1.10 HEAP SORT

Question:

Given an array of integers nums, sort the array in ascending order and return it. You must solve the problem without using any built-in functions in O(nlog(n)) time complexity and with the smallest space complexity possible.

AIM:

To develop a efficient sorting solution.

ALGORITHM:

1. Build a Max Heap from the input array.
2. Swap the root (largest value) with the last element.
3. Reduce the heap size and heapify the root.
4. Repeat until the heap is reduced to 1.

PROGRAM:

```
def merge_sort(arr):
    if len(arr) <= 1:
        return arr
    mid = len(arr) // 2
    left = merge_sort(arr[:mid])
    right = merge_sort(arr[mid:])
    return merge(left, right)

def merge(left, right):
    result = []
    i = j = 0
    while i < len(left) and j < len(right):
        if left[i] <= right[j]:
            result.append(left[i])
            i += 1
        else:
            result.append(right[j])
            j += 1
    result.extend(left[i:])
    result.extend(right[j:])
    return result

def run_custom_sort():
    arr = list(map(int, input("Enter array elements: ").split()))
    sorted_arr = merge_sort(arr)
    print("Sorted array:", sorted_arr)

run_custom_sort()
```

Input:

nums = [5, 2, 9, 1, 5, 6]

Output:

```
Enter array elements: 5 2 9 1 5 6
Sorted array: [1, 2, 5, 5, 6, 9]
>>>
```

RESULT:

Thus the program is successfully executed, and the output is verified.

PERFORMANCE ANALYSIS:

Time complexity:

- Best Case Time O(n log n)
- Average Case Time O(n log n)
- Worst Case Time O(n log n)

Space Complexity:

- O(1) (in-place sorting)