Q4.

$$T(n) = 1, \quad if \quad n = 1$$
$$= 4T(\frac{n}{2}) + n\log n, \quad if \quad n \geq 2$$

Let $n = 2^b$:

$$\begin{aligned}
T(2^b) &= 1, \quad if \quad n = 1 \\
&= 4T(2^{b-1}) + 2^b b, \quad if \quad n \geq 2 \\
&= 4\{4T(2^{b-2}) + 2^{b-1}(b-1)\} + 2^b b \\
&= 4^2 T(2^{b-2}) + 2^{b+1}(b-1) + 2^b b \\
&= 4^3 T(2^{b-3}) + 2^{b+2}(b-2) + 2^{b+1}(b-1) + 2^b b \\
&. \\
&. \\
&. \\
&= 4^k T(2^{b-k}) + \sum_{i=0}^{k-1} 2^{b-i}(b-i)
\end{aligned}$$

T(1)=1 thus $2^{b-k} = 1 \quad when\ k = b$. Therefore:

$$= 4^b T(1) + \sum_{i=0}^{b-1} 2^{b-i}(b-i)$$
$$= (2^b)^2 + \sum_{i=0}^{b-1} 2^{b-i}(b-i)$$

It can be shown that $\sum_{i=0}^{b-1} 2^{b-i}(b-i) = 2(2^b \bullet b - 2^b + 1)$. Therefore putting this into the earlier equation and substituting n back in, we get:

$$T(n) = n^2 + 2n\log n - 2n + 1$$

Q5.

As the question requires us to sort items in the array in O(n) and there is a restriction on the input values, the answer is likely to be using radix sort. However, we need to know how many digits the largest number will be:

Let d equal the number of digits in the largest possible number, which in this case is $n^2 - 1$. The number of digits of the number will be dependent on the base we use. Thus:

$d = O(\log_b n)$ where b is our choice of base.

Therefore, the running time for this radix sort will be $O(d(n+b)) = O(\log_b n(n+b))$. If we make the b=n, then this will be equal to O(n). Therefore, to sort this sequence in O(n) time, we need implement Radix with the values written in base of n.

The method:

Convert all the numbers in the sequence into base n. Start by separating all the numbers by their LSB, and maintaining their order, continue separating them till their MSB. In the end, you will have a sorted list of the sequence in base n. Convert back to base 10 and you will have a sorted sequence.