PROGRAM STRUCTURES AND ALGORITHMS

FALL 2021
Assignment - 5(Parallel Sort)
Shashwat Shrey -- 002128122

Implementation for testing different number of threads

```
public static int threadCount = 3;
public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
private static CompletableFuture<int[]> parsort(int[] array, int from, int to) {
    return CompletableFuture.supplyAsync(
            () -> {
                int[] result = new int[to - from];
                // TO IMPLEMENT
                System.arraycopy(array, from, result, destPos: 0, result.length);
                sort(result, from: 0, to: to - from);
                return result;
            },myPool
    );
}
```

Main method modified to take input from the user for cutoff and threadcount

```
public static void main(String[] args) {
    //System.out.println("The Program is running");
    ParSort.threadCount = Integer.parseInt(args[0]);
    ParSort.cutoff = Integer.parseInt(args[1]);
    processArgs(args);
    System.out.println("Degree of parallelism: " + ForkJoinPool.getCommonPoolParallelism());
    Random random = new Random();
    int[] array = new int[2000000];
    ArrayList<Long> timeList = new ArrayList<>();
    for (int j = 50; j < 100; j++) {
        ParSort.cutoff = 10000 * (j + 1);
        // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
        long time;
        long startTime = System.currentTimeMillis();
        for (int t = 0; t < 10; t++) {
            for (int i = 0; i < array.length; i++) array[i] = random.nextInt( bound: 10000000);
            ParSort.sort(array, from: 0, array.length);
        }
        long endTime = System.currentTimeMillis();
        time = (endTime - startTime);
        timeList.add(time);
```

Output :

	Cutoff/Threads	Cutoff/Threads	Cutoff/Threads	Cutoff/Threads
	1000/5	1000/10	500/3	1000/7
510000	1071	888	1492	1023
520000	883	692	1197	660
530000	843	655	1162	619
540000	837	638	1150	588
550000	852	637	1157	585
560000	837	630	1164	584
570000	844	721	1254	585
580000	901	654	1176	588
590000	880	660	1213	582
600000	896	636	1184	579
610000	861	693	1175	578
620000	842	632	1165	581
630000	857	667	1164	587
640000	846	632	1185	577
650000	844	632	1169	576
660000	845	644	1159	581
670000	845	644	1171	584
680000	845	650	1165	577
690000	844	639	1156	578
700000	846	631	1158	581
710000	848	637	1158	579
720000	848	634	1164	578
730000	848	633	1168	583
740000	851	655	1182	580
750000	845	640	1208	580
760000	846	631	1200	576
770000	845	722	1201	580
780000	844	725	1215	578
790000	850	678	1238	576
800000	846	669	1230	577
810000	848	665	1208	578
820000	846	670	1209	577
830000	846	646	1209	580
840000	844	652	1216	578
850000	846	632	1224	583
860000	849	638	1239	579
870000	862	656	1219	581
880000	848	635	1218	577