\infty} = 0$

$\boxed{\text{Therefore } f(n) \in O(g(n))}$

P (d) $f(n) = n^n \quad , \quad g(n) = 10^n$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{n^n}{10^n} = \lim_{n\to\infty} \left(\frac{n}{10}\right)^n = \frac{\infty}{10} = \infty$$

$\boxed{\text{Therefore } f(n) \in \Omega(g(n))}$

(e) $f(n) = 8n \cdot \sqrt[5]{2n} \quad , \quad g(n) = n \cdot \sqrt[3]{n}$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{8\cancel{n} \cdot \sqrt[5]{2n}}{\cancel{n} \cdot \sqrt[3]{n}} = \lim_{n\to\infty} \frac{8 \cdot 2^{1/5} \cdot n^{1/5}}{n^{1/3}} = \frac{8 \cdot 2^{\frac{1}{5}}}{n^{2/15}} \rightarrow 2^{\frac{3}{5}}$$

$$\lim_{n\to\infty} \left(\frac{2^{3/5}}{n^{2/15}}\right) = \frac{2^{3/5}}{\infty} = 0$$

$\boxed{\text{Therefore } f(n) \in \Omega(g(n))}$

Question 2

a) The complexity of this method is $O(n)$. This is because the loop is executed as many times as the size of the array (n times), and one operation is performed at each step.

b) The complexity of this method is $O(n2)$. There are two nested loops. The first loop executes n times and the second loop exeutes n times. Since methodA is called at each inner loop step, the complexity of this method is $O(n)$. Therefore, the total comlexity is $O(n^2)$

c) MethodC contains two nested loops. The first loop runs n times and the second loop runs n times. However, methodB is called at each inner loop step, and since the comlexity of methodB is $O(n^2)$, the total comlexity must be $O(n^4)$.

d) The loop inside methodD proceeds by changing the value of i between 0 and $n-1$ to process each element of the array. However, since $(i--)$ is used in the body of the loop, i is decremented by 1 at each step. Therefore, the value of i will never increase and the loop will continue forever. In this case, since there is an infinite loop in methodD, the complexity of this method is undefined and cannot be analysed.

Ⓒ The complexity of this method is O(n). A single loop is executed n times and each step is a fixed-time operation.

Question 3

A) pseudo-code

```
function maxDifferenceSorted (array A):
        n = length of A
        if n < 2 : return -1
return A[n-1] - A[0]
```

In the case of an ordered array, the first element will be the smallest value. Therefore, in an ordered array, we can assume that the smallest element is the first element of the array. Since the first element is the smallest value, we can get the largest difference by simply taking the last element and substracting it from the first element. In this case, the complexity of the algorithm would be O(1).

B) pseudo-code

```
function maxDiffereneUnsorted (array A)
        n = length of A
        if n < 2 : return -1
        min_val = A[0]
        max_diff = A[1] - A[0]
        for i from 1 to n-1:
            if A[i] - min_val > max_diff:
                    max_diff = A[i] - min_val
            if A[i] < min_val
                    min_val = A[i]
return max-diff
```

To provide this functionality we can keep track of the maximum and minimum values using a loop. We can then calculate the maximum difference as the difference between these two values. This algorithm has O(n) complexity for a non-ordered array because it visits all elements in a single loop and performs a fixed number of operations for each element. In this case, at most n operations are performed during the cycle. Therefore, the complexity of the algorithm is the O(n).