PROGRAM STRUCTURES & ALGORITHMS
INFO – 6205

Parallel Sorting

Assignment 5

SIDDHARTH RAWAT
002963295

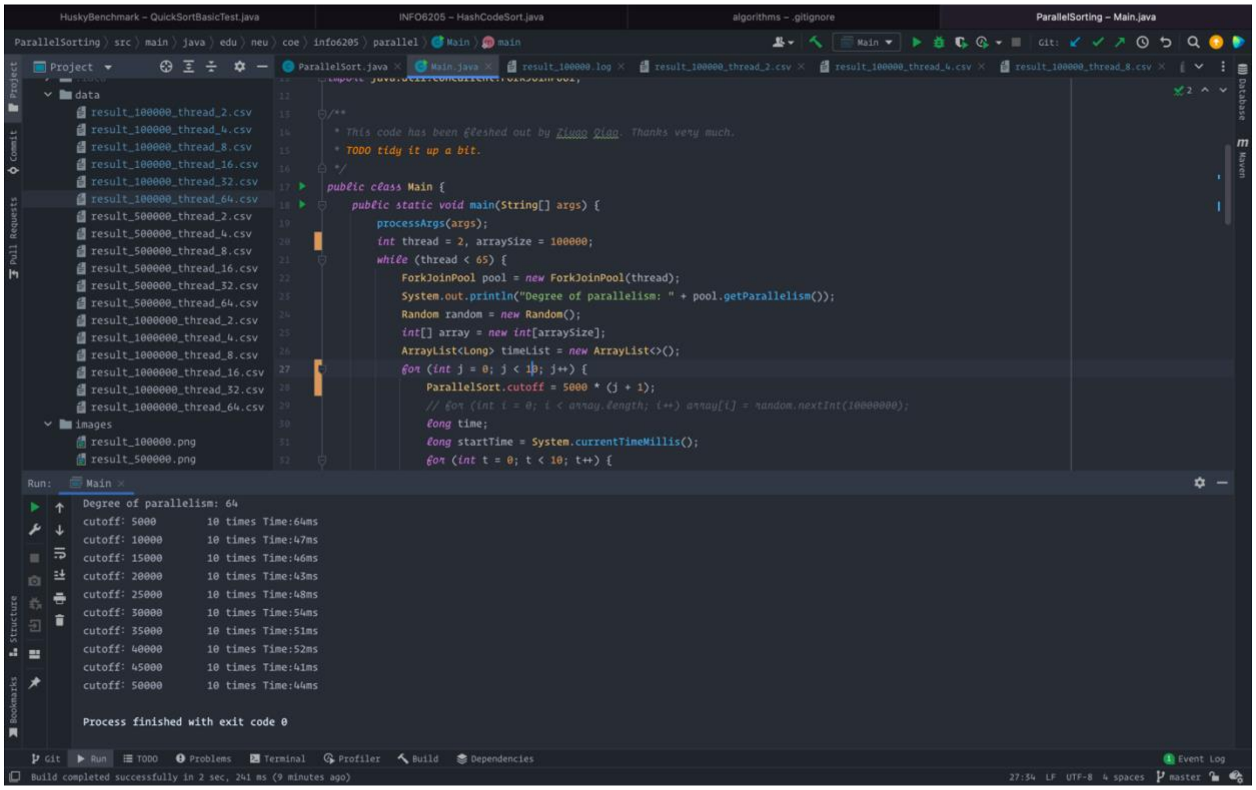
Table of Contents

1. Screenshots ***2***
2. Output ***5***
3. Observations ***10***
4. Code ***14***

1. Screenshots

Below is the screenshot of all the outputs for the given algorithm with varying array size and cutoff values:

<u>Array Size: 100,000 and Cutoff: 5000</u>



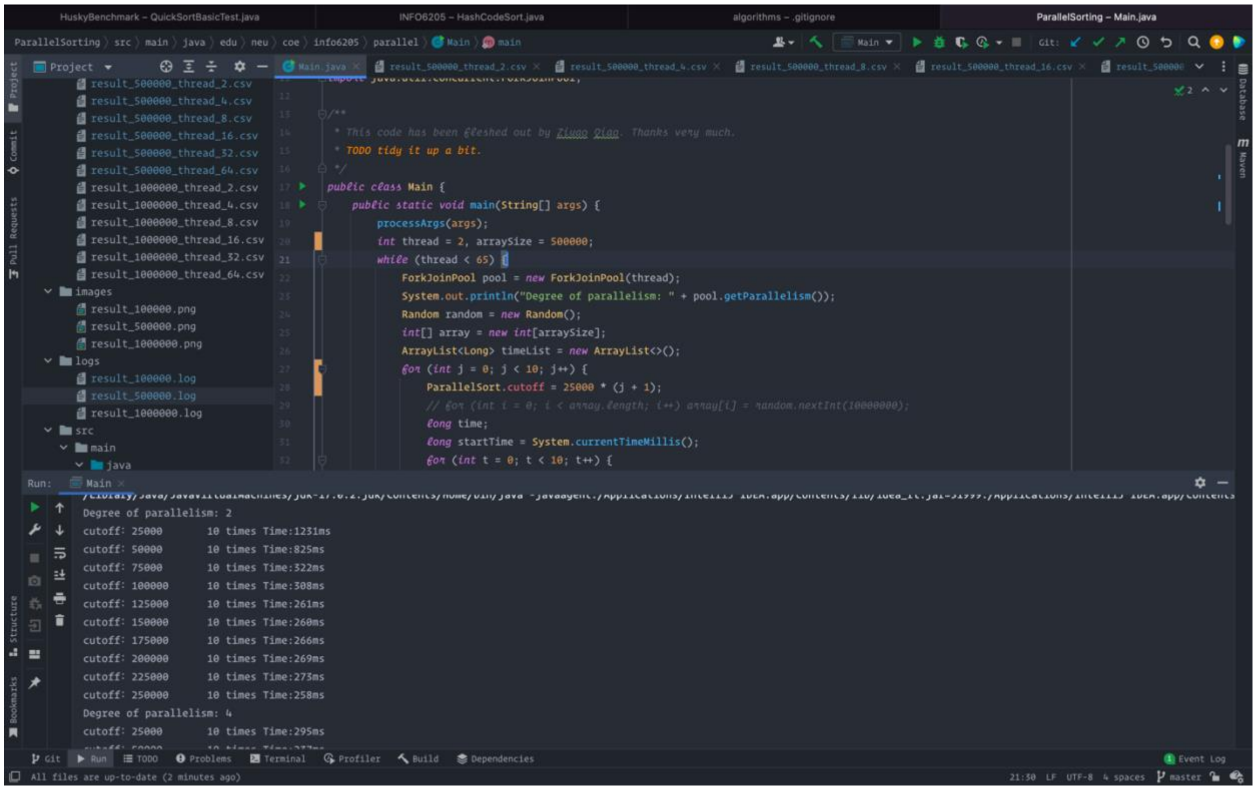
```java
/**
 * This code has been fleshed out by Ziyao Qiao. Thanks very much.
 * TODO tidy it up a bit.
 */
public class Main {
    public static void main(String[] args) {
        processArgs(args);
        int thread = 2, arraySize = 100000;
        while (thread < 65) {
            ForkJoinPool pool = new ForkJoinPool(thread);
            System.out.println("Degree of parallelism: " + pool.getParallelism());
            Random random = new Random();
            int[] array = new int[arraySize];
            ArrayList<Long> timeList = new ArrayList<>();
            for (int j = 0; j < 10; j++) {
                ParallelSort.cutoff = 5000 * (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++) {
```

```
Degree of parallelism: 64
cutoff: 5000        10 times Time:64ms
cutoff: 10000       10 times Time:47ms
cutoff: 15000       10 times Time:46ms
cutoff: 20000       10 times Time:43ms
cutoff: 25000       10 times Time:48ms
cutoff: 30000       10 times Time:54ms
cutoff: 35000       10 times Time:51ms
cutoff: 40000       10 times Time:52ms
cutoff: 45000       10 times Time:41ms
cutoff: 50000       10 times Time:44ms

Process finished with exit code 0
```

```
Build completed successfully in 2 sec, 241 ms (9 minutes ago)
```

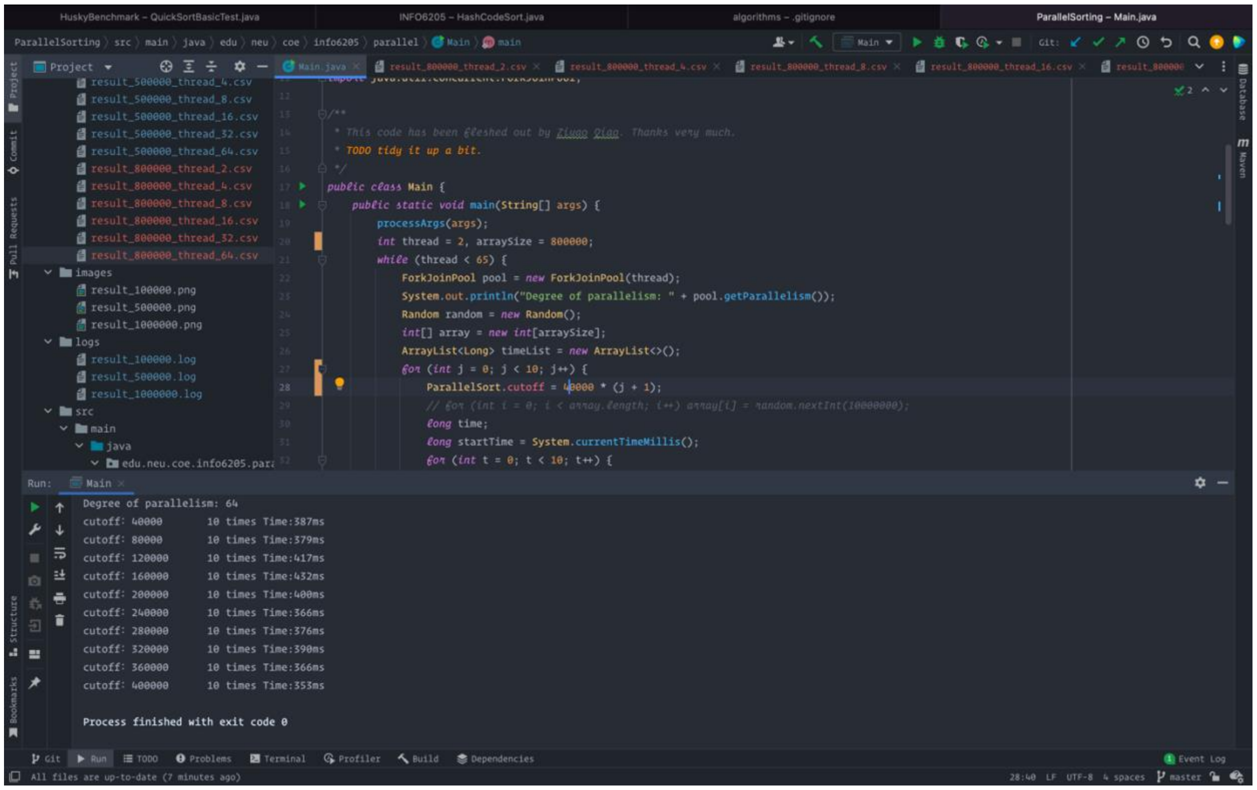
Array Size: 500,000 and Cutoff: 25,000



```java
/**
 * This code has been fleshed out by Ziyao Qiao. Thanks very much.
 * TODO tidy it up a bit.
 */
public class Main {
    public static void main(String[] args) {
        processArgs(args);
        int thread = 2, arraySize = 500000;
        while (thread < 65) {
            ForkJoinPool pool = new ForkJoinPool(thread);
            System.out.println("Degree of parallelism: " + pool.getParallelism());
            Random random = new Random();
            int[] array = new int[arraySize];
            ArrayList<Long> timeList = new ArrayList<>();
            for (int j = 0; j < 10; j++) {
                ParallelSort.cutoff = 25000 * (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++) {
```

```
Degree of parallelism: 2
cutoff: 25000       10 times Time:1231ms
cutoff: 50000       10 times Time:825ms
cutoff: 75000       10 times Time:322ms
cutoff: 100000      10 times Time:308ms
cutoff: 125000      10 times Time:261ms
cutoff: 150000      10 times Time:260ms
cutoff: 175000      10 times Time:266ms
cutoff: 200000      10 times Time:269ms
cutoff: 225000      10 times Time:273ms
cutoff: 250000      10 times Time:258ms
Degree of parallelism: 4
cutoff: 25000       10 times Time:295ms
```

Array Size: 800,000 and Cutoff: 40,000



```java
/**
 * This code has been fleshed out by Ziyao Qiao. Thanks very much.
 * TODO tidy it up a bit.
 */
public class Main {
    public static void main(String[] args) {
        processArgs(args);
        int thread = 2, arraySize = 800000;
        while (thread < 65) {
            ForkJoinPool pool = new ForkJoinPool(thread);
            System.out.println("Degree of parallelism: " + pool.getParallelism());
            Random random = new Random();
            int[] array = new int[arraySize];
            ArrayList<Long> timeList = new ArrayList<>();
            for (int j = 0; j < 10; j++) {
                ParallelSort.cutoff = 40000 * (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++) {
```

```
Degree of parallelism: 64
cutoff: 40000       10 times Time:387ms
cutoff: 80000       10 times Time:379ms
cutoff: 120000      10 times Time:417ms
cutoff: 160000      10 times Time:432ms
cutoff: 200000      10 times Time:400ms
cutoff: 240000      10 times Time:366ms
cutoff: 280000      10 times Time:376ms
cutoff: 320000      10 times Time:390ms
cutoff: 360000      10 times Time:366ms
cutoff: 400000      10 times Time:353ms

Process finished with exit code 0
```

2. Terminal Output

The below terminal output for the different array sizes and cutoff values to run the parallel sort algorithm.

```
========================================
Array Size: 100,000 and Cutoff: 5000
========================================

Degree of parallelism: 2
cutoff : 5000          10 times Time:415ms
cutoff : 10000         10 times Time:272ms
cutoff : 15000         10 times Time:229ms
cutoff : 20000         10 times Time:222ms
cutoff : 25000         10 times Time:180ms
cutoff : 30000         10 times Time:99ms
cutoff : 35000         10 times Time:51ms
cutoff : 40000         10 times Time:76ms
cutoff : 45000         10 times Time:59ms
cutoff : 50000         10 times Time:65ms
Degree of parallelism: 4
cutoff : 5000          10 times Time:80ms
cutoff : 10000         10 times Time:56ms
cutoff : 15000         10 times Time:67ms
cutoff : 20000         10 times Time:64ms
cutoff : 25000         10 times Time:66ms
cutoff : 30000         10 times Time:58ms
cutoff : 35000         10 times Time:105ms
cutoff : 40000         10 times Time:62ms
cutoff : 45000         10 times Time:53ms
cutoff : 50000         10 times Time:52ms
Degree of parallelism: 8
cutoff : 5000          10 times Time:72ms
cutoff : 10000         10 times Time:58ms
cutoff : 15000         10 times Time:44ms
cutoff : 20000         10 times Time:43ms
cutoff : 25000         10 times Time:43ms
cutoff : 30000         10 times Time:42ms
cutoff : 35000         10 times Time:47ms
cutoff : 40000         10 times Time:41ms
cutoff : 45000         10 times Time:40ms
cutoff : 50000         10 times Time:46ms
Degree of parallelism: 16
cutoff : 5000          10 times Time:63ms
cutoff : 10000         10 times Time:60ms
cutoff : 15000         10 times Time:46ms
cutoff : 20000         10 times Time:54ms
cutoff : 25000         10 times Time:46ms
cutoff : 30000         10 times Time:40ms
cutoff : 35000         10 times Time:50ms
cutoff : 40000         10 times Time:39ms
cutoff : 45000         10 times Time:43ms
cutoff : 50000         10 times Time:43ms
Degree of parallelism: 32
cutoff : 5000          10 times Time:59ms
cutoff : 10000         10 times Time:50ms
cutoff : 15000         10 times Time:44ms
cutoff : 20000         10 times Time:41ms
cutoff : 25000         10 times Time:47ms
cutoff : 30000         10 times Time:46ms
cutoff : 35000         10 times Time:49ms
cutoff : 40000         10 times Time:40ms
cutoff : 45000         10 times Time:41ms
cutoff : 50000         10 times Time:49ms
Degree of parallelism: 64
cutoff : 5000          10 times Time:64ms
cutoff : 10000         10 times Time:47ms
cutoff : 15000         10 times Time:46ms
cutoff : 20000         10 times Time:43ms
cutoff : 25000         10 times Time:48ms
cutoff : 30000         10 times Time:54ms
cutoff : 35000         10 times Time:51ms
cutoff : 40000         10 times Time:52ms
cutoff : 45000         10 times Time:41ms
cutoff : 50000         10 times Time:44ms

Process finished with exit code 0

========================================
Array Size: 500,000 and Cutoff: 25,000
========================================

Degree of parallelism: 2
```

```
cutoff : 25000        10 times Time:1231ms
cutoff : 50000        10 times Time:825ms
cutoff : 75000        10 times Time:322ms
cutoff : 100000       10 times Time:308ms
cutoff : 125000       10 times Time:261ms
cutoff : 150000       10 times Time:260ms
cutoff : 175000       10 times Time:266ms
cutoff : 200000       10 times Time:269ms
cutoff : 225000       10 times Time:273ms
cutoff : 250000       10 times Time:258ms
Degree of parallelism: 4
cutoff : 25000        10 times Time:295ms
cutoff : 50000        10 times Time:237ms
cutoff : 75000        10 times Time:230ms
cutoff : 100000       10 times Time:260ms
cutoff : 125000       10 times Time:280ms
cutoff : 150000       10 times Time:214ms
cutoff : 175000       10 times Time:226ms
cutoff : 200000       10 times Time:211ms
cutoff : 225000       10 times Time:227ms
cutoff : 250000       10 times Time:231ms
Degree of parallelism: 8
cutoff : 25000        10 times Time:297ms
cutoff : 50000        10 times Time:226ms
cutoff : 75000        10 times Time:227ms
cutoff : 100000       10 times Time:231ms
cutoff : 125000       10 times Time:227ms
cutoff : 150000       10 times Time:227ms
cutoff : 175000       10 times Time:220ms
cutoff : 200000       10 times Time:225ms
cutoff : 225000       10 times Time:218ms
cutoff : 250000       10 times Time:278ms
Degree of parallelism: 16
cutoff : 25000        10 times Time:266ms
cutoff : 50000        10 times Time:228ms
cutoff : 75000        10 times Time:275ms
cutoff : 100000       10 times Time:228ms
cutoff : 125000       10 times Time:242ms
cutoff : 150000       10 times Time:233ms
cutoff : 175000       10 times Time:286ms
cutoff : 200000       10 times Time:229ms
```

```
cutoff：225000        10 times Time:241ms
cutoff：250000        10 times Time:220ms
Degree of parallelism: 32
cutoff：25000         10 times Time:243ms
cutoff：50000         10 times Time:278ms
cutoff：75000         10 times Time:240ms
cutoff：100000        10 times Time:225ms
cutoff：125000        10 times Time:231ms
cutoff：150000        10 times Time:238ms
cutoff：175000        10 times Time:240ms
cutoff：200000        10 times Time:227ms
cutoff：225000        10 times Time:225ms
cutoff：250000        10 times Time:226ms
Degree of parallelism: 64
cutoff：25000         10 times Time:242ms
cutoff：50000         10 times Time:235ms
cutoff：75000         10 times Time:232ms
cutoff：100000        10 times Time:227ms
cutoff：125000        10 times Time:232ms
cutoff：150000        10 times Time:220ms
cutoff：175000        10 times Time:227ms
cutoff：200000        10 times Time:218ms
cutoff：225000        10 times Time:226ms
cutoff：250000        10 times Time:222ms

Process finished with exit code 0

========================================
Array Size: 800,000 and Cutoff: 40,000
========================================
Degree of parallelism: 2
cutoff：40000         10 times Time:1226ms
cutoff：80000         10 times Time:553ms
cutoff：120000        10 times Time:456ms
cutoff：160000        10 times Time:424ms
cutoff：200000        10 times Time:442ms
cutoff：240000        10 times Time:456ms
cutoff：280000        10 times Time:441ms
cutoff：320000        10 times Time:433ms
cutoff：360000        10 times Time:478ms
cutoff：400000        10 times Time:439ms
Degree of parallelism: 4
```

```
cutoff : 40000         10 times Time:483ms
cutoff : 80000         10 times Time:380ms
cutoff : 120000        10 times Time:383ms
cutoff : 160000        10 times Time:506ms
cutoff : 200000        10 times Time:407ms
cutoff : 240000        10 times Time:375ms
cutoff : 280000        10 times Time:364ms
cutoff : 320000        10 times Time:429ms
cutoff : 360000        10 times Time:369ms
cutoff : 400000        10 times Time:360ms
Degree of parallelism: 8
cutoff : 40000         10 times Time:433ms
cutoff : 80000         10 times Time:383ms
cutoff : 120000        10 times Time:383ms
cutoff : 160000        10 times Time:370ms
cutoff : 200000        10 times Time:367ms
cutoff : 240000        10 times Time:372ms
cutoff : 280000        10 times Time:364ms
cutoff : 320000        10 times Time:357ms
cutoff : 360000        10 times Time:385ms
cutoff : 400000        10 times Time:376ms
Degree of parallelism: 16
cutoff : 40000         10 times Time:453ms
cutoff : 80000         10 times Time:442ms
cutoff : 120000        10 times Time:388ms
cutoff : 160000        10 times Time:372ms
cutoff : 200000        10 times Time:389ms
cutoff : 240000        10 times Time:351ms
cutoff : 280000        10 times Time:387ms
cutoff : 320000        10 times Time:544ms
cutoff : 360000        10 times Time:380ms
cutoff : 400000        10 times Time:377ms
Degree of parallelism: 32
cutoff : 40000         10 times Time:567ms
cutoff : 80000         10 times Time:386ms
cutoff : 120000        10 times Time:367ms
cutoff : 160000        10 times Time:405ms
cutoff : 200000        10 times Time:386ms
cutoff : 240000        10 times Time:358ms
cutoff : 280000        10 times Time:355ms
cutoff : 320000        10 times Time:350ms
cutoff : 360000        10 times Time:359ms
cutoff : 400000        10 times Time:372ms
Degree of parallelism: 64
cutoff : 40000         10 times Time:387ms
cutoff : 80000         10 times Time:379ms
cutoff : 120000        10 times Time:417ms
cutoff : 160000        10 times Time:432ms
cutoff : 200000        10 times Time:400ms
cutoff : 240000        10 times Time:366ms
cutoff : 280000        10 times Time:376ms
cutoff : 320000        10 times Time:390ms
cutoff : 360000        10 times Time:366ms
cutoff : 400000        10 times Time:353ms

Process finished with exit code 0
```

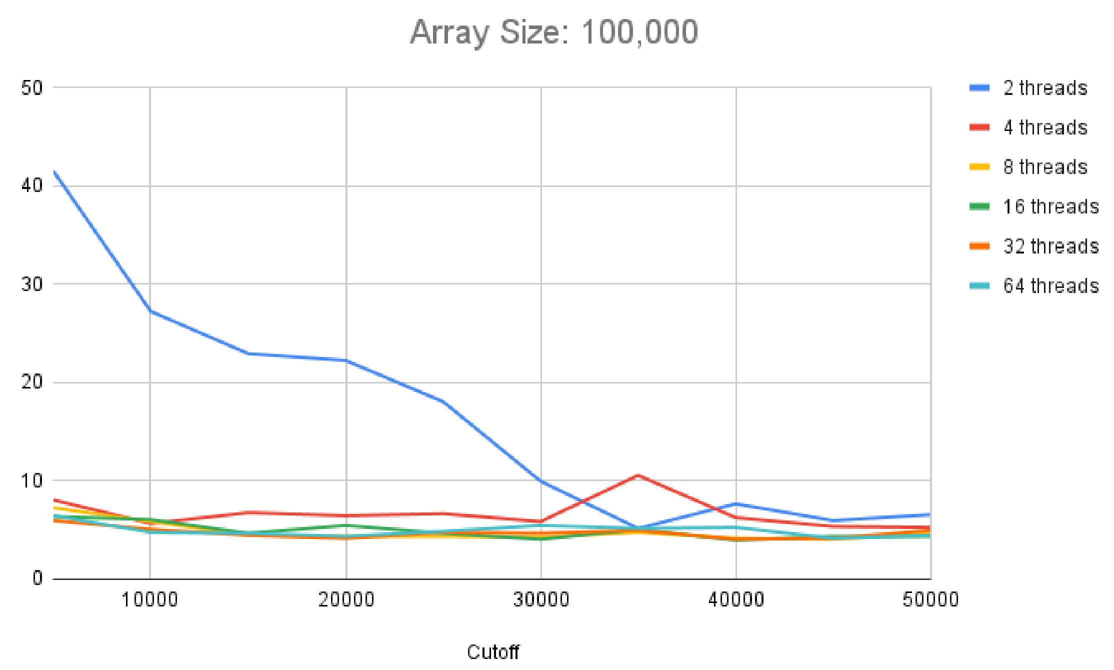
3. Observations

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

Array Size: 100,000 and Cutoff: 5000

Cutoff	2 threads	4 threads	8 threads	16 threads	32 threads	64 threads
5000	41.5	8	7.2	6.3	5.9	6.4
10000	27.2	5.6	5.8	6	5	4.7
15000	22.9	6.7	4.4	4.6	4.4	4.6
20000	22.2	6.4	4.3	5.4	4.1	4.3
25000	18	6.6	4.3	4.6	4.7	4.8
30000	9.9	5.8	4.2	4	4.6	5.4
35000	5.1	10.5	4.7	5	4.9	5.1
40000	7.6	6.2	4.1	3.9	4	5.2
45000	5.9	5.3	4	4.3	4.1	4.1
50000	6.5	5.2	4.6	4.3	4.9	4.4

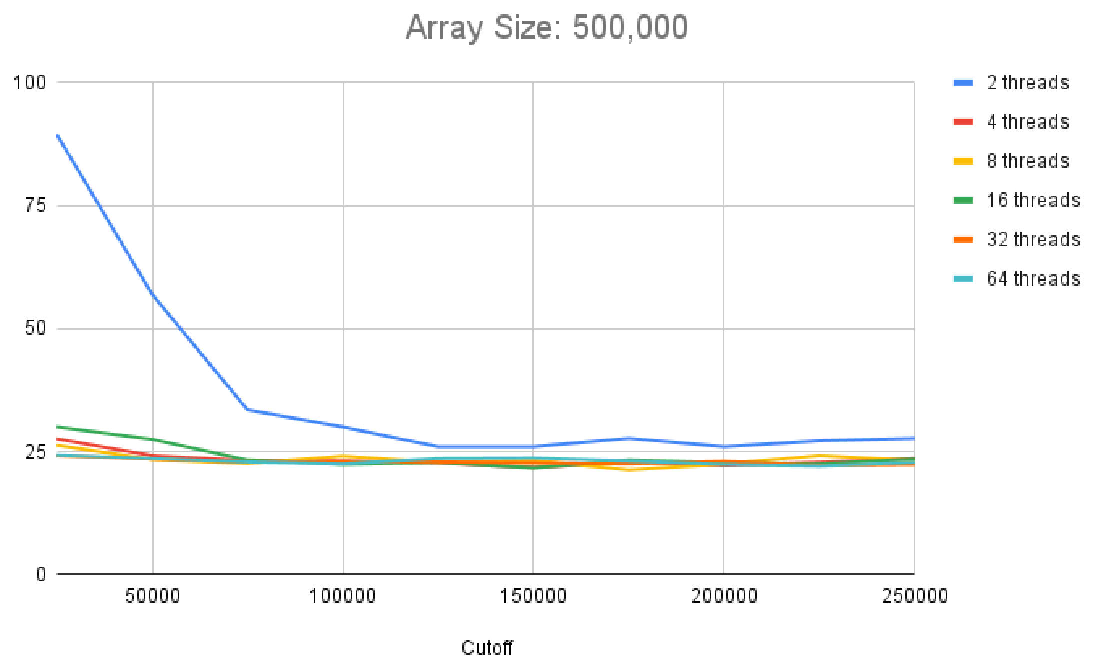
Here’s the graph for the same:



<u>Array Size: 500,000 and Cutoff: 25,000</u>

Cutoff	2 threads	4 threads	8 threads	16 threads	32 threads	64 threads
25000	89.5	27.6	26.3	30	24.2	24.3
50000	57	24.2	23.4	27.5	23.5	23.6
75000	33.5	23.2	22.6	23.3	22.9	22.9
100000	30	23.1	24.1	22.4	22.9	22.5
125000	26	22.7	22.8	22.8	22.9	23.6
150000	26	21.8	23.2	21.7	22.7	23.7
175000	27.7	22.9	21.3	23.3	22.5	23.1
200000	26	22.2	22.5	22.7	23	22.4
225000	27.2	22.8	24.2	22.5	22.2	22.1
250000	27.7	23.6	23.2	23.5	22.4	22.8

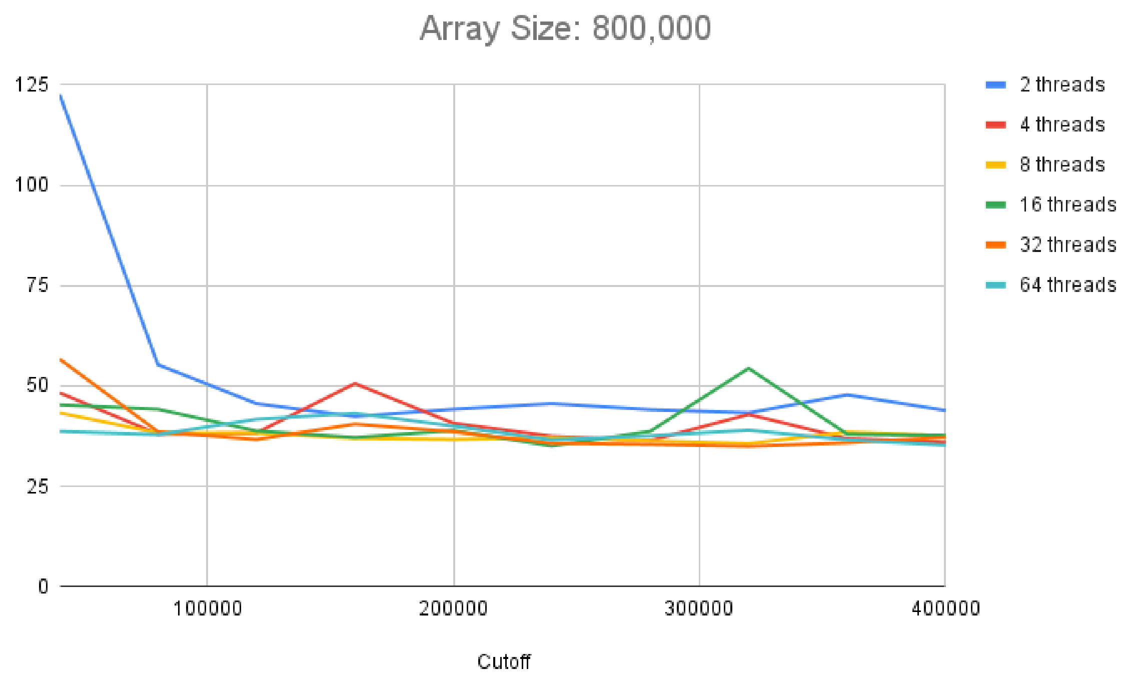
Below is the graph for the same:



Array Size: 800,000 and Cutoff: 40,000

Cutoff	**2 threads**	**4 threads**	**8 threads**	**16 threads**	**32 threads**	**64 threads**
40000	122.6	48.3	43.3	45.3	56.7	38.7
80000	55.3	38	38.3	44.2	38.6	37.9
120000	45.6	38.3	38.3	38.8	36.7	41.7
160000	42.4	50.6	37	37.2	40.5	43.2
200000	44.2	40.7	36.7	38.9	38.6	40
240000	45.6	37.5	37.2	35.1	35.8	36.6
280000	44.1	36.4	36.4	38.7	35.5	37.6
320000	43.3	42.9	35.7	54.4	35	39
360000	47.8	36.9	38.5	38	35.9	36.6
400000	43.9	36	37.6	37.7	37.2	35.3

Below is the graph for the same:



Comparing the above graphs and data from the tables, we can conclude the following:

- After changing the cutoff values and number of threads for different array sizes, the **number of threads bigger than 8** does not improve the performance of the algorithm. Hence **forking 8 threads** is the best option.
- Referring to the graphs above, we can see 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.

The data and observations can be found **here on Google Sheets**.

4. Code

The code for this assignment is available on **my GitHub repository**. The excel sheet containing the graph and other observations can be found **here**.