1. Yes, this algorithm is correct. Since a main job of sorting algorithms is to swap elements in the sequence to make it in a sorted order.

2. We would expect the algorithm to take $\Omega(1)$ time in the best case. In this case, the sequence only have two out-of-place elements and the algorithm happened to choose the two elements in the first iteration of the loop; we only iterated the loop once and leave, since the sequence is now sorted.

3. In the worst case, this algorithm will run forever. In this case, the algorithm happened to always choose the wrong elements to swap, and we never arrived at the state that the sequence is sorted, the loop is an infinite loop.

4. On average, we will still expect the time required to sort the sequence to be infinite.
If we assume that, after each iteration of the loop, we have an equal probability of the sequence being sorted. That is, P(sorted after X iterations) = c; for some constant c.
Since we have to at least iterate the loop once and at most iterate forever, we can say the possible value for X is any integer in the range of [1, positive infinity).
Therefore, we can say that X follows a discrete uniform distribution with parameters {1, positive infinity}.
Then, the average value of X can be calculated by finding the mean.
That is, E(X) = (1 + positive infinity) / 2 = positive infinity.