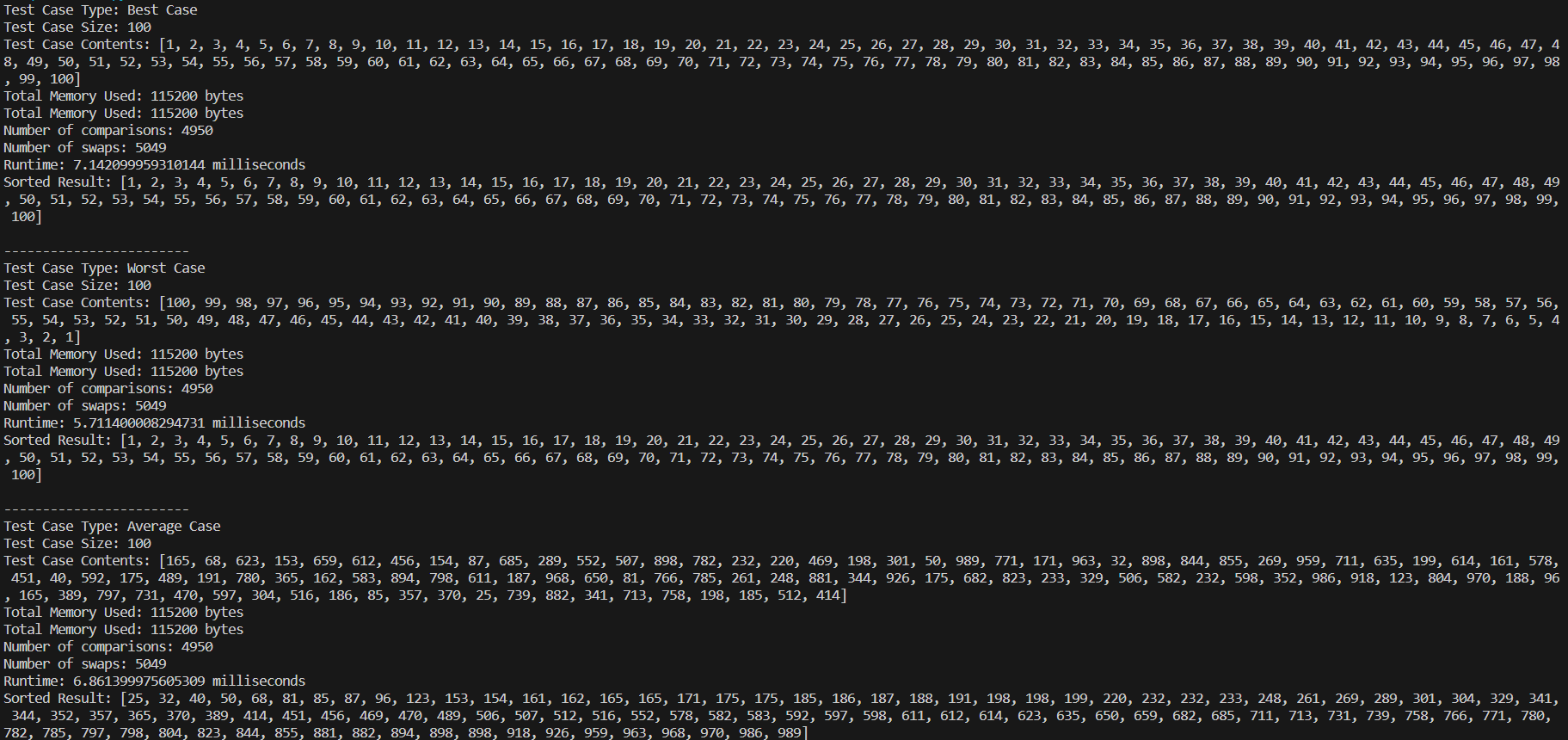
DAA – Assignment.

1.

i) In-place quick sort algorithm



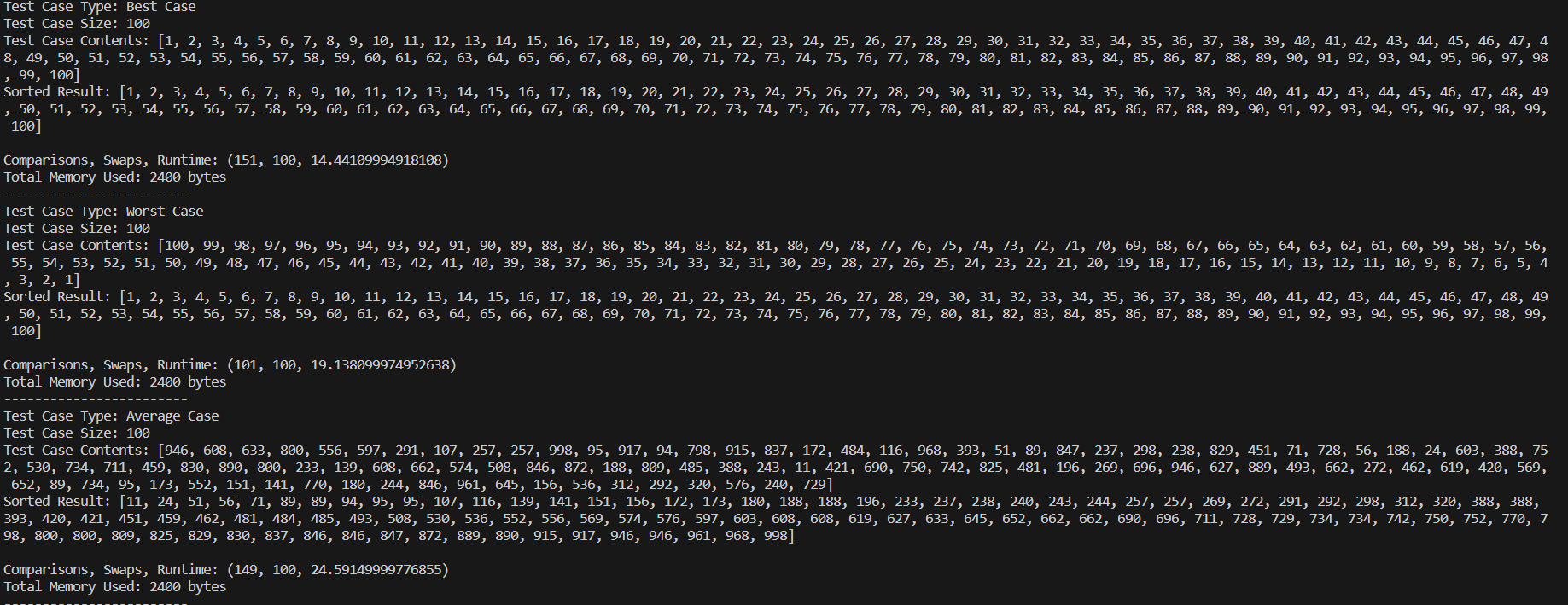
The "maximum recursion depth exceeded" error occurs because the depth of recursion in Python is limited. To correct this error, you can increase the recursion limit using sys.setrecursionlimit() or refactor the code to use an iterative approach, which is often preferred for large input sizes to avoid stack overflow errors.

It makes approximately O(n\*log(n)) comparisons on average.

In the worst case (when the pivot choice is poor), it can make O(n^2) comparisons.

The average number of swaps is relatively low, and in most practical cases, it's very efficient. However, the choice of pivot strategy is crucial for its performance.

ii) Merge sort



The time complexity is O(N \* log2N).

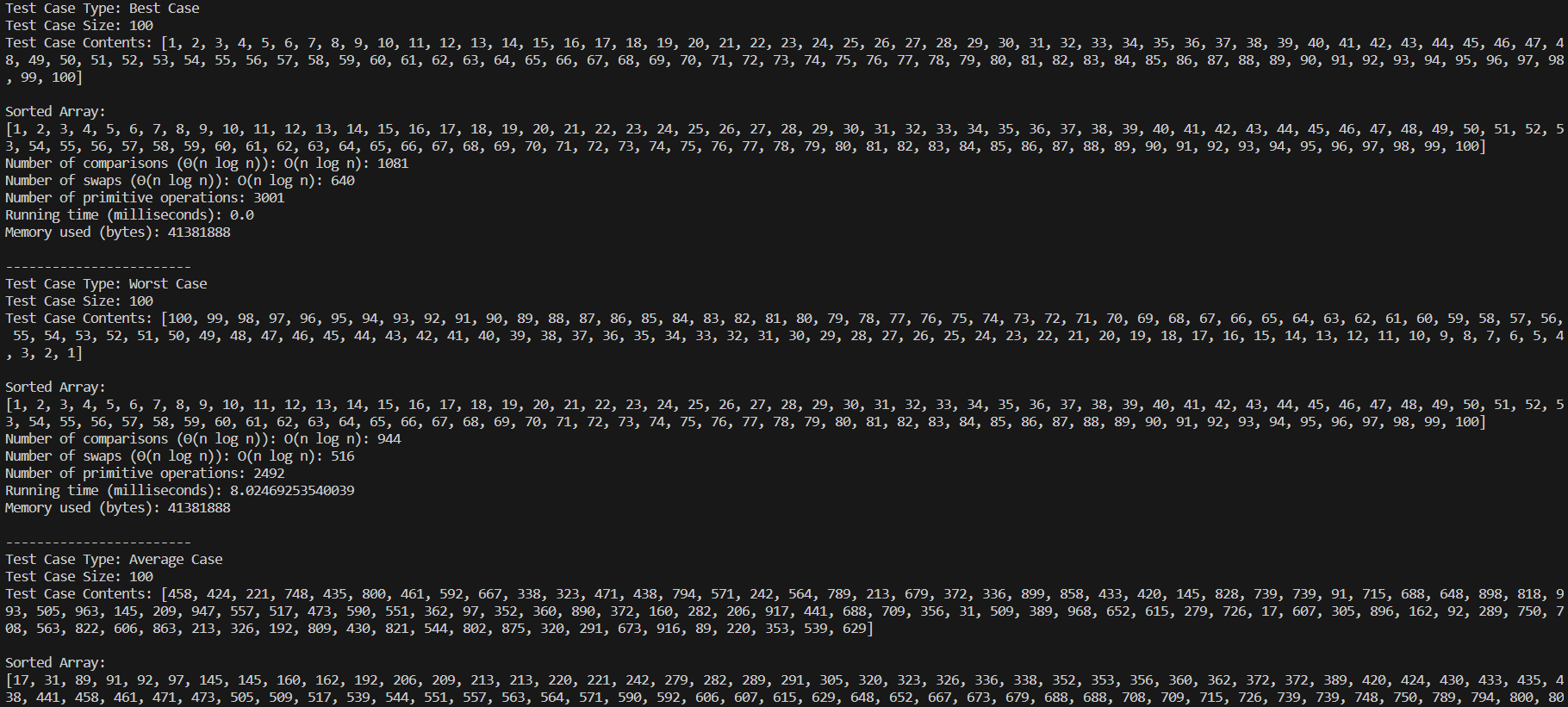
So, in the best case, the worst case and the average case the time complexity is the same.

In case of merge sort, we need to build a temporary array to merge the sorted arrays. So the auxiliary space requirement is O(N).

Time Complexity: O(N log(N)), Merge Sort is a recursive algorithm and time complexity can be expressed as following recurrence relation.

T(n) = 2T(n/2) + θ(n)

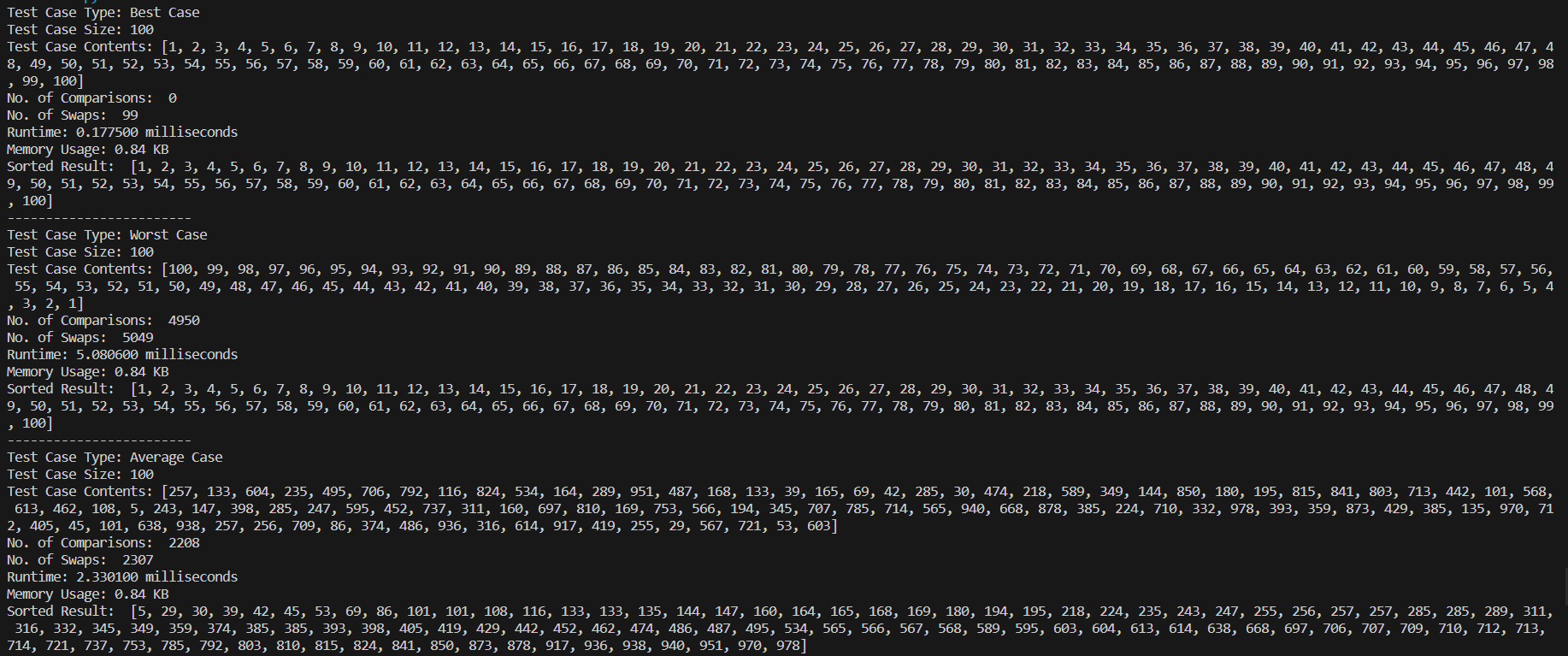
iii) In-place Heap Sort



Implies that only additional O(1) extra data structure items are required to solve the problem.

In the worst-case scenario, the time complexity of in-place Heap Sort is O(n log n), which makes it efficient for large datasets.

iv) Insertion Sort



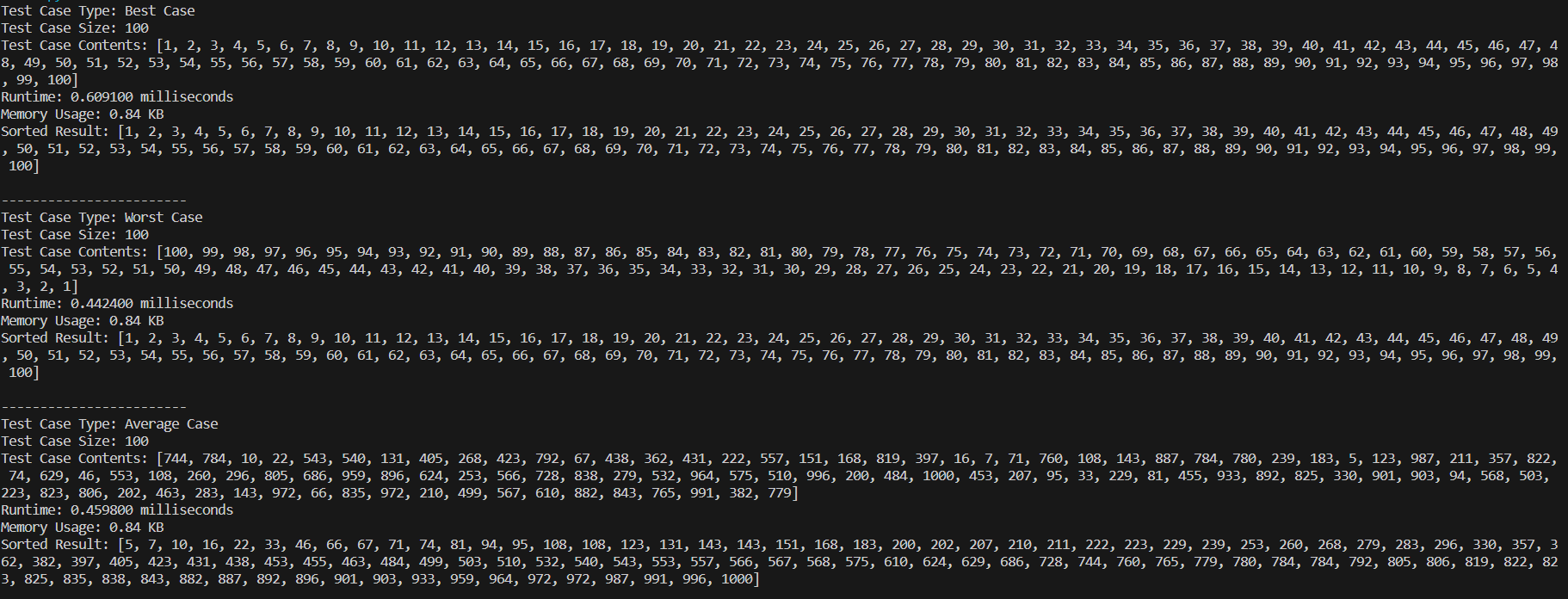
Despite having a worst-case time complexity of O(n^2), Insertion Sort is often faster than more complex algorithms like Quick Sort and Merge Sort for small input sizes or when the input data is already partially sorted.

Best-case time complexity: O(n)

Average-case time complexity: O(n^2)

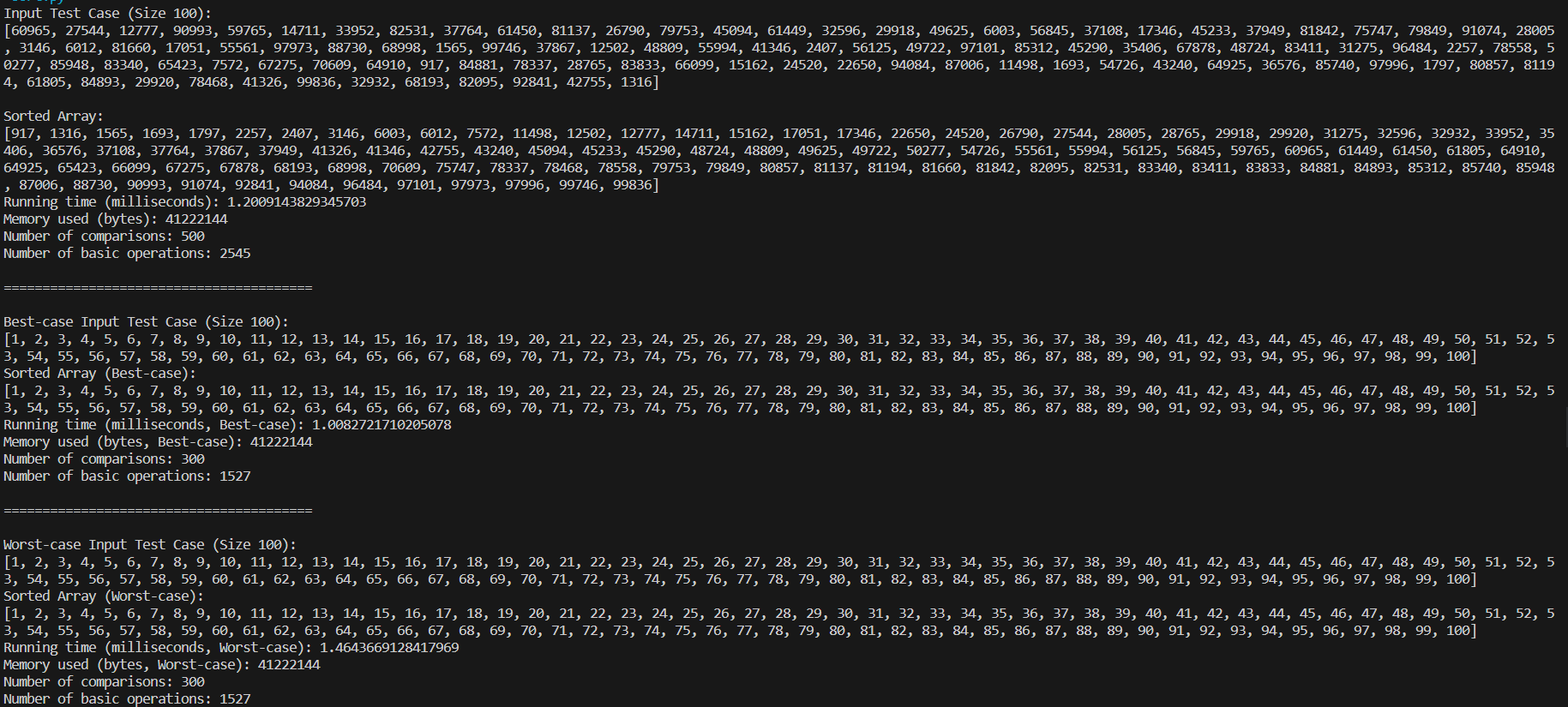
Worst-case time complexity: O(n^2)

v) Bucket Sort



Bucket Sort is used to sort an array where elements are uniformly distributed, or where the elements of the array range between 0 and 1. Bucket sort can exhibit the best case time complexity of O(n+k), where n is the number of buckets and k is the bucket size.

vi) Radix sort



Radix Sort is a stable sorting algorithm with a general time complexity of O(k · (b + n)), where k is the maximum length of the elements to sort ("key length"), and b is the base. If the maximum length of the elements to sort is known, and the basis is fixed, then the time complexity is O(n).

Radix Sort is an in-place sorting algorithm, which means it doesn't require additional memory proportional to the size of the input. The space complexity is typically O(n) for the input and some extra variables.