**NAME: SHREYA BALIGA**

**NUID: 002795178**

# PARALLEL SORTING

## TASK:

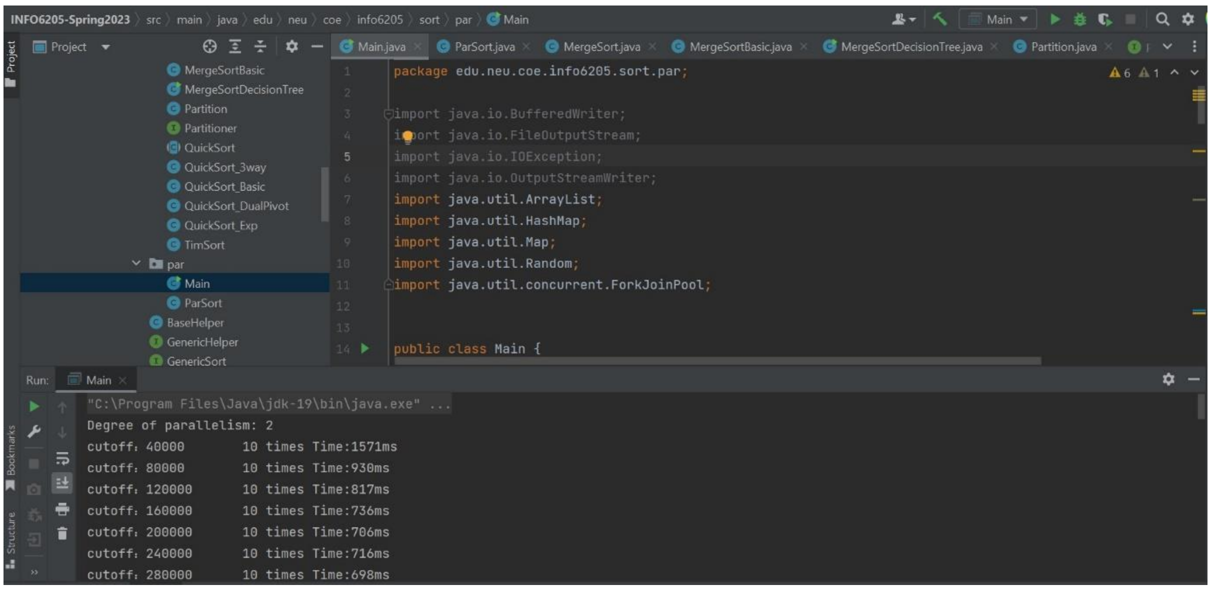
Your task is to implement a parallel sorting algorithm such that each partition of the array is sorted in parallel. You will consider two different schemes for deciding whether to sort in parallel.

1. A cutoff (defaults to, say, 1000) which you will update according to the first argument in the command line when running. It's your job to experiment and come up with a 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.

## OUTPUT:

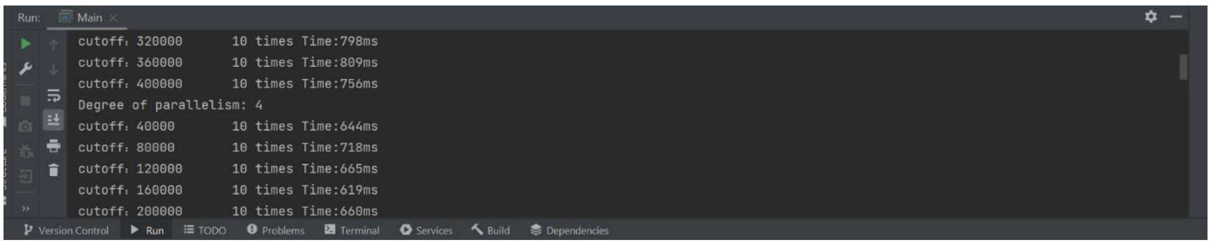
The below screenshot displays the output:

**for arraySize = 800000 and cut_off value=40000**.

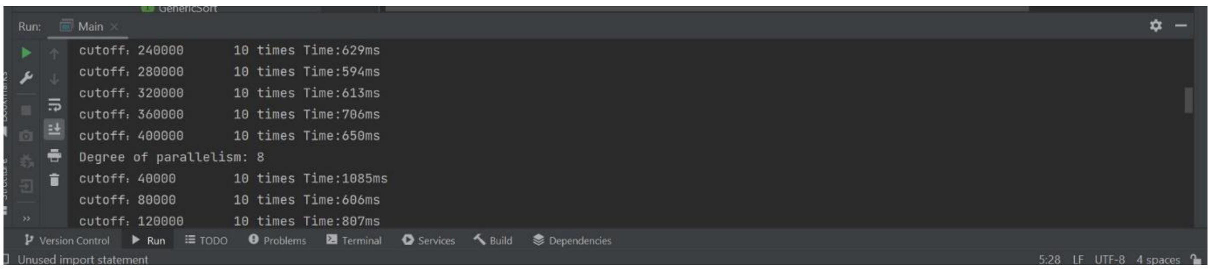


```
package edu.neu.coe.info6205.sort.par;

import java.io.BufferedWriter;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Random;
import java.util.concurrent.ForkJoinPool;


public class Main {
```

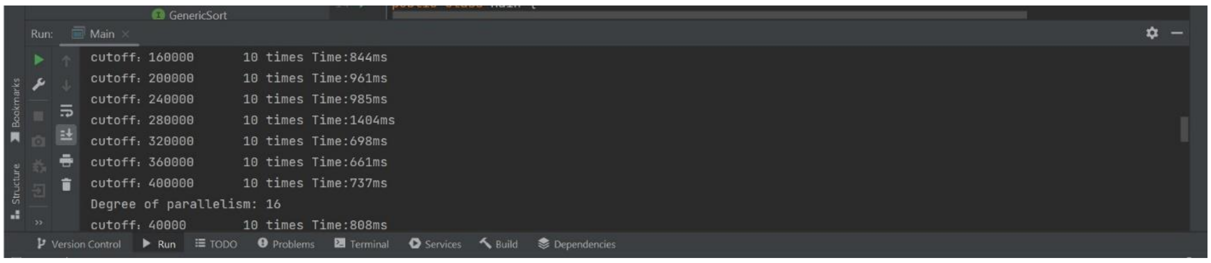
```
"C:\Program Files\Java\jdk-19\bin\java.exe" ...
Degree of parallelism: 2
cutoff: 40000       10 times Time:1571ms
cutoff: 80000       10 times Time:930ms
cutoff: 120000      10 times Time:817ms
cutoff: 160000      10 times Time:736ms
cutoff: 200000      10 times Time:706ms
cutoff: 240000      10 times Time:716ms
cutoff: 280000      10 times Time:698ms
```



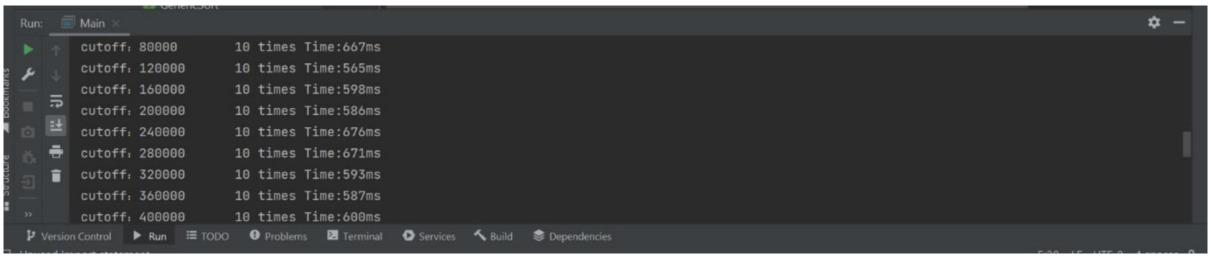
```
cutoff: 320000      10 times Time:798ms
cutoff: 360000      10 times Time:809ms
cutoff: 400000      10 times Time:756ms
Degree of parallelism: 4
cutoff: 40000       10 times Time:644ms
cutoff: 80000       10 times Time:718ms
cutoff: 120000      10 times Time:665ms
cutoff: 160000      10 times Time:619ms
cutoff: 200000      10 times Time:660ms
```



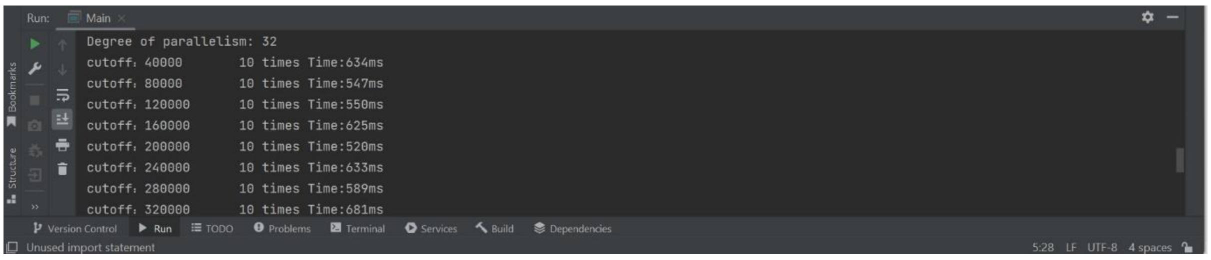
```
cutoff: 240000      10 times Time:629ms
cutoff: 280000      10 times Time:594ms
cutoff: 320000      10 times Time:613ms
cutoff: 360000      10 times Time:706ms
cutoff: 400000      10 times Time:650ms
Degree of parallelism: 8
cutoff: 40000       10 times Time:1085ms
cutoff: 80000       10 times Time:606ms
cutoff: 120000      10 times Time:807ms
```



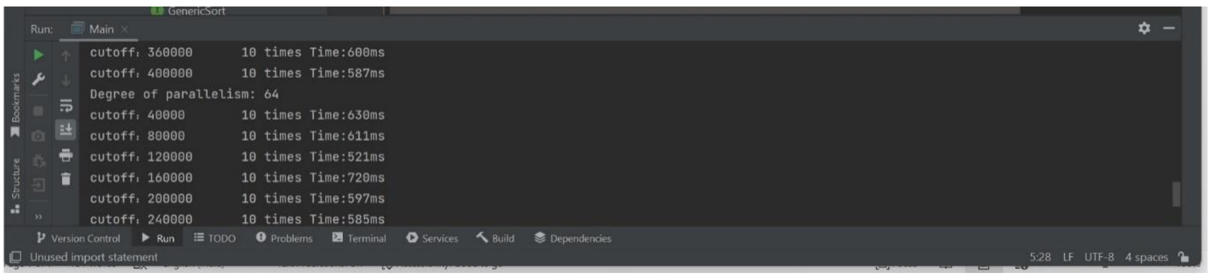
```
cutoff: 160000      10 times Time:844ms
cutoff: 200000      10 times Time:961ms
cutoff: 240000      10 times Time:985ms
cutoff: 280000      10 times Time:1404ms
cutoff: 320000      10 times Time:698ms
cutoff: 360000      10 times Time:661ms
cutoff: 400000      10 times Time:737ms
Degree of parallelism: 16
cutoff: 40000       10 times Time:808ms
```



```
cutoff: 80000       10 times Time:667ms
cutoff: 120000      10 times Time:565ms
cutoff: 160000      10 times Time:598ms
cutoff: 200000      10 times Time:586ms
cutoff: 240000      10 times Time:676ms
cutoff: 280000      10 times Time:671ms
cutoff: 320000      10 times Time:593ms
cutoff: 360000      10 times Time:587ms
cutoff: 400000      10 times Time:600ms
```

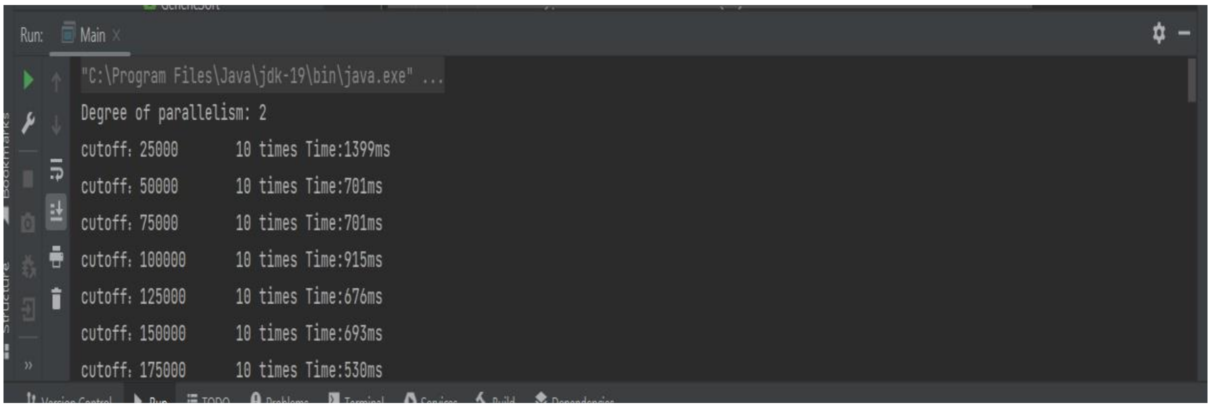


```
Degree of parallelism: 32
cutoff: 40000       10 times Time:634ms
cutoff: 80000       10 times Time:547ms
cutoff: 120000      10 times Time:550ms
cutoff: 160000      10 times Time:625ms
cutoff: 200000      10 times Time:520ms
cutoff: 240000      10 times Time:633ms
cutoff: 280000      10 times Time:589ms
cutoff: 320000      10 times Time:681ms
```

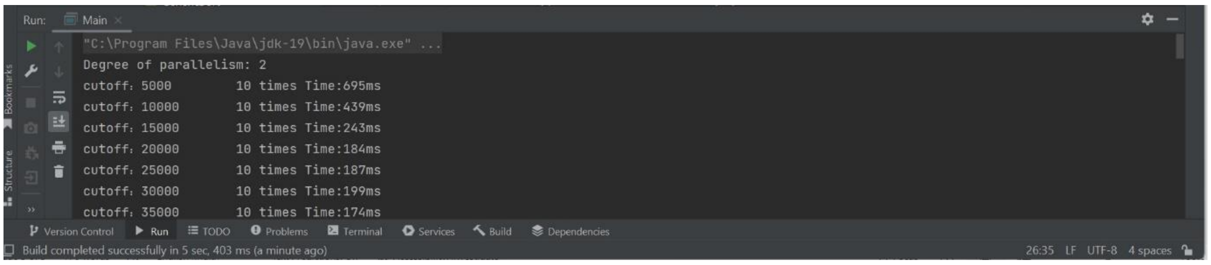


```
cutoff: 360000      10 times Time:600ms
cutoff: 400000      10 times Time:587ms
Degree of parallelism: 64
cutoff: 40000       10 times Time:630ms
cutoff: 80000       10 times Time:611ms
cutoff: 120000      10 times Time:521ms
cutoff: 160000      10 times Time:720ms
cutoff: 200000      10 times Time:597ms
cutoff: 240000      10 times Time:585ms
```

## For arraySize:500,000 and cut  off value 25000:


```
"C:\Program Files\Java\jdk-19\bin\java.exe" ...
Degree of parallelism: 2
cutoff: 25000       10 times Time:1399ms
cutoff: 50000       10 times Time:701ms
cutoff: 75000       10 times Time:701ms
cutoff: 100000      10 times Time:915ms
cutoff: 125000      10 times Time:676ms
cutoff: 150000      10 times Time:693ms
cutoff: 175000      10 times Time:530ms
```

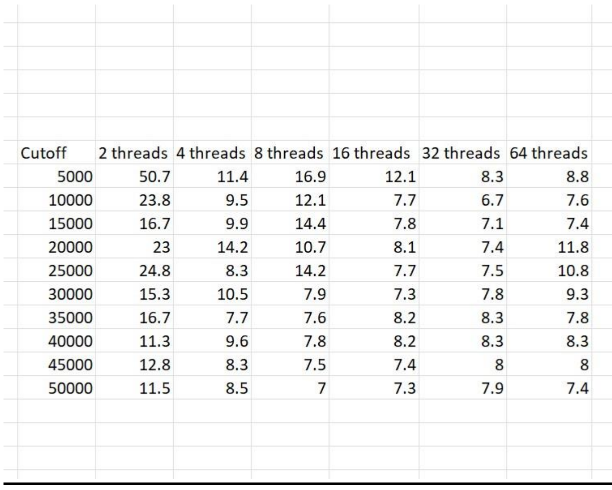
## For arraySize:100,000 and cut  off value 5000:


```
"C:\Program Files\Java\jdk-19\bin\java.exe" ...
Degree of parallelism: 2
cutoff: 5000        10 times Time:695ms
cutoff: 10000       10 times Time:439ms
cutoff: 15000       10 times Time:243ms
cutoff: 20000       10 times Time:184ms
cutoff: 25000       10 times Time:187ms
cutoff: 30000       10 times Time:199ms
cutoff: 35000       10 times Time:174ms
```
Build completed successfully in 5 sec, 403 ms (a minute ago)

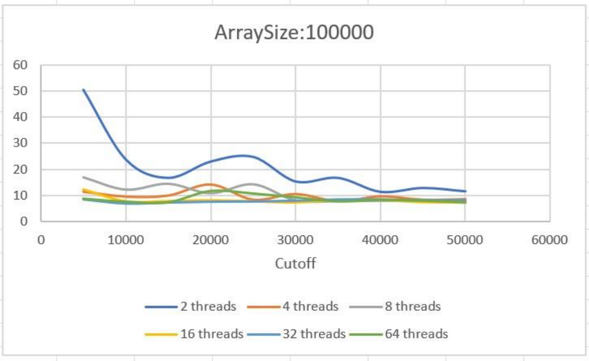
## EVIDENCE/OBSERVATIONS:

**Below is the table that displays the values of cutoff and the number of threads forked for various values of cutoff and array-size:**

## For array size:100000 and cutoff:5000

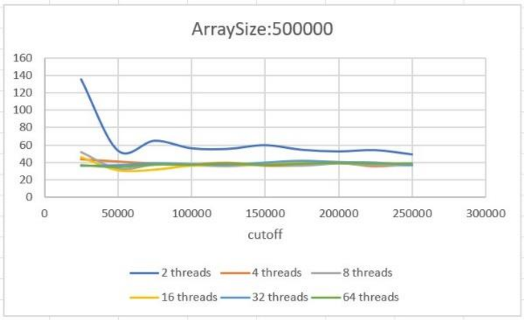


| Cutoff | 2 threads | 4 threads | 8 threads | 16 threads | 32 threads | 64 threads |
|---|---|---|---|---|---|---|
| 5000 | 50.7 | 11.4 | 16.9 | 12.1 | 8.3 | 8.8 |
| 10000 | 23.8 | 9.5 | 12.1 | 7.7 | 6.7 | 7.6 |
| 15000 | 16.7 | 9.9 | 14.4 | 7.8 | 7.1 | 7.4 |
| 20000 | 23 | 14.2 | 10.7 | 8.1 | 7.4 | 11.8 |
| 25000 | 24.8 | 8.3 | 14.2 | 7.7 | 7.5 | 10.8 |
| 30000 | 15.3 | 10.5 | 7.9 | 7.3 | 7.8 | 9.3 |
| 35000 | 16.7 | 7.7 | 7.6 | 8.2 | 8.3 | 7.8 |
| 40000 | 11.3 | 9.6 | 7.8 | 8.2 | 8.3 | 8.3 |
| 45000 | 12.8 | 8.3 | 7.5 | 7.4 | 8 | 8 |
| 50000 | 11.5 | 8.5 | 7 | 7.3 | 7.9 | 7.4 |



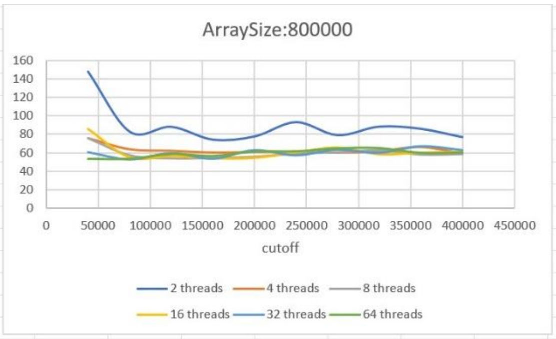
**<u>For arraySize:500,000 and cut_off value 25000:</u>**

| CutOff | 2 threads | 4 threads | 8 threads | 16 threads | 32 threads | 64 threads |
|---|---|---|---|---|---|---|
| 25000 | 135.5 | 43.2 | 52 | 45.6 | 35.5 | 36.5 |
| 50000 | 53.4 | 40.7 | 32.8 | 31.1 | 36.2 | 34.2 |
| 75000 | 64.7 | 38.6 | 37.5 | 31.7 | 38.5 | 37.3 |
| 100000 | 56 | 37.6 | 37 | 36.2 | 37.5 | 37 |
| 125000 | 55.3 | 39.4 | 35.7 | 36.6 | 36.6 | 38.5 |
| 150000 | 59.6 | 36.1 | 37.9 | 37.8 | 39.3 | 36.8 |
| 175000 | 54.3 | 36.6 | 36.4 | 40.7 | 41.3 | 38.3 |
| 200000 | 52.4 | 39.4 | 38.9 | 40.3 | 39.8 | 39.1 |
| 225000 | 53.9 | 35.3 | 36.8 | 39.1 | 39.3 | 38.4 |
| 250000 | 49 | 38.4 | 36.8 | 37.8 | 36.3 | 38.6 |



**<u>for arraySize = 800000 and cut_off value=40000</u>**:

| CutOff | 2 threads | 4 threads | 8 threads | 16 threads | 32 threads | 64 threads |
|---|---|---|---|---|---|---|
| 40000 | 148.2 | 75.7 | 75.7 | 86 | 60.3 | 53.2 |
| 80000 | 83.2 | 63.7 | 57.3 | 55.3 | 52.5 | 53.3 |
| 120000 | 88.5 | 62.1 | 54 | 56.7 | 59.3 | 58.6 |
| 160000 | 74.6 | 60.2 | 54.5 | 54.2 | 53.3 | 56.4 |
| 200000 | 78 | 60.8 | 55.7 | 54.8 | 62.5 | 61.4 |
| 240000 | 93.5 | 61.1 | 59 | 60.6 | 57 | 61.6 |
| 280000 | 79.4 | 60 | 60.5 | 65.8 | 63.1 | 64.9 |
| 320000 | 88.7 | 61.6 | 62.7 | 58.7 | 59.6 | 65.2 |
| 360000 | 86.3 | 66.2 | 58.1 | 60.4 | 66.7 | 59.9 |
| 400000 | 77.3 | 59.2 | 58.6 | 61.3 | 62.5 | 60.2 |



## <u>RELATIONSHIP CONCLUSION:</u>

Comparing the above graphs and data from the tables, we can conclude that after varying the cutoff values and number of threads for varied array sizes the number of threads greater than 8 does not improve the performance of the algorithm. Hence forking 8 threads is the best option. Apart from this, data collected from graphs represents that the optimal performance is seen around ~25% of the array size. With this, we can conclude that this leads to least algorithm performance time.

**<u>Code:</u>**

**<u>https://github.com/ShreyaBaliga2408/Parallel_Sorting</u>**