Q1-What is the difference between in-place and out-place sorting algorithms?

A1 - Usually, an algorithm is categorized as an in-place or an out-of-place algorithm based on the explicit storage allocated by the algorithm.

#In place Sorting Algorithm-

1. An in-place algorithm transforms the input without using any extra memory. As the algorithm executes, the input is usually overwritten by the output, and no additional space is needed for this operation.
2. An in-place algorithm may require a small amount of extra memory for its operation. However, the amount of memory required must not be dependent on the input size and should be constant.
3. An in-place sorting algorithm sorts the elements in place: that is, it needs only O(1) extra space.

Examples of in-place algorithms - insertion sort, selection sort, quick sort, bubble sort, heap sort, etc.

#Out place Sorting Algorithm-

1. An algorithm that is not in-place is an out-of-place algorithm.
2. Unlike an in-place algorithm, the extra space used by an out-of-place algorithm depends on the input size.
3. Usually this means O(n) extra space.

Examples of out-place algorithms - The standard merge sort algorithm is an example of out-of-place algorithm as it requires O(n) extra space for merging.

Q2- Implement Insertion sort in both (in-place and out-place) manner.
A2-Done in vs code

Q3- Suggest some practical examples of using in-place and out-place techniques
A3-In place practical examples: Heap Sorting is used in sim card store where there are many customers in line. The customers who have to pay bills can be dealt with first because their work will take less time. This method will save time for customers .
One more real-world example of insertion sort is how tailors arrange shirts in a cupboard, they always keep them in sorted order of size and

thus insert new shirts at the right position very quickly by moving other shirts forward to keep the right place for a new shirt.
For bubble sort , for eg,there are five cars travelling on a road and each of the cars' speeds have been set to slightly different values.

When a car is travelling faster than the car in front, it will overtake it, and occupy the slower car's position in the traffic. This will keep happening, with each car switching positions with any slower car in front of it. Eventually the cars will sort themselves according to their speeds, with the fastest car being at the front of the queue of traffic.

Out place practical examples:
Merge Sort is useful for sorting linked lists in O(n Log n) time and this sort can be implemented without extra space for linked lists .It is used for counting inversions in a list, is used in external sorting.