Program Structures and Algorithms

Spring 2023(SEC –1)

NAME: Foram Kamani

NUID: 002732551

Task:

Your task is to implement a parallel sorting algorithm such that each partition of the array is sorted in parallel. You will consider two different schemes for deciding whether to sort in parallel.

1. A cutoff (defaults to, say, 1000) which you will update according to the first argument in the command line when running. It is your job to experiment and come up with good value for this cutoff. If there are fewer elements to sort than the cutoff, then you should use the system sort instead.
2. Recursion depth or the number of available threads. Using this determination, you might decide on an ideal number (*t*) of separate threads (stick to powers of 2) and arrange for that number of partitions to be parallelized (by preventing recursion after the depth of *lg t* is reached).
3. An appropriate combination of these.

Relationship Conclusion:

1. As per performing analysis on varied sizes of arrays for several different threads ranging from 2 to 64, we can determine that the 4 threads would be a desirable choice and using any thread beyond 4 would be waste of resources.

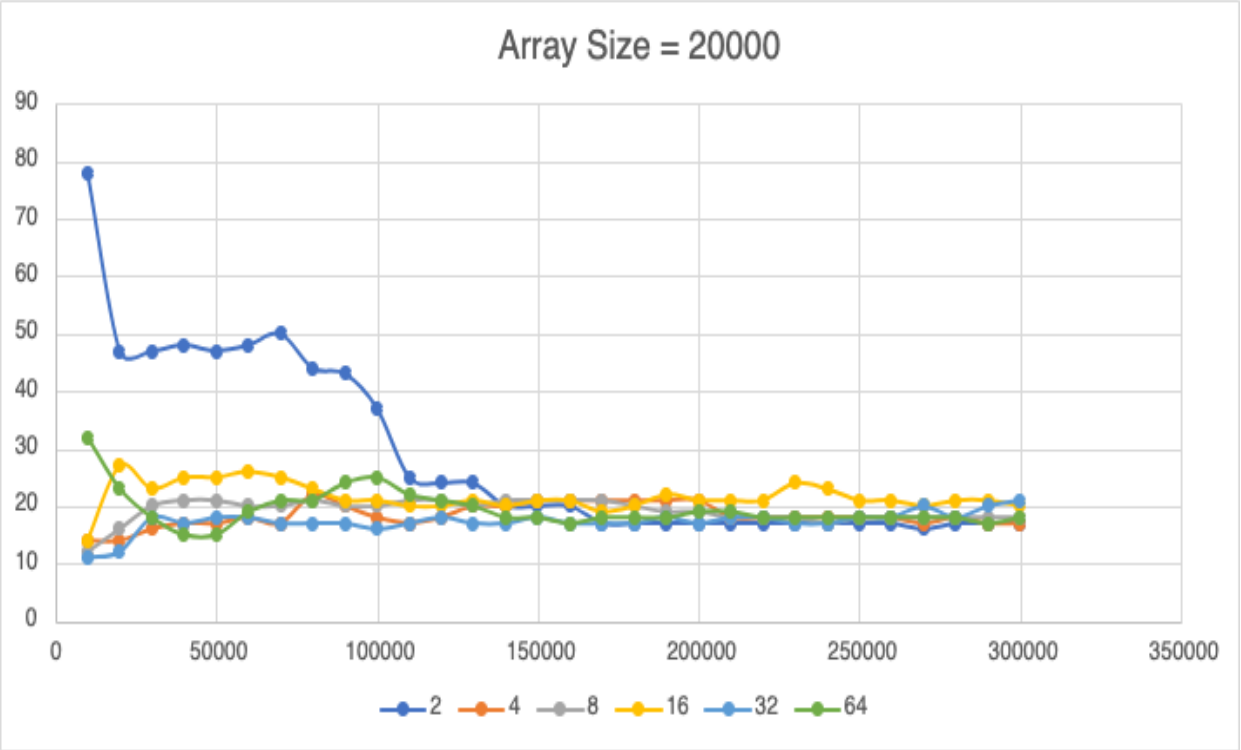
The following conclusion was drawn for the thread count (T) and recursion depth (d):

$T = 2^d$, lg (size of array/cutoff)

And, at any given time, the depth will not be more than maximum depth as the array reaches the cutoff.

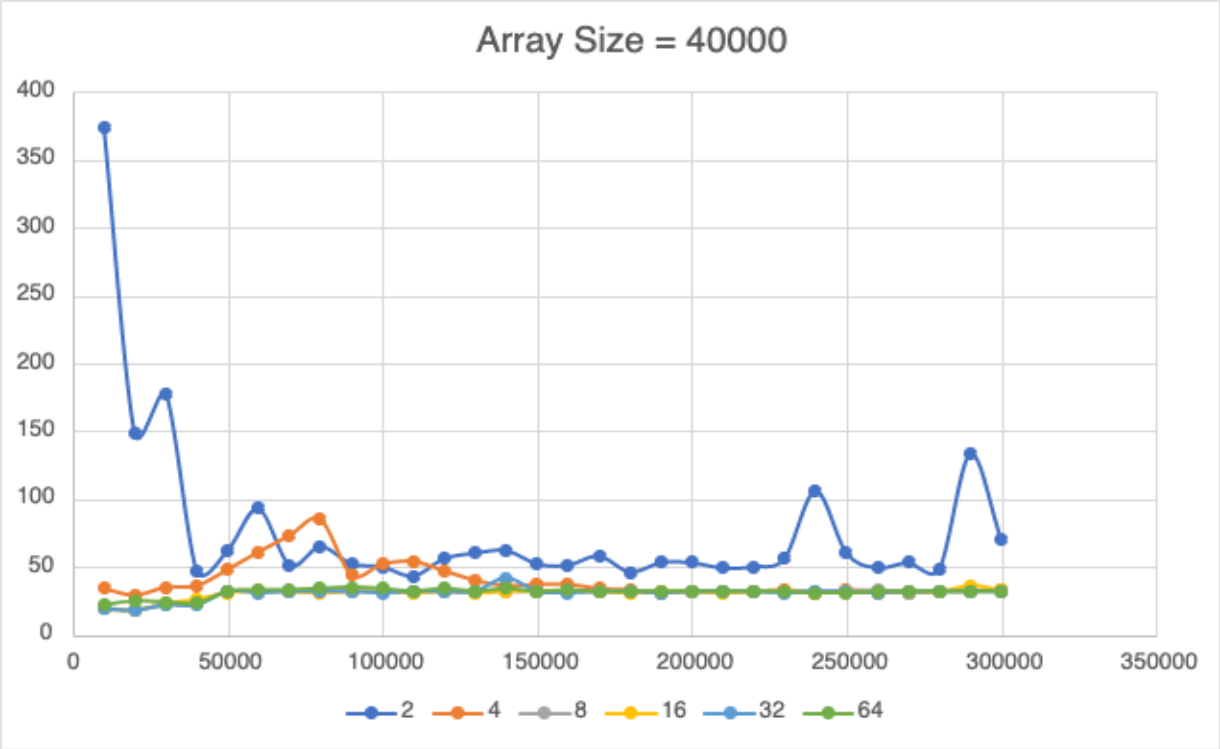
Graphical Representation:

Array Size = 20000						
cutoff	**2**	**4**	**8**	**16**	**32**	**64**
10000	78	14	12	14	11	32
20000	47	14	16	27	12	23
30000	47	16	20	23	18	18
40000	48	17	21	25	17	15
50000	47	17	21	25	18	15
60000	48	18	20	26	18	19
70000	50	17	20	25	17	21
80000	44	22	21	23	17	21
90000	43	20	20	21	17	24
100000	37	18	20	21	16	25
110000	25	17	21	20	17	22
120000	24	18	21	20	18	21
130000	24	20	20	21	17	20
140000	20	20	21	20	17	18
150000	20	21	21	21	18	18
160000	20	21	21	21	17	17
170000	17	21	21	19	17	18
180000	17	21	20	20	17	18
190000	17	21	19	22	18	18
200000	17	21	19	21	17	19
210000	17	18	18	21	18	19
220000	17	18	18	21	18	18
230000	17	18	18	24	17	18
240000	17	18	17	23	17	18
250000	17	18	18	21	18	18
260000	17	18	18	21	18	18
270000	16	17	18	20	20	18
280000	17	18	18	21	18	18
290000	17	17	18	21	20	17
300000	17	17	18	20	21	18



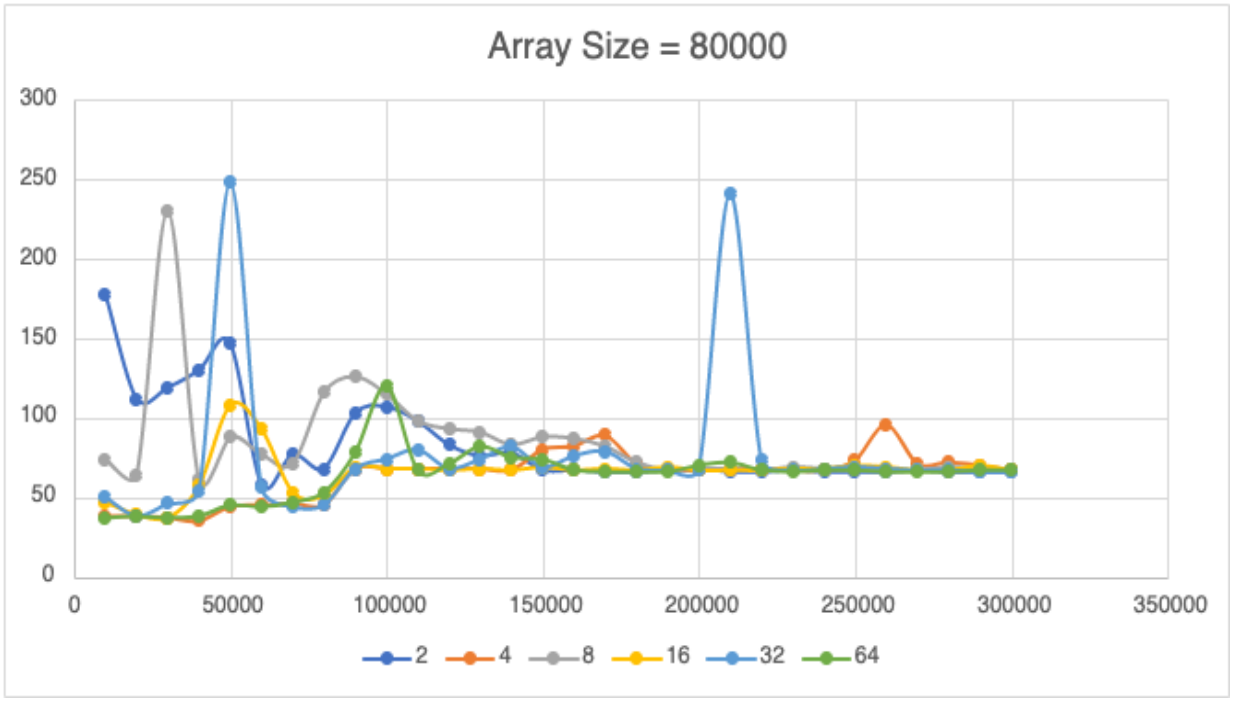
Array Size = 40000						
cutoff	**2**	**4**	**8**	**16**	**32**	**64**
10000	374	34	20	19	19	22
20000	148	29	18	18	18	25
30000	178	35	22	23	22	24
40000	47	36	23	26	22	23
50000	62	48	32	31	32	32
60000	93	61	32	32	31	33
70000	51	73	32	32	32	33
80000	65	85	32	31	32	34
90000	52	44	32	32	32	35
100000	50	52	32	32	31	34
110000	43	54	32	31	32	32
120000	56	47	32	32	32	34
130000	60	40	32	31	32	32
140000	62	36	32	32	42	34
150000	52	37	32	32	32	32
160000	51	37	32	32	31	33
170000	58	34	32	32	32	32

180000	46	33	32	31	32	32
190000	53	31	32	32	31	32
200000	53	32	32	32	32	32
210000	49	32	32	31	32	32
220000	50	32	32	32	32	32
230000	56	33	32	32	31	32
240000	106	31	32	31	32	31
250000	60	33	32	32	32	31
260000	49	32	33	31	31	32
270000	53	31	31	32	32	32
280000	48	32	32	32	32	32
290000	133	32	32	36	32	32
300000	70	33	32	33	32	32



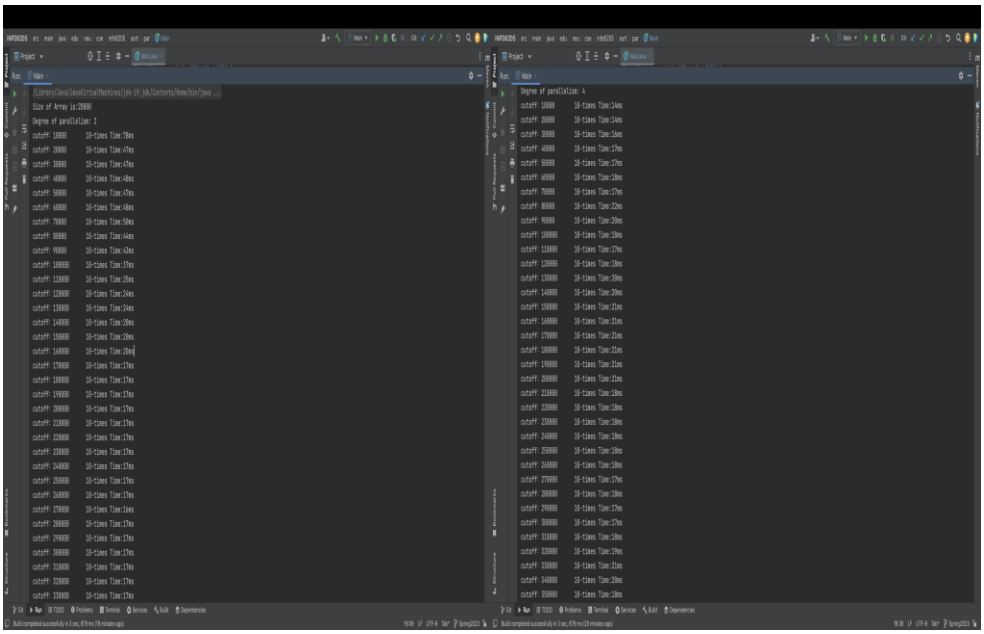
Array Size = 80000						
cutoff	**2**	**4**	**8**	**16**	**32**	**64**
10000	178	38	74	47	50	37
20000	112	38	64	39	38	38

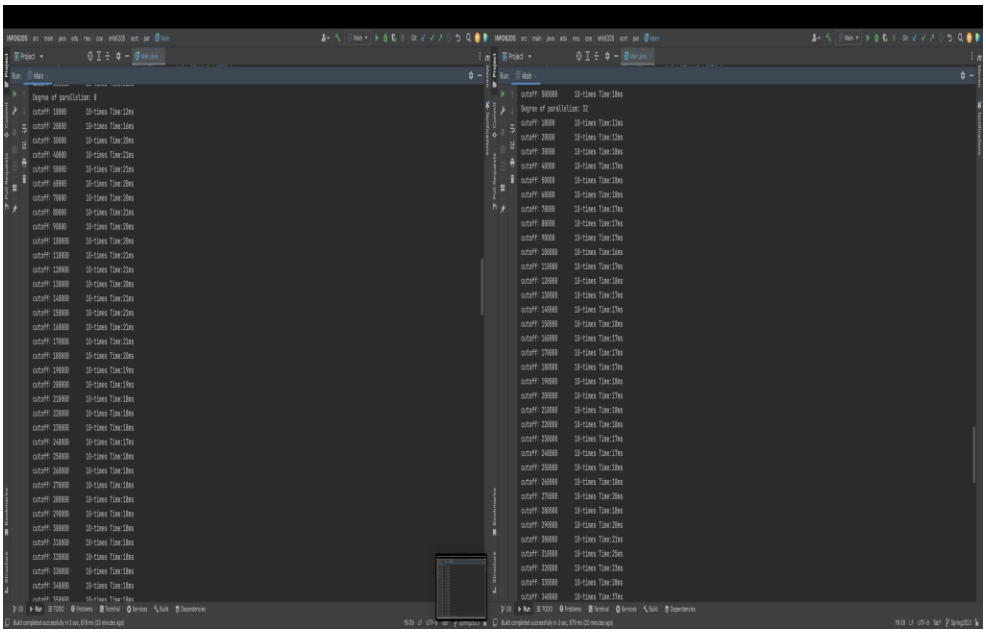
30000	119	37	230	37	46	37
40000	130	35	61	57	54	38
50000	147	44	88	108	248	45
60000	58	45	77	93	56	44
70000	77	46	71	53	44	47
80000	68	45	116	51	45	53
90000	103	68	126	69	68	78
100000	107	68	115	68	74	120
110000	98	68	98	68	80	67
120000	83	68	93	67	68	71
130000	76	68	91	68	74	82
140000	78	67	83	67	82	75
150000	67	80	88	69	69	74
160000	67	82	87	67	76	67
170000	66	89	82	68	79	66
180000	66	71	72	67	68	66
190000	67	67	67	69	67	66
200000	68	67	68	67	68	70
210000	66	68	68	67	241	72
220000	66	68	67	67	74	67
230000	67	67	69	68	67	66
240000	66	67	68	68	68	67
250000	66	73	67	70	69	67
260000	66	96	67	69	68	66
270000	67	71	67	67	67	66
280000	66	72	69	67	67	66
290000	66	70	68	70	66	67
300000	66	67	68	67	66	67

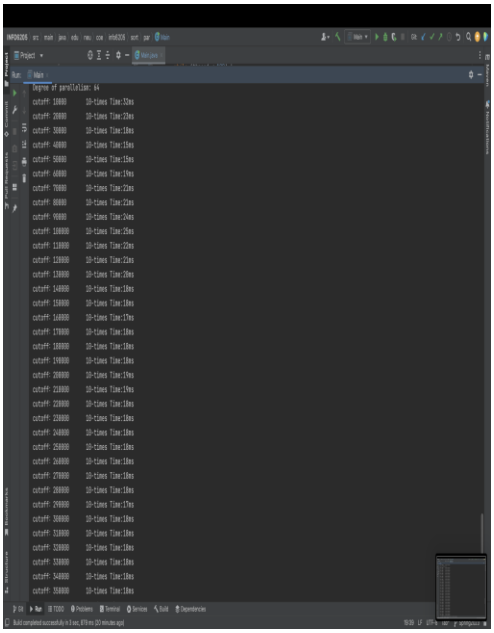


Evidence to support Conclusion:

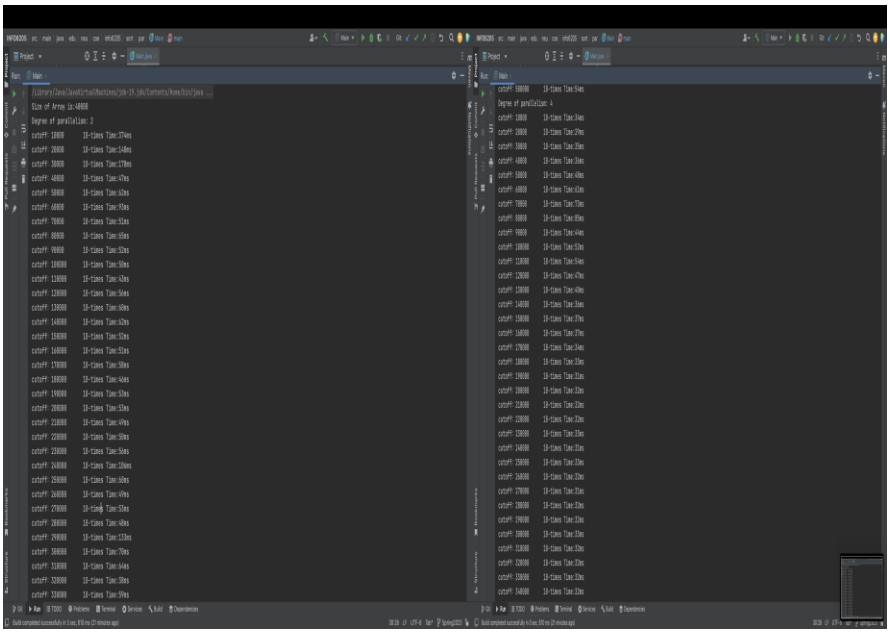
Array Size = 20000

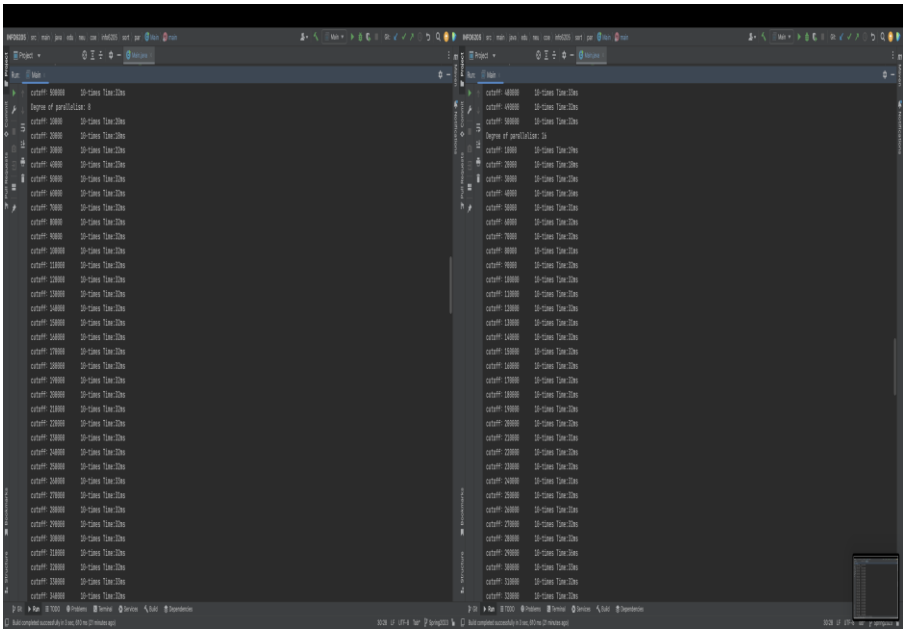


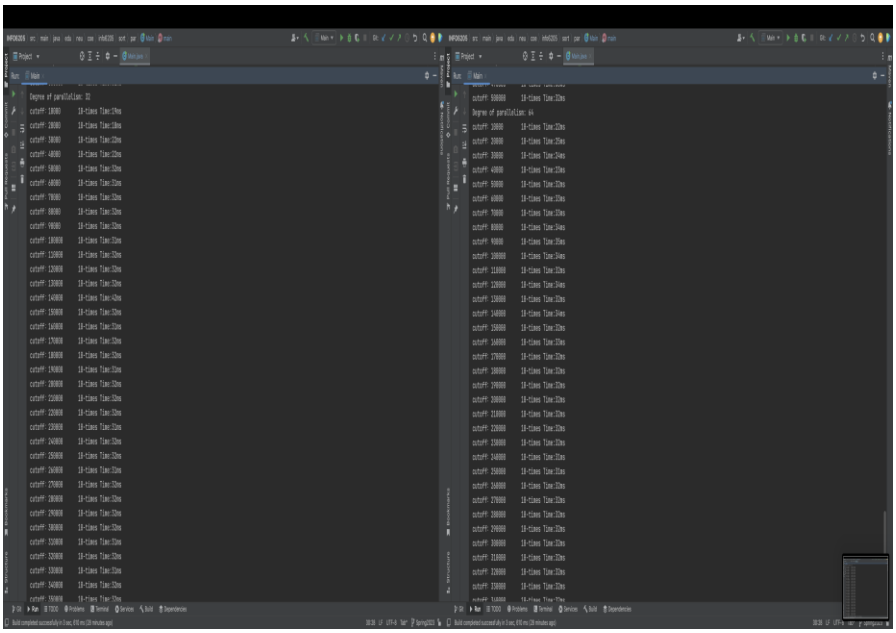




Array Size = 40000







Array Size = 80000