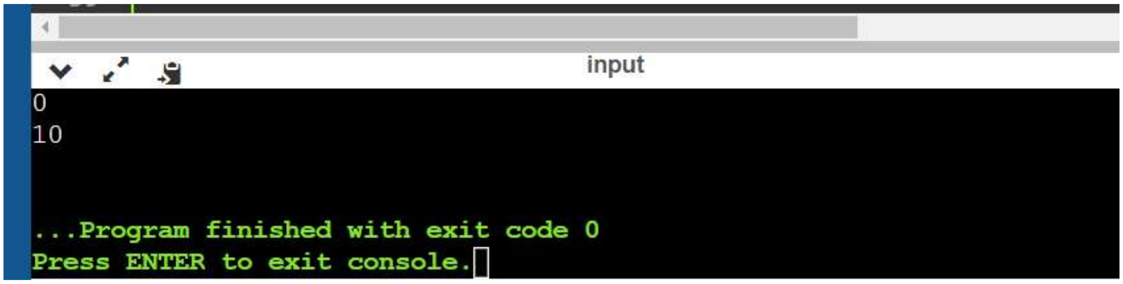
Problem 6: Given an array of N integers, count the inversion of thearray (using merge-sort).

What is an inversion of an array? Definition: for all i & j < size of array, if i < j then you have to find pair (A[i],A[j]) such that A[j] < A[i].

```
def mergeSortAndCountInversions(array):
    inversions = 0

    if len(array) <= 1:
        return inversions

    mid = len(array) // 2
    left_half = array[:mid]
    right_half = array[mid:]

    inversions += mergeSortAndCountInversions(left_half)
    inversions += mergeSortAndCountInversions(right_half)

    i = j = 0
    while i < len(left_half) and j < len(right_half):
        if left_half[i] > right_half[j]:
            inversions += mid - i
            array[i + j] = right_half[j]
            j += 1
        else:
            array[i + j] = left_half[i]
            i += 1

    while i < len(left_half):
        array[i + j] = left_half[i]
        i += 1

    while j < len(right_half):
        array[i + j] = right_half[j]
        j += 1

    return inversions

# Test cases
array1 = [1, 2, 3, 4, 5]
print(mergeSortAndCountInversions(array1))
array2 = [5, 4, 3, 2, 1]
print(mergeSortAndCountInversions(array2))
```



```
0
10


...Program finished with exit code 0
Press ENTER to exit console.
```