Practical 3

Task 1 - Merge sort : (20 marks)

Merge Sort is an efficient, stable, comparison-based, divide-and-conquer sorting algorithm. The main idea is to divide the array into two halves, recursively sort each half, and then merge the two sorted halves back together.

Complete the function mergeSort(int arr[], int left, int right), which takes an integer array arr and two indices left and right. This function should sort the array from left to right using the Merge Sort algorithm. You will also need to complete the helper function void merge(int arr[], int left, int mid, int right) to merge the two halves.

Task 2 - valid Mountain Array: (20 marks)

A mountain array is defined as an array where:

1. The length of the array is at least 3.
2. There exists an index i (where 0 < i < arr.length - 1) such that:
    a. The elements strictly increase from the start of the array to the element at index i.
    b. The elements strictly decrease from the element at index i to the end of the array.

Function Signature:
Create a function bool validMountainArray(int arr[], int size) that takes an integer array arr and its size and returns true if it is a valid mountain array; otherwise, it returns false.

Example:

Input: arr = [2, 1]
Output: false
Explanation: The array length is less than 3.

Input: arr = [3, 5, 5]
Output: false
Explanation: The array does not strictly increase and then strictly decrease.

Input: arr = [0, 3, 2, 1]
Output: true
Explanation: The array increases to 3 and then decreases to 1.