DAA

Name: Navya Sree Chagamreddy

ID: 1002197805

Hands On-4

Problem-0:

1) Code uploaded in Github.

2) The following are the recursive calls and function call stack, if we pass the 'n' value as '5'. fib(5):

Order of Recursive calls:

fib(5) → fib(4) → fib(3) → fib(2) → fib(1) → fib(0) → fib(1) → fib(2) → fib(1) → fib(0) → fib(3) → fib(2) → fib(1) → fib(0) → fib(1)

Problem 1:

1) Code Uploaded in Github

2) Let n1, n2, n3 are the lengths of array1, array2 and array3,

Let n = n1 + n2 + n3, be the total no. of elements in all three arrays

The Time Complexity of merge Sorted Arrays algorithm is:

1) Initialization of the O/P array: O(n)
2) Merging Loop : O(n)
3) Copying Remaining Elements : O(n)

∴ The overall time complexity is $O(n)$, where 'n' is the total no. of elements in the input arrays.

3) We can use a 'Priority Queue' (Min-heap) to keep track of the current smallest elements from the each array, which reduces the time complexity of finding the smallest element to $O(\log k)$, where k is the no. of Arrays.

<u>Problem 2:</u>

1) code uploaded in Github.

2) Let;   n - no. of elements in the input array 'arr'

maxVal - max. value in the input array 'arr

The time complexity of the algorithm;

$$O\left(\sum_{i=1}^{n} 1\right).$$

Simplifies to → $O(n)$ is the overall time complexity of the 'removeDuplicates'.

3) - 'Hash set' can be used to store the unique elements while iterating through the Array.

- 'Hash set' provides O(1) insertion & lookup time complexity, which improve the pso performance.