Question 1

Question 2

(a)

If sorting the array described in the question, I would recommend using Merge Sort. Although both sorting algorithms would be suitable, Merge sort provides a better time & space complexity for the given array.

Looking at the time complexity, Mergesort time complexity is guaranteed O(n log(n)), however using Distribution sort would bring the worst case time complexity up to O(n + n2) = O(n2). This is because the range of values in the array is (n2) is greater than the number of elements in the array (n).

In terms of space complexity, Merge sort provides a space complexity of O(n), which is better than Distribution Sort, which would again be O(n2), the range of the values in the array.

Merge Sort Complexities for Q2a

Time: O(n log (n))

Space: O(n)

Distribution Sort Complexities for Q2a

Time: O(n2)

Space: O(n2)

(b)

If the array only contained multiples of n, I would recommend using Distribution sort, as this additional information significantly improves the time and space complexities for Distribution sort, giving better complexities than Mergesort. The time complexity for Distribution sort would become O(n + (n+1)) = O(n). The range of values in the array shifts from (n2) to (n + 1). The space complexity would also become O(n), the same as Mergesort.

Merge Sort Complexities for Q2b

Time: O(n log (n))

Space: O(n)

Distribution Sort Complexities for Q2b

Time: O(n)

Space: O(n)