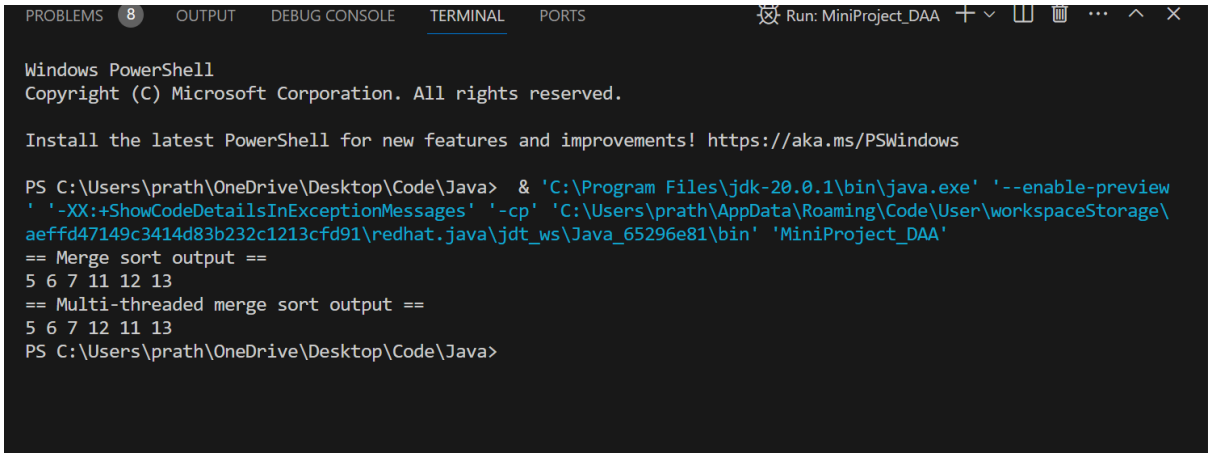
```java
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.TimeUnit;

public class MiniProject_DAA {
    //Merge Sort
    static void sort(int arr[], int l, int r) {
        if (l < r) {

            // Find the middle point
            int m = l + (r - l) / 2;

            // Sort first and second halves
            sort(arr, l, m);
            sort(arr, m + 1, r);

            // Merge the sorted halves
            merge(arr, l, m, r);
        }
    }

    public static void main(String[] args) throws InterruptedException {
        int[] arr = { 12, 11, 13, 5, 6, 7 };
        int[] arr1 ={ 12, 11, 13, 5, 6, 7 };

        sort(arr1, 0, arr1.length-1);

        ExecutorService executor =
Executors.newFixedThreadPool(Runtime.getRuntime().availableProcessors());
        mergeSort(arr, 0, arr.length - 1, executor);
        executor.shutdown();
        executor.awaitTermination(1, TimeUnit.SECONDS);

        System.out.println("== Merge sort output ==");
        for (int num : arr1) {
            System.out.print(num + " ");
        }
        System.out.println();
        System.out.println("== Multi-threaded merge sort output ==");
        for(int num: arr){
            System.out.print(num + " ");
        }
    }

    static void mergeSort(int[] arr, int left, int right, ExecutorService
executor) {
        if (left < right) {
            int mid = left + (right - left) / 2;
```

```
            executor.submit(() -> mergeSort(arr, left, mid, executor));
            executor.submit(() -> mergeSort(arr, mid + 1, right, executor));
            merge(arr, left, mid, right);
        }
    }

    static void merge(int[] arr, int left, int mid, int right) {
        int n1 = mid - left + 1;
        int n2 = right - mid;

        int[] leftArr = new int[n1];
        int[] rightArr = new int[n2];

        for (int i = 0; i < n1; i++) {
            leftArr[i] = arr[left + i];
        }
        for (int i = 0; i < n2; i++) {
            rightArr[i] = arr[mid + 1 + i];
        }

        int i = 0, j = 0, k = left;

        while (i < n1 && j < n2) {
            if (leftArr[i] <= rightArr[j]) {
                arr[k] = leftArr[i];
                i++;
            } else {
                arr[k] = rightArr[j];
                j++;
            }
            k++;
        }

        while (i < n1) {
            arr[k] = leftArr[i];
            i++;
            k++;
        }

        while (j < n2) {
            arr[k] = rightArr[j];
            j++;
            k++;
        }
    }
}
```


PROBLEMS 8
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Run: MiniProject_DAA
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\prath\OneDrive\Desktop\Code\Java> & 'C:\Program Files\jdk-20.0.1\bin\java.exe' '--enable-preview ' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\prath\AppData\Roaming\Code\User\workspaceStorage\aeffd47149c3414d83b232c1213cfd91\redhat.java\jdt_ws\Java_65296e81\bin' 'MiniProject_DAA'
== Merge sort output ==
5 6 7 11 12 13
== Multi-threaded merge sort output ==
5 6 7 12 11 13
PS C:\Users\prath\OneDrive\Desktop\Code\Java>