Program Structures and Algorithms
Spring 2023(SEC –03)

NAME: Patel Dhruv Rajeshkumar
NUID: 002928881

Task: Compare parallel sort results with default sort provided by Java. Set multiple cut-off values to check which after which value parallel sort with multi-threading should be applied. Here, I have taken three different array size, which are 50000, 100000 and 1000000. For each example, cut-off values are divided in 5% of the array size.

Relationship Conclusion: As the array size increases, the more threads help sort the data faster than less threads. Also, the cut-off value between 10% to 20% gives fastest time with 8 to 16 threads.

Evidence to support that conclusion and graphical representation:
The cut-off values are in gap of 5% of the array size.

For array size 50000: Best results were found with cut-off 20000 (40%) and 4,8 and 16 threads.
Fir array size 100000: Best results were found with cut-off 25000 (25%) and 8 threads.
For array size 1000000: Best results were found with cut-off 50000 (5%) and 32 threads.

Thus, from these values, we can say that as the size of data increases, low cut-off value with more threads will give the best results.

Here, horizontal headings represents threads and Vertical titles represents the cut-off value.

Array Size: 50000					
	2	**4**	**8**	**16**	**32**
2500	73ms	20ms	19ms	17ms	16ms
5000	48ms	15ms	14ms	**12ms**	14ms
7500	41ms	14ms	14ms	13ms	13ms
10000	20ms	16ms	13ms	**12ms**	14ms
12500	18ms	15ms	14ms	14ms	13ms
15000	18ms	13ms	13ms	**12ms**	13ms
17500	25ms	13ms	13ms	12ms	13ms
20000	26ms	**12ms**	**12ms**	**12ms**	13ms
22500	19ms	13ms	13ms	13ms	13ms
25000	18ms	13ms	**12ms**	13ms	12ms
27500	18ms	17ms	17ms	16ms	17ms
30000	17ms	17ms	16ms	17ms	16ms
32500	18ms	16ms	16ms	16ms	16ms
35000	17ms	17ms	16ms	17ms	16ms

37500	23ms	16ms	16ms	16ms	16ms
40000	17ms	16ms	17ms	17ms	16ms
42500	16ms	17ms	17ms	16ms	17ms
45000	17ms	16ms	16ms	17ms	16ms
47500	16ms	17ms	17ms	16ms	16ms
50000	17ms	16ms	16ms	16ms	16ms
Array Size: 100000					
	2	**4**	**8**	**16**	**32**
5000	109ms	39ms	33ms	31ms	26ms
10000	70ms	28ms	26ms	25ms	27ms
15000	36ms	27ms	26ms	24ms	26ms
20000	35ms	27ms	25ms	24ms	24ms
25000	35ms	26ms	**23ms**	24ms	24ms
30000	50ms	25ms	25ms	25ms	25ms
35000	43ms	25ms	24ms	26ms	25ms
40000	38ms	26ms	25ms	25ms	25ms
45000	37ms	26ms	26ms	26ms	26ms
50000	38ms	25ms	25ms	25ms	25ms
55000	35ms	33ms	33ms	33ms	34ms
60000	34ms	34ms	34ms	33ms	33ms
65000	34ms	33ms	33ms	33ms	33ms
70000	33ms	34ms	34ms	33ms	32ms
75000	34ms	34ms	33ms	34ms	33ms
80000	34ms	33ms	33ms	33ms	33ms
85000	34ms	33ms	33ms	33ms	33ms
90000	33ms	33ms	33ms	33ms	33ms
95000	34ms	33ms	34ms	32ms	34ms
100000	34ms	33ms	33ms	33ms	33ms
Array Size: 1000000					
	2	**4**	**8**	**16**	**32**
50000	511ms	313ms	295ms	308ms	**238ms**
100000	313ms	267ms	243ms	243ms	240ms

150000	347ms	280ms	241ms	241ms	240ms
200000	353ms	280ms	243ms	241ms	242ms
250000	347ms	279ms	245ms	242ms	240ms
300000	391ms	268ms	267ms	265ms	267ms
350000	395ms	267ms	268ms	267ms	268ms
400000	396ms	267ms	266ms	266ms	267ms
450000	392ms	268ms	267ms	268ms	267ms
500000	389ms	266ms	266ms	267ms	266ms
550000	376ms	374ms	371ms	373ms	373ms
600000	375ms	374ms	374ms	375ms	372ms
650000	376ms	373ms	375ms	373ms	373ms
700000	375ms	374ms	374ms	372ms	374ms
750000	375ms	375ms	375ms	373ms	373ms
800000	374ms	374ms	374ms	373ms	373ms
850000	372ms	373ms	373ms	374ms	373ms
900000	375ms	373ms	374ms	373ms	372ms
950000	374ms	374ms	375ms	373ms	373ms
1000000	372ms	375ms	371ms	374ms	372ms