Question 4:

(a) $2x^4$ is $O(x^3+3x+2)$

For the above statement to hold, the following must hold for constant witnesses C and k.

$$2x^4 \le C(x^3+3x+2), x > k$$

Given that $1 \le x \le x^2 \le x^3$ for $x > 1$,

$$2x^4 \le C(x^3+3x+2) \le C(x^3+3x^3+2x^3)$$

$$2x^4 \le C(6x^3)$$

$$\frac{1}{3}x < C$$

As x increases, so does $\frac{1}{3}x$, this means that for any pair of witnesses C and k, there exists a value for x which will make $2x^4$ greater than $C(x^3+3x+2)$, meaning the statement $2x^4$ is $O(x^3+3x+2)$ is not true.

(b) $4x^3 + x^2 \cdot \log x + 1$ is $O(x^3)$

$$4x^3 + x^2 \log x + 1 \le Cx^3$$

$$4 + \frac{\log x}{x} + \frac{1}{x^3} \le C$$

$\frac{1}{x^3}$ decreases as x increases for $x > 0$, as $x < \log x$ for $x \ge 1$ ($\log 0$ is undefined). For the witnesses this means that taking a value of $k = 1$ yields $4 + \frac{\log 1}{1} + \frac{1}{1} \le C$, or $5 \le C$, thus $C = 5$ can be used. The witnesses $k = 1, C = 5$ show that $4x^3 + x^2 \cdot \log x + 1$ is $O(x^3)$.

(c) $3x^2 + 7x + 1$ is $\omega(x \cdot \log x)$

$$3x^2 + 7x + 1 = \omega(x \cdot \log x) \Leftrightarrow x \cdot \log x = o(3x^2 + 7x + 1)$$

For this to be true,

$$\lim_{x\to\infty}\left(\frac{3x^2+7x+1}{x \cdot \log x}\right) = 0$$

$$= \lim_{x\to\infty}\left(\frac{3x+7+\frac{1}{x}}{\log x}\right)$$

$$= 3\lim_{x\to\infty}\left(\frac{x}{\log x}\right) + \lim_{x\to\infty}\left(\frac{7}{\log x}\right) + \lim_{x\to\infty}\left(\frac{1}{\mathrm{x}\log x}\right)$$

Using L'Hôpital's rule:

$$= 3\lim_{x\to\infty}\left(\frac{1}{\frac{1}{x\ln 2}}\right) + 0 + 0 = 3\lim_{x\to\infty}(x \ln 2) = \infty \ne 0$$

As the limit does not equal zero, $3x^2 + 7x + 1 \ne \omega(x \cdot \log x)$.

(d) $x^2 + 4x$ is $\Omega(x \cdot \log x)$

$$x^2 + 4x \geq C \cdot x \cdot \log x$$

$$x + 4 \geq C \cdot \log x$$

$$x \geq \log x^C - \log 16$$

$$x \geq \log \frac{x^c}{16}$$

This means that for $k = 1$, $1 > \log\frac{1^C}{16}$ for all $C > 1$, hence the witnesses $k = 1, C = 1$ work to show that $x^2 + 4x$ is $\Omega(x \cdot \log x)$.

(e) $f(x) + g(x)$ is $\Theta(f(x) \cdot g(x))$

This is not necessarily true. For example take $f(x) = x^2$ and $g(x) = x^3$, this would mean that to be true, $C_0 \cdot f(x) \cdot g(x) \leq f(x) + g(x) \leq C_1 \cdot f(x) \cdot g(x)$ for some k where $x \geq k > 0$. Substituting $f(x)$ and $g(x)$ in this produces $C_0 \cdot x^5 \leq x^2 + x^3 \leq C_1 \cdot x^5, x \geq k$. For the left hand inequality this reduces to $C_0 \leq x^{-3} + x^{-2}$, choosing a value for $C_0 > 0$ here is not possible as the right hand side of the inequality will always get arbitrarily closer to zero.

<u>Question 5</u>:

(a) $T(n) = 9T\left(\frac{n}{3}\right) + n^2$

This is of the form $T(n) = aT\left(\frac{n}{b}\right) + f(n)$, where $a = 9$, $b = 3$ and $f(n) = n^2$. This is the second of the three cases in master theorem as $f(n) = \Theta\left(n^{\log_b a}\right) = \Theta\left(n^{\log_3 9}\right) = \Theta(n^2) = \Theta(f(n))$ (this clearly demonstrates that $f(n)$ is $\Theta(n^2)$ with witnesses $k = 1$, $C_0 = 1$, $C_1 = 1$), meaning the runtime classification in this case is $\Theta(\mathrm{n}^2 \log n)$.

(b) $T(n) = 4T\left(\frac{n}{2}\right) + 100n$

For this runtime recurrence, $a = 4$, $b = 2$ and $f(n) = 100n$. Here the first case of Master Theorem can be applied as follows.

$$n^{\log_b a - \epsilon} = n^{2-\epsilon}$$

For $\epsilon = 1$,

$$f(n) = 100n = O(n^{2-1}) = O(n)$$

In this case $100n$ is $O(n)$ with witnesses $k = 1$ and $C = 100$, hence using the first case the runtime classification is $\Theta\left(n^{\log_b a}\right) = \Theta(n^2)$.

(c) $T(n) = 2^n T\left(\frac{n}{2}\right) + n^3$

Here $a = 2^n$, as this is not constant the runtime complexity cannot be resolved using the Master Theorem.

(d) $T(n) = 3T\left(\frac{n}{3}\right) + c \cdot n$

Here $a = 3$, $b = 3$, $f(n) = cn$, here the second case of Master Theorem can be used as follows.

$$n^{\log_b a} = n^1 = n$$

Using witnesses $k = 1$, $C_0 = c$, $C_1 = c$ (assuming $c > 0$),

$$f(n) = cn = \Theta(n) = \Theta(\mathrm{n}^{\log_b a})$$

Hence, the runtime classification is $\Theta(\mathrm{n} \log n)$.

(e) $T(n) = 0.99T\left(\frac{n}{7}\right) + \frac{1}{n^2}$

In this case, $a = 0.99$ (< 1). The use of the Master Theorem requires $a \geq 1$ thus here it cannot be used.

Question 6: (b)

For an array length of four of less the worst-case input is the same as that of the insertion sort – a reverse sorted list (in this case from low to high), this will cause the insertion sort part of the algorithm to compare every item. When the merge sort is added in some worst-case examples are:

Number of Items	Number of Comparisons	Worst Case Example
0	0	[]
1	0	[1]
2	1	[1, 2]
3	3	[1, 2, 3]
4	6	[1, 2, 3, 4]
5	8	[1, 3, 2, 4, 5]
6	11	[1, 3, 4, 2, 5, 6]
7	15	[1, 3, 4, 2, 5, 6, 7]
8	19	[1, 3, 4, 5, 2, 6, 7, 8]
9	22	[1, 3, 4, 5, 2, 7, 6, 8, 9]
10	25	[1, 4, 3, 5, 6, 2, 8, 7, 9, 10]

This was done using the following code by calculating every permutation of the list [1, …, n] and counting the number of comparisons. The variable c is global and is incremented on each comparison. The worse cases shown occur when merge sort has to compare every value in the merge function.

```python
c = 0
if __name__ == '__main__':
    from itertools import permutations
    max_c = 0
    permutation_ = []
    t = 0
    l = []
    for i in range(10):
        for permutation in permutations(l):
            c = 0
            hybrid_sort(list(permutation))
            if c > max_c:
                max_c = c
                permutation_ = permutation
            t += 1
        l.append(i + 1)

        print(i, max_c, list(permutation_))
```