Program Structures & Algorithms

Assignment 4 (Parallel Sorting)

Arjun Raja Yogidas
NUID: 002964082

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.

Relationship Conclusions:

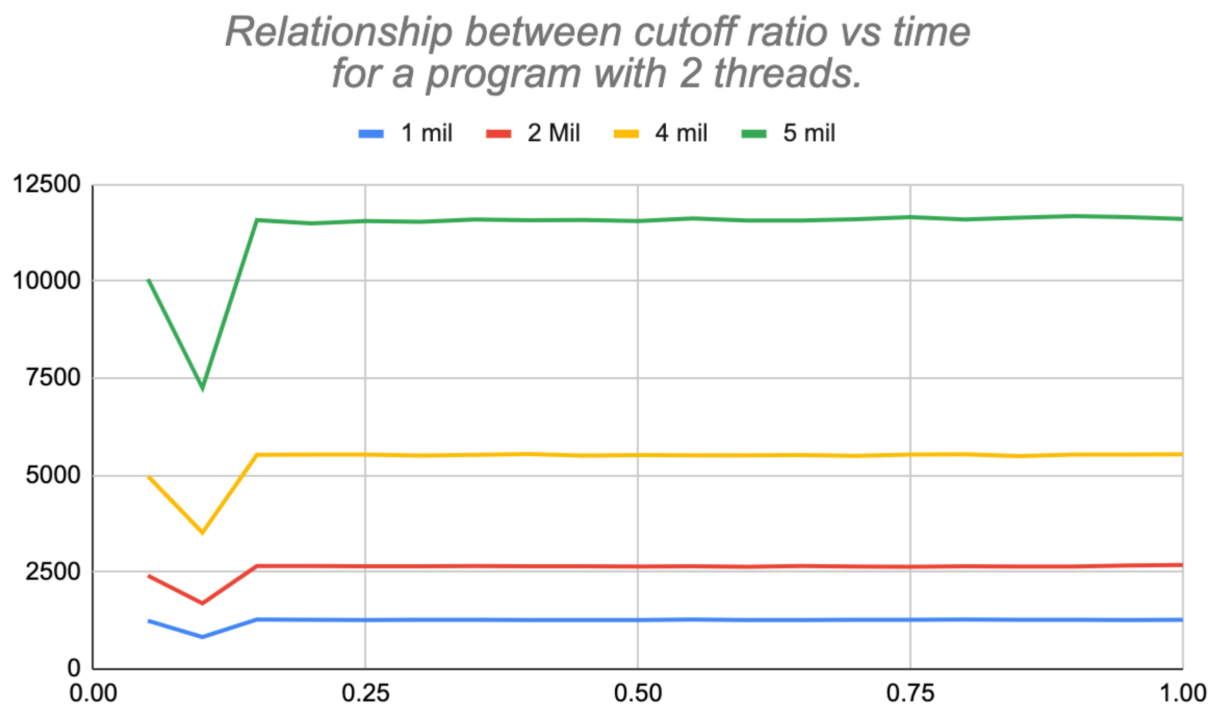
1. As per the cutoff ratio (cutoff/size of the array), we see that irrespective of the array sizes,and the number of threads available for parallel sorting, there is a range of cutoff ratio between 0.1 and 0.55 for which parallel sort can be an optimal solution
2. As per performing analysis on different sizes of arrays (2 Million and 16 Million), for a number of different threads ranging from 2 to 16, we see that 6 threads works as a best case scenario

Evidence to support the conclusion:
1. Finding the optimal cutoff value, for different sizes of input array(No of threads constant)

Threads 2, Array sizes varied from : 1 Million to 6 Million, doubling each run. Cutoff ratio: (Cutoff value/Size of the array): Ranging from 0.05 to to 1.0 in increments of 0.05

Cutoff ratio	1 Million	2 Million	4 Million	8 Million	16 Million
0.05	676	1245	2412	4971	10058
0.1	397	825	1694	3520	7254
0.15	617	1279	2657	5526	11580
0.2	607	1269	2656	5533	11499
0.25	603	1264	2653	5531	11557
0.3	605	1271	2650	5507	11539
0.35	596	1270	2658	5528	11595
0.4	602	1266	2650	5549	11575
0.45	606	1267	2651	5504	11583
0.5	599	1267	2645	5516	11560
0.55	601	1275	2654	5510	11625
0.6	599	1262	2637	5512	11570
0.65	597	1262	2658	5522	11568
0.7	600	1269	2647	5501	11604
0.75	602	1270	2635	5534	11656
0.8	600	1275	2649	5537	11600
0.85	600	1269	2644	5490	11646
0.9	599	1272	2643	5531	11687

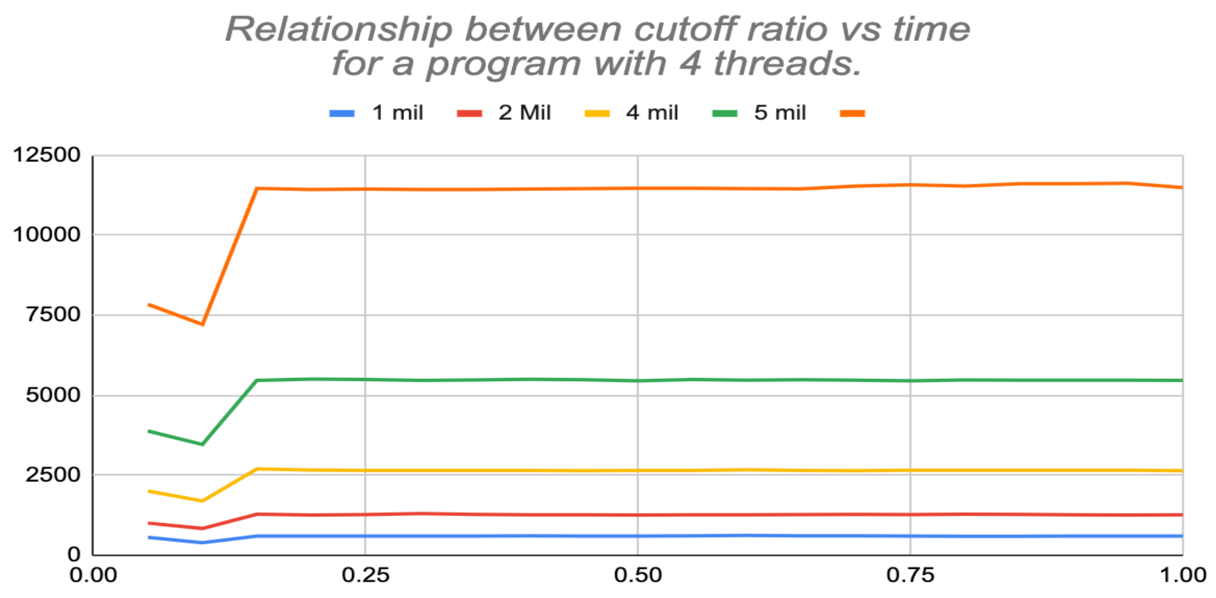
0.95	601	1266	2672	5530	11659
1.00	598	1272	2685	5540	11609



Threads: 4
Array sizes varied from : 1 Million to 6 Million, doubling each run. Cutoff ratio: (Cutoff value/Size of the array): Ranging from 0.05 to to 1.0 in increments of 0.05

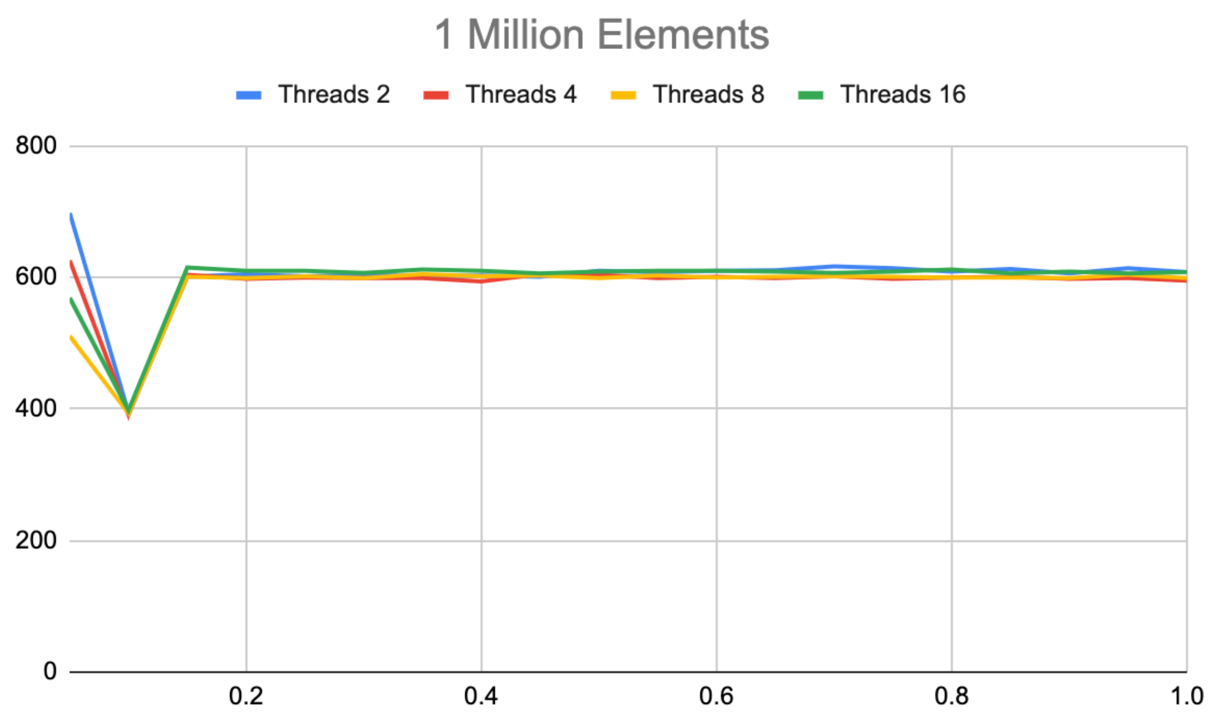
Cutoff ratio	1 Million	2 Million	4 Million	8 Million	16 Million
0.05	563	1006	2009	3883	7841
0.1	392	839	1696	3462	7211
0.15	603	1281	2695	5467	11462

0.2	600	1256	2664	5509	11431
0.25	600	1269	2654	5493	11440
0.3	601	1304	2653	5464	11431
0.35	600	1279	2648	5480	11429
0.4	604	1265	2648	5496	11445
0.45	600	1263	2642	5484	11458
0.5	600	1254	2649	5450	11470
0.55	606	1261	2653	5493	11472
0.6	618	1261	2670	5472	11454
0.65	608	1268	2650	5487	11447
0.7	607	1279	2641	5473	11539
0.75	601	1271	2658	5453	11575
0.8	597	1284	2658	5477	11538
0.85	597	1278	2656	5475	11611
0.9	599	1262	2655	5472	11613
0.95	601	1260	2656	5470	11621
1.00	599	1264	2644	5465	11491



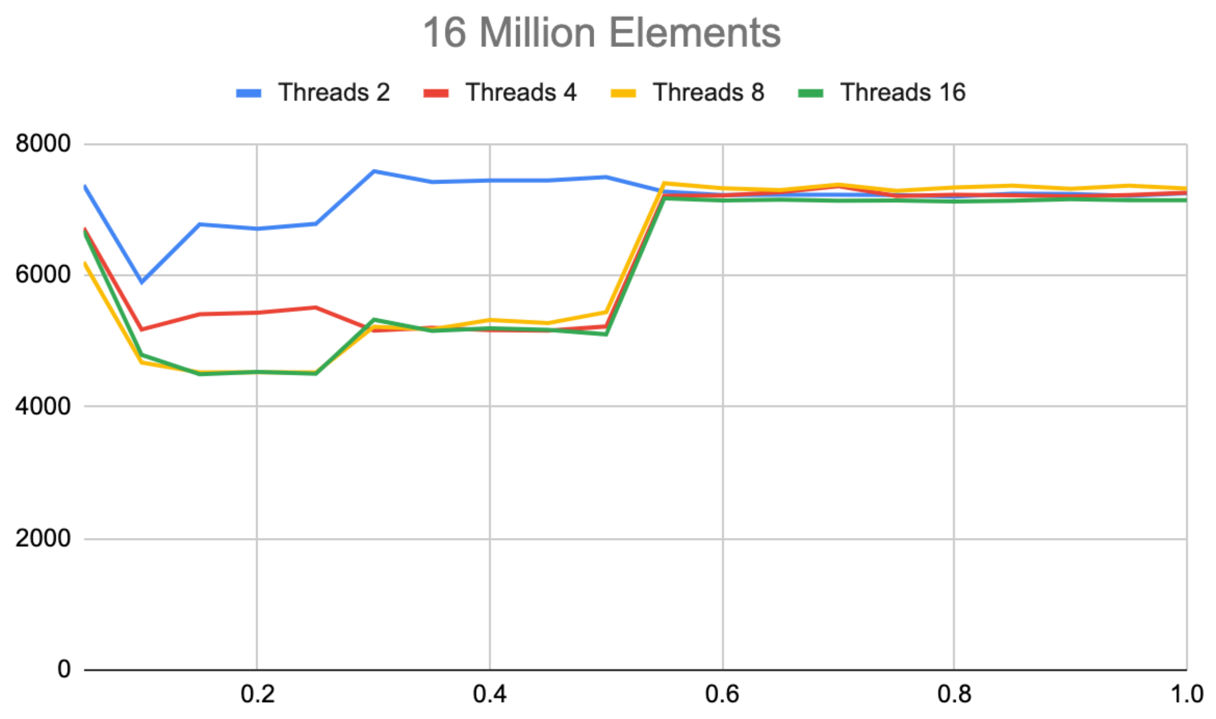
2. Checking for different threads from 2 to 16 for an array of 1000000 elements, with cutoff ratios changing from 0.05 to 1.0

Cutoff ratio	Threads 2	Threads 4	Threads 8	Threads 16
0.05	698	626	511	569
0.1	395	391	393	398
0.15	601	604	601	615
0.2	605	598	599	610
0.25	602	600	602	610
0.3	604	599	599	607
0.35	602	599	605	612
0.4	603	594	602	610
0.45	601	605	603	606
0.5	610	604	599	609
0.55	608	599	603	610
0.6	610	601	600	610
0.65	611	599	601	609
0.7	617	602	602	607
0.75	614	598	601	609
0.8	609	600	600	612
0.85	613	601	600	606
0.9	606	598	599	609
0.95	614	599	604	606
1	608	595	599	608

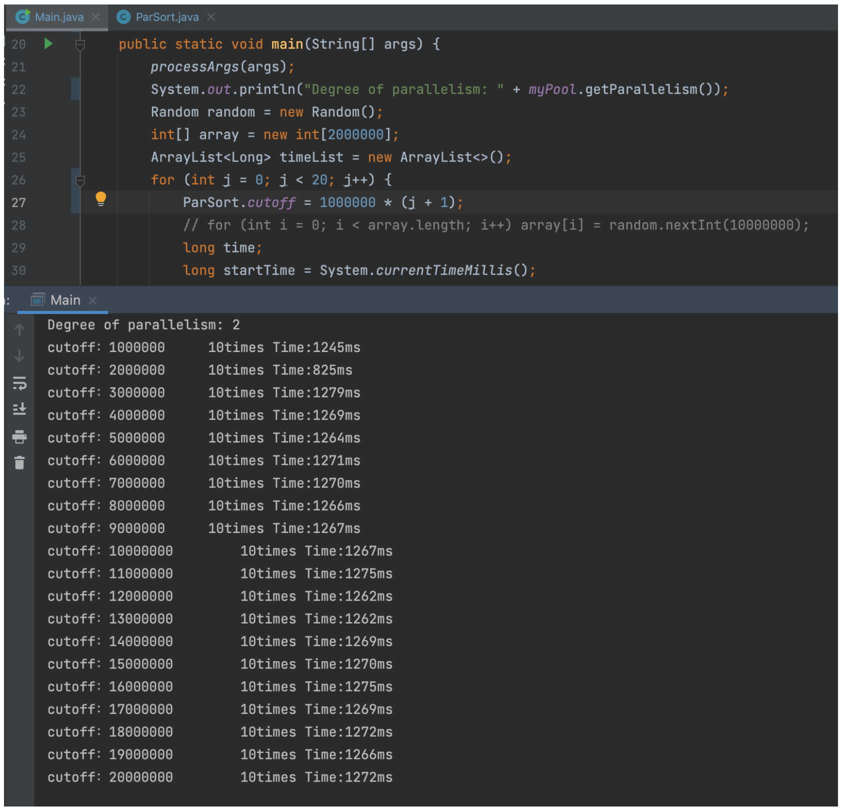


Cutoff ratio	Threads 2	Threads 4	Threads 8	Threads 16
0.05	7373	6720	6202	6679
0.1	5901	5177	4676	4792
0.15	6776	5410	4524	4498
0.2	6710	5433	4528	4533
0.25	6784	5512	4525	4507
0.3	7585	5162	5220	5327
0.35	7422	5205	5180	5159
0.4	7445	5168	5324	5196
0.45	7443	5162	5274	5174
0.5	7494	5223	5441	5105
0.55	7273	7211	7402	7171
0.6	7222	7214	7327	7139

0.65	7225	7268	7297	7150
0.7	7225	7356	7380	7135
0.75	7224	7208	7285	7138
0.8	7198	7226	7339	7123
0.85	7242	7219	7366	7137
0.9	7239	7203	7319	7161
0.95	7211	7222	7366	7145
1	7250	7260	7320	7144

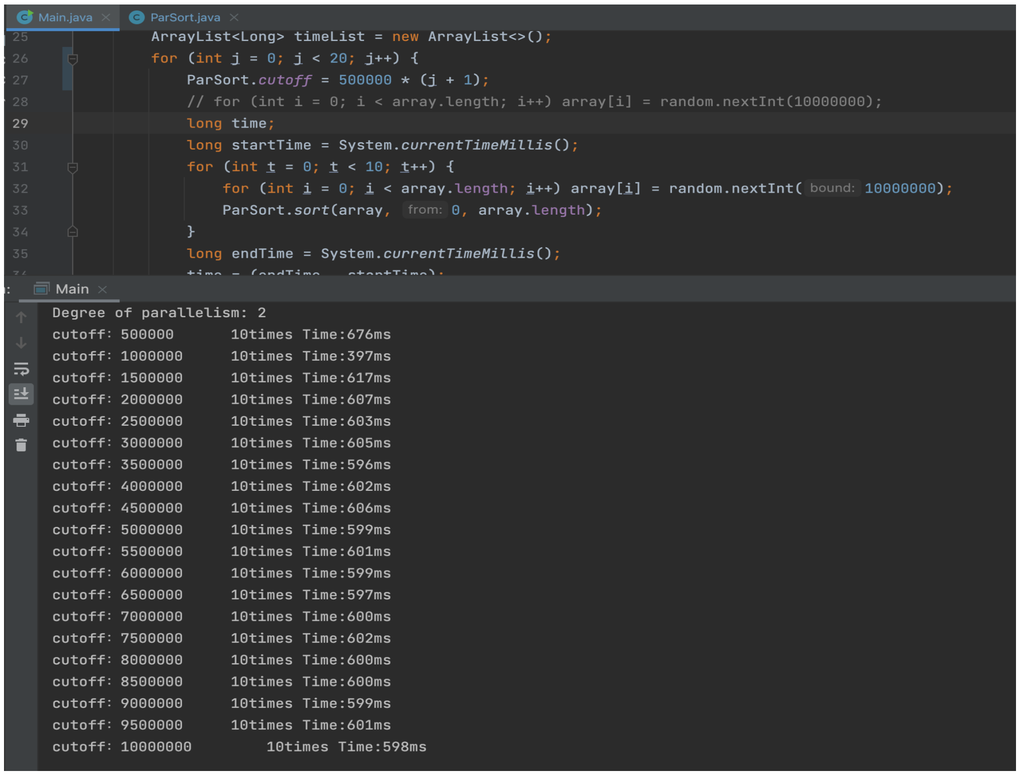


Screenshots proof for conclusion 1



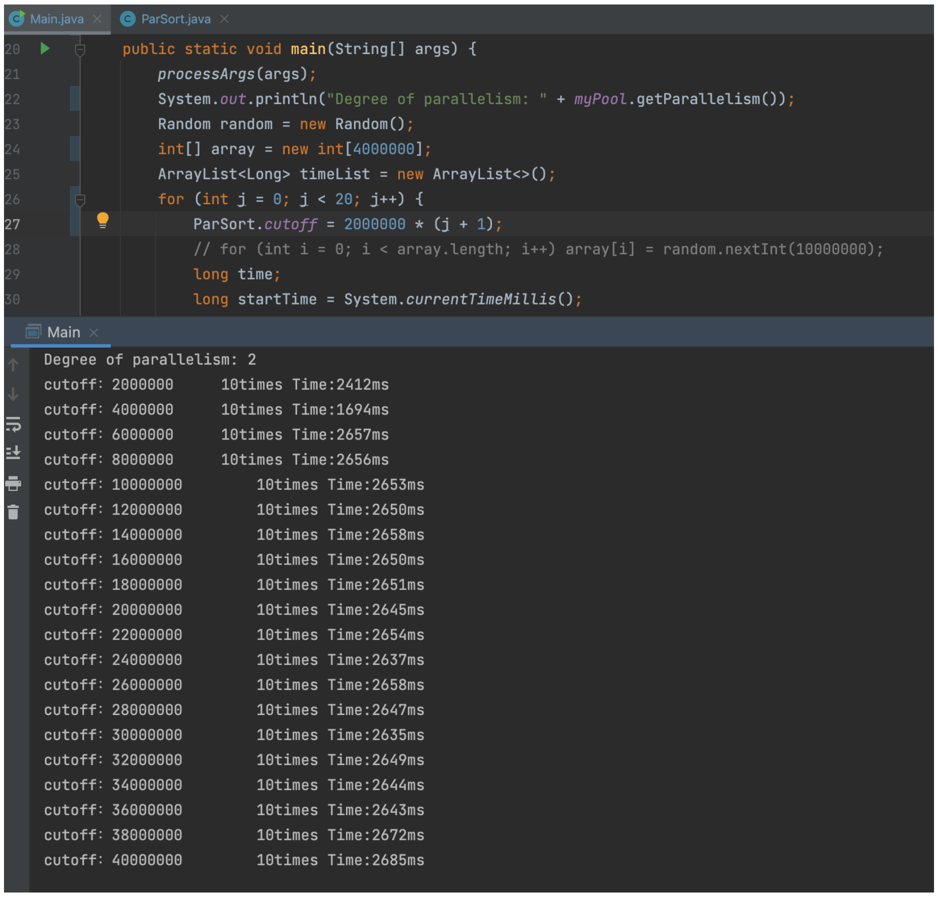
```
public static void main(String[] args) {
    processArgs(args);
    System.out.println("Degree of parallelism: " + myPool.getParallelism());
    Random random = new Random();
    int[] array = new int[2000000];
    ArrayList<Long> timeList = new ArrayList<>();
    for (int j = 0; j < 20; j++) {
        ParSort.cutoff = 1000000 * (j + 1);
        // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
        long time;
        long startTime = System.currentTimeMillis();
```

```
Degree of parallelism: 2
cutoff: 1000000     10times Time:1245ms
cutoff: 2000000     10times Time:825ms
cutoff: 3000000     10times Time:1279ms
cutoff: 4000000     10times Time:1269ms
cutoff: 5000000     10times Time:1264ms
cutoff: 6000000     10times Time:1271ms
cutoff: 7000000     10times Time:1270ms
cutoff: 8000000     10times Time:1266ms
cutoff: 9000000     10times Time:1267ms
cutoff: 10000000        10times Time:1267ms
cutoff: 11000000        10times Time:1275ms
cutoff: 12000000        10times Time:1262ms
cutoff: 13000000        10times Time:1262ms
cutoff: 14000000        10times Time:1269ms
cutoff: 15000000        10times Time:1270ms
cutoff: 16000000        10times Time:1275ms
cutoff: 17000000        10times Time:1269ms
cutoff: 18000000        10times Time:1272ms
cutoff: 19000000        10times Time:1266ms
cutoff: 20000000        10times Time:1272ms
```



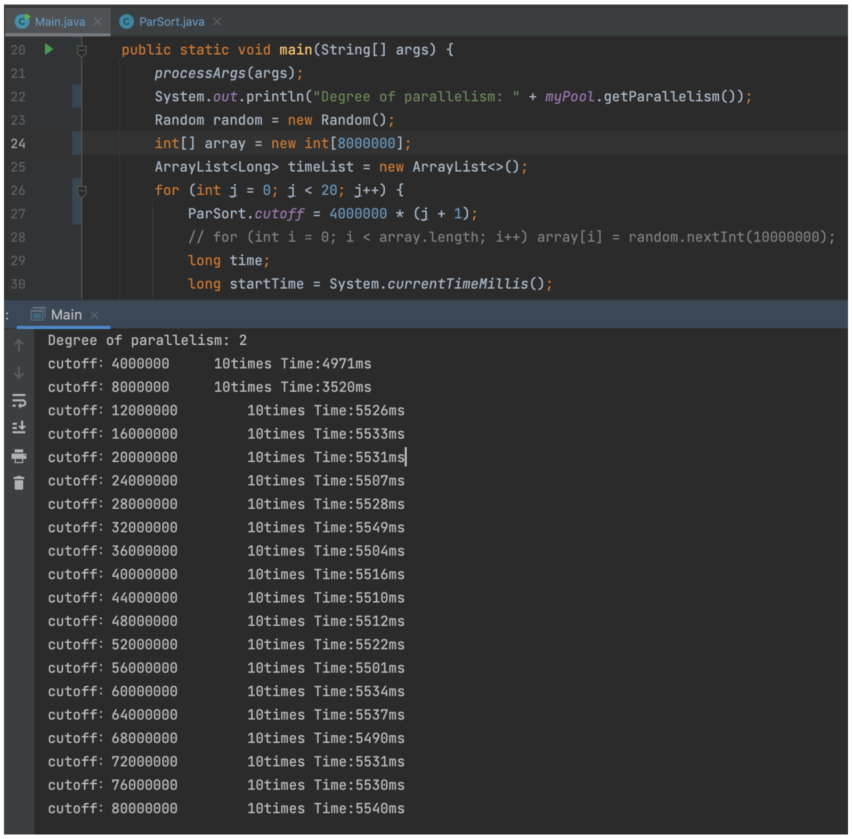
```
    ArrayList<Long> timeList = new ArrayList<>();
    for (int j = 0; j < 20; j++) {
        ParSort.cutoff = 500000 * (j + 1);
        // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
        long time;
        long startTime = System.currentTimeMillis();
        for (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();
```

```
Degree of parallelism: 2
cutoff: 500000      10times Time:676ms
cutoff: 1000000     10times Time:397ms
cutoff: 1500000     10times Time:617ms
cutoff: 2000000     10times Time:607ms
cutoff: 2500000     10times Time:603ms
cutoff: 3000000     10times Time:605ms
cutoff: 3500000     10times Time:596ms
cutoff: 4000000     10times Time:602ms
cutoff: 4500000     10times Time:606ms
cutoff: 5000000     10times Time:599ms
cutoff: 5500000     10times Time:601ms
cutoff: 6000000     10times Time:599ms
cutoff: 6500000     10times Time:597ms
cutoff: 7000000     10times Time:600ms
cutoff: 7500000     10times Time:602ms
cutoff: 8000000     10times Time:600ms
cutoff: 8500000     10times Time:600ms
cutoff: 9000000     10times Time:599ms
cutoff: 9500000     10times Time:601ms
cutoff: 10000000        10times Time:598ms
```



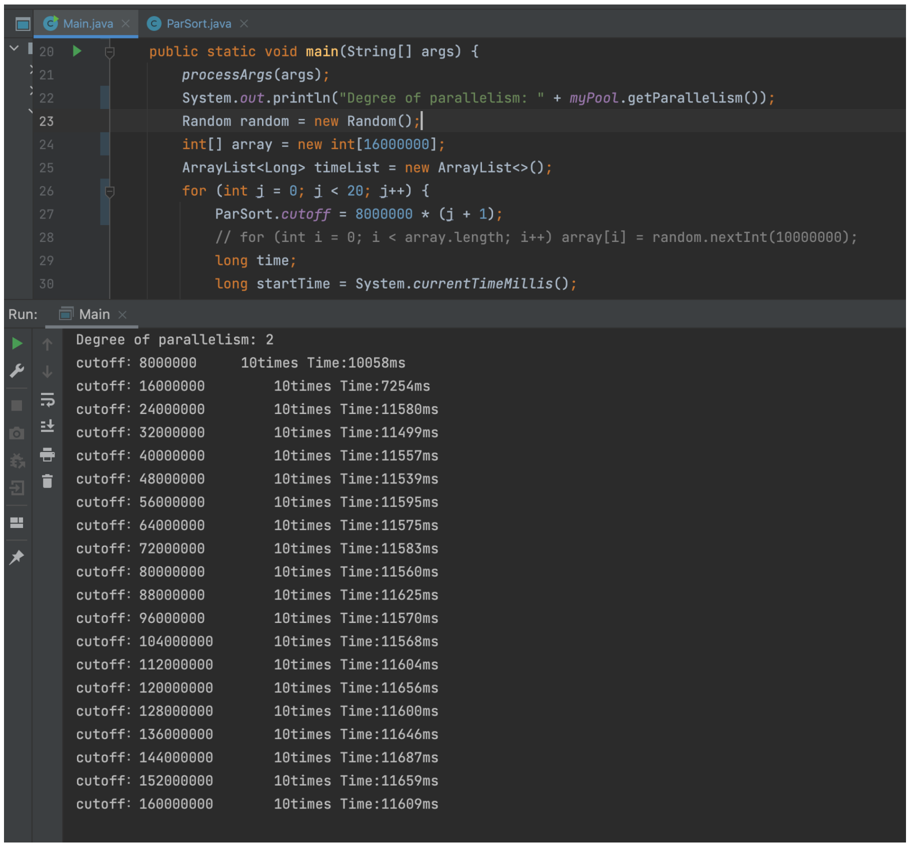
```java
public static void main(String[] args) {
    processArgs(args);
    System.out.println("Degree of parallelism: " + myPool.getParallelism());
    Random random = new Random();
    int[] array = new int[4000000];
    ArrayList<Long> timeList = new ArrayList<>();
    for (int j = 0; j < 20; j++) {
        ParSort.cutoff = 2000000 * (j + 1);
        // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
        long time;
        long startTime = System.currentTimeMillis();
```

```
Degree of parallelism: 2
cutoff: 2000000     10times Time:2412ms
cutoff: 4000000     10times Time:1694ms
cutoff: 6000000     10times Time:2657ms
cutoff: 8000000     10times Time:2656ms
cutoff: 10000000        10times Time:2653ms
cutoff: 12000000        10times Time:2650ms
cutoff: 14000000        10times Time:2658ms
cutoff: 16000000        10times Time:2650ms
cutoff: 18000000        10times Time:2651ms
cutoff: 20000000        10times Time:2645ms
cutoff: 22000000        10times Time:2654ms
cutoff: 24000000        10times Time:2637ms
cutoff: 26000000        10times Time:2658ms
cutoff: 28000000        10times Time:2647ms
cutoff: 30000000        10times Time:2635ms
cutoff: 32000000        10times Time:2649ms
cutoff: 34000000        10times Time:2644ms
cutoff: 36000000        10times Time:2643ms
cutoff: 38000000        10times Time:2672ms
cutoff: 40000000        10times Time:2685ms
```



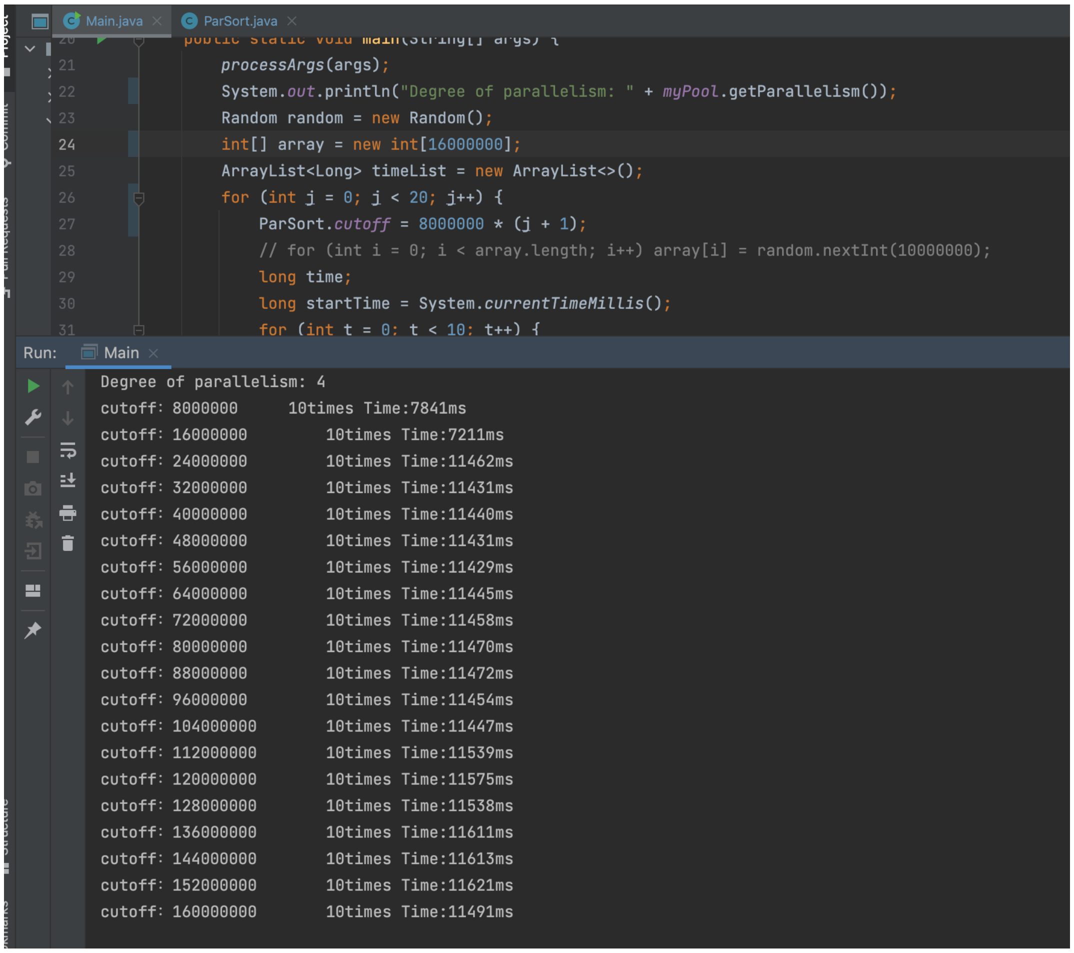
```java
public static void main(String[] args) {
    processArgs(args);
    System.out.println("Degree of parallelism: " + myPool.getParallelism());
    Random random = new Random();
    int[] array = new int[8000000];
    ArrayList<Long> timeList = new ArrayList<>();
    for (int j = 0; j < 20; j++) {
        ParSort.cutoff = 4000000 * (j + 1);
        // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
        long time;
        long startTime = System.currentTimeMillis();
```

```
Degree of parallelism: 2
cutoff: 4000000     10times Time:4971ms
cutoff: 8000000     10times Time:3520ms
cutoff: 12000000        10times Time:5526ms
cutoff: 16000000        10times Time:5533ms
cutoff: 20000000        10times Time:5531ms
cutoff: 24000000        10times Time:5507ms
cutoff: 28000000        10times Time:5528ms
cutoff: 32000000        10times Time:5549ms
cutoff: 36000000        10times Time:5504ms
cutoff: 40000000        10times Time:5516ms
cutoff: 44000000        10times Time:5510ms
cutoff: 48000000        10times Time:5512ms
cutoff: 52000000        10times Time:5522ms
cutoff: 56000000        10times Time:5501ms
cutoff: 60000000        10times Time:5534ms
cutoff: 64000000        10times Time:5537ms
cutoff: 68000000        10times Time:5490ms
cutoff: 72000000        10times Time:5531ms
cutoff: 76000000        10times Time:5530ms
cutoff: 80000000        10times Time:5540ms
```



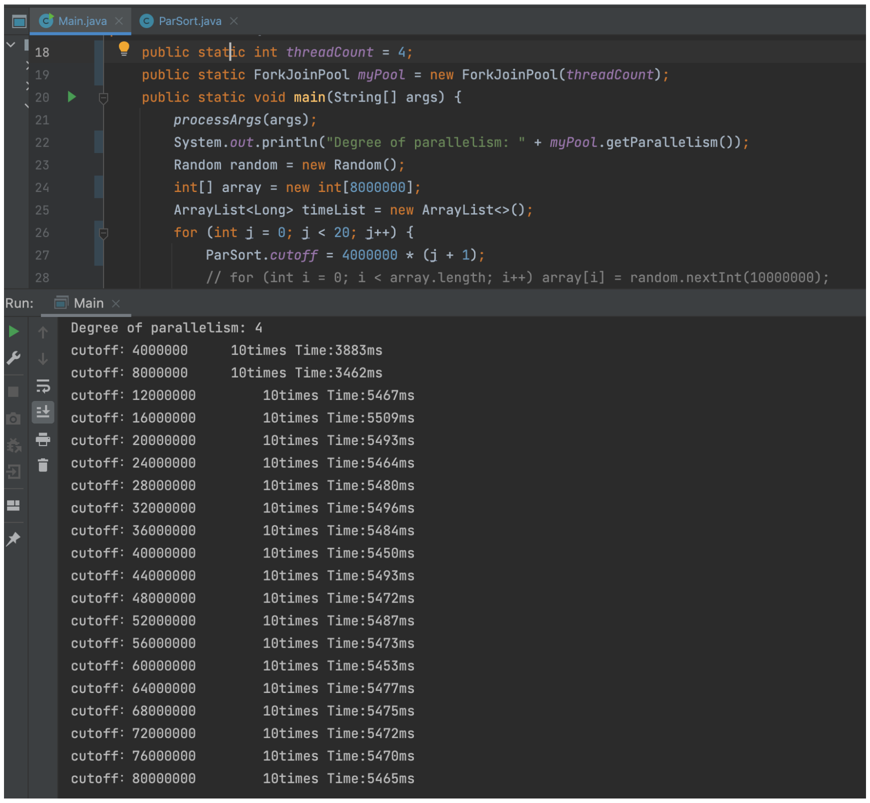
```java
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[16000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 8000000 * (j + 1);
            // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
            long time;
            long startTime = System.currentTimeMillis();
```

```
Degree of parallelism: 2
cutoff: 8000000     10times Time:10058ms
cutoff: 16000000        10times Time:7254ms
cutoff: 24000000        10times Time:11580ms
cutoff: 32000000        10times Time:11499ms
cutoff: 40000000        10times Time:11557ms
cutoff: 48000000        10times Time:11539ms
cutoff: 56000000        10times Time:11595ms
cutoff: 64000000        10times Time:11575ms
cutoff: 72000000        10times Time:11583ms
cutoff: 80000000        10times Time:11560ms
cutoff: 88000000        10times Time:11625ms
cutoff: 96000000        10times Time:11570ms
cutoff: 104000000       10times Time:11568ms
cutoff: 112000000       10times Time:11604ms
cutoff: 120000000       10times Time:11656ms
cutoff: 128000000       10times Time:11600ms
cutoff: 136000000       10times Time:11646ms
cutoff: 144000000       10times Time:11687ms
cutoff: 152000000       10times Time:11659ms
cutoff: 160000000       10times Time:11609ms
```



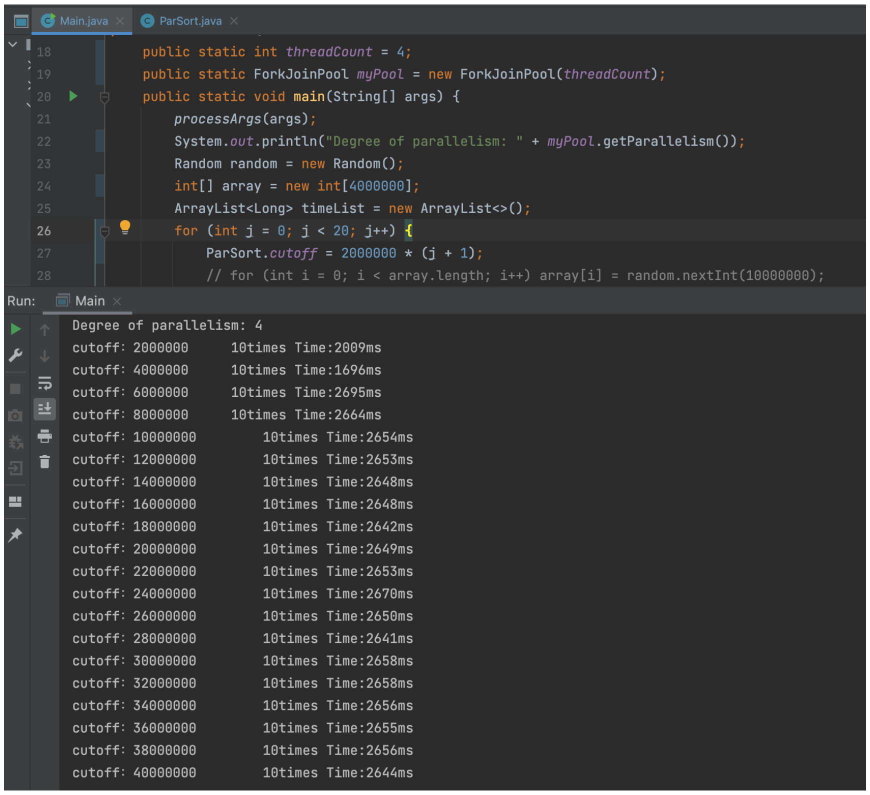
```java
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[16000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 8000000 * (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: 4
cutoff: 8000000     10times Time:7841ms
cutoff: 16000000        10times Time:7211ms
cutoff: 24000000        10times Time:11462ms
cutoff: 32000000        10times Time:11431ms
cutoff: 40000000        10times Time:11440ms
cutoff: 48000000        10times Time:11431ms
cutoff: 56000000        10times Time:11429ms
cutoff: 64000000        10times Time:11445ms
cutoff: 72000000        10times Time:11458ms
cutoff: 80000000        10times Time:11470ms
cutoff: 88000000        10times Time:11472ms
cutoff: 96000000        10times Time:11454ms
cutoff: 104000000       10times Time:11447ms
cutoff: 112000000       10times Time:11539ms
cutoff: 120000000       10times Time:11575ms
cutoff: 128000000       10times Time:11538ms
cutoff: 136000000       10times Time:11611ms
cutoff: 144000000       10times Time:11613ms
cutoff: 152000000       10times Time:11621ms
cutoff: 160000000       10times Time:11491ms
```



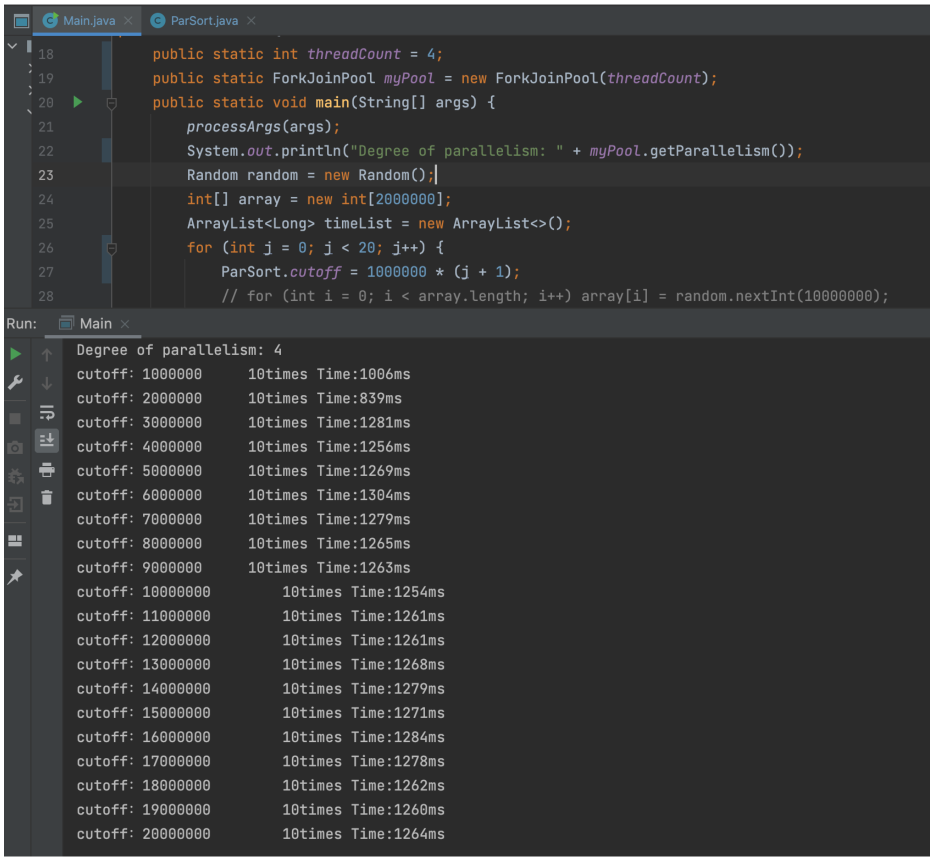
```
public static int threadCount = 4;
public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
public static void main(String[] args) {
    processArgs(args);
    System.out.println("Degree of parallelism: " + myPool.getParallelism());
    Random random = new Random();
    int[] array = new int[8000000];
    ArrayList<Long> timeList = new ArrayList<>();
    for (int j = 0; j < 20; j++) {
        ParSort.cutoff = 4000000 * (j + 1);
        // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
```

```
Degree of parallelism: 4
cutoff: 4000000     10times Time:3883ms
cutoff: 8000000     10times Time:3462ms
cutoff: 12000000        10times Time:5467ms
cutoff: 16000000        10times Time:5509ms
cutoff: 20000000        10times Time:5493ms
cutoff: 24000000        10times Time:5464ms
cutoff: 28000000        10times Time:5480ms
cutoff: 32000000        10times Time:5496ms
cutoff: 36000000        10times Time:5484ms
cutoff: 40000000        10times Time:5450ms
cutoff: 44000000        10times Time:5493ms
cutoff: 48000000        10times Time:5472ms
cutoff: 52000000        10times Time:5487ms
cutoff: 56000000        10times Time:5473ms
cutoff: 60000000        10times Time:5453ms
cutoff: 64000000        10times Time:5477ms
cutoff: 68000000        10times Time:5475ms
cutoff: 72000000        10times Time:5472ms
cutoff: 76000000        10times Time:5470ms
cutoff: 80000000        10times Time:5465ms
```



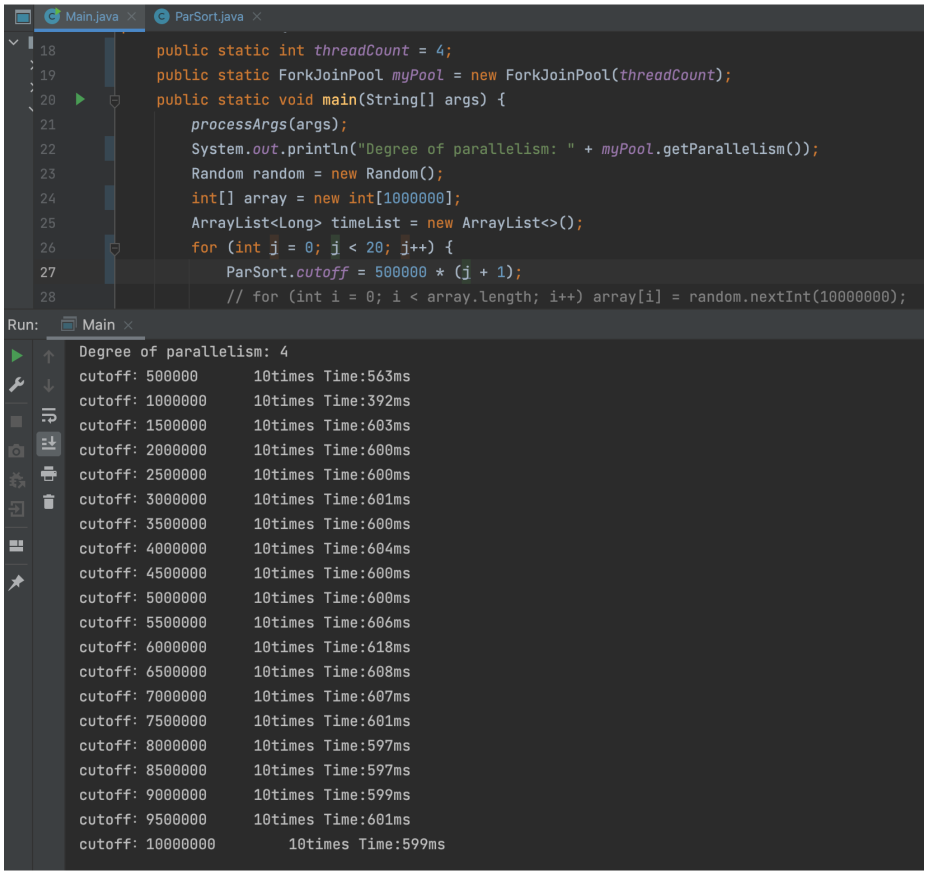
```
public static int threadCount = 4;
public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
public static void main(String[] args) {
    processArgs(args);
    System.out.println("Degree of parallelism: " + myPool.getParallelism());
    Random random = new Random();
    int[] array = new int[4000000];
    ArrayList<Long> timeList = new ArrayList<>();
    for (int j = 0; j < 20; j++) {
        ParSort.cutoff = 2000000 * (j + 1);
        // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
```

```
Degree of parallelism: 4
cutoff: 2000000     10times Time:2009ms
cutoff: 4000000     10times Time:1696ms
cutoff: 6000000     10times Time:2695ms
cutoff: 8000000     10times Time:2664ms
cutoff: 10000000        10times Time:2654ms
cutoff: 12000000        10times Time:2653ms
cutoff: 14000000        10times Time:2648ms
cutoff: 16000000        10times Time:2648ms
cutoff: 18000000        10times Time:2642ms
cutoff: 20000000        10times Time:2649ms
cutoff: 22000000        10times Time:2653ms
cutoff: 24000000        10times Time:2670ms
cutoff: 26000000        10times Time:2650ms
cutoff: 28000000        10times Time:2641ms
cutoff: 30000000        10times Time:2658ms
cutoff: 32000000        10times Time:2658ms
cutoff: 34000000        10times Time:2656ms
cutoff: 36000000        10times Time:2655ms
cutoff: 38000000        10times Time:2656ms
cutoff: 40000000        10times Time:2644ms
```



```java
    public static int threadCount = 4;
    public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[2000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 1000000 * (j + 1);
            // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
```

```
Degree of parallelism: 4
cutoff: 1000000     10times Time:1006ms
cutoff: 2000000     10times Time:839ms
cutoff: 3000000     10times Time:1281ms
cutoff: 4000000     10times Time:1256ms
cutoff: 5000000     10times Time:1269ms
cutoff: 6000000     10times Time:1304ms
cutoff: 7000000     10times Time:1279ms
cutoff: 8000000     10times Time:1265ms
cutoff: 9000000     10times Time:1263ms
cutoff: 10000000        10times Time:1254ms
cutoff: 11000000        10times Time:1261ms
cutoff: 12000000        10times Time:1261ms
cutoff: 13000000        10times Time:1268ms
cutoff: 14000000        10times Time:1279ms
cutoff: 15000000        10times Time:1271ms
cutoff: 16000000        10times Time:1284ms
cutoff: 17000000        10times Time:1278ms
cutoff: 18000000        10times Time:1262ms
cutoff: 19000000        10times Time:1260ms
cutoff: 20000000        10times Time:1264ms
```

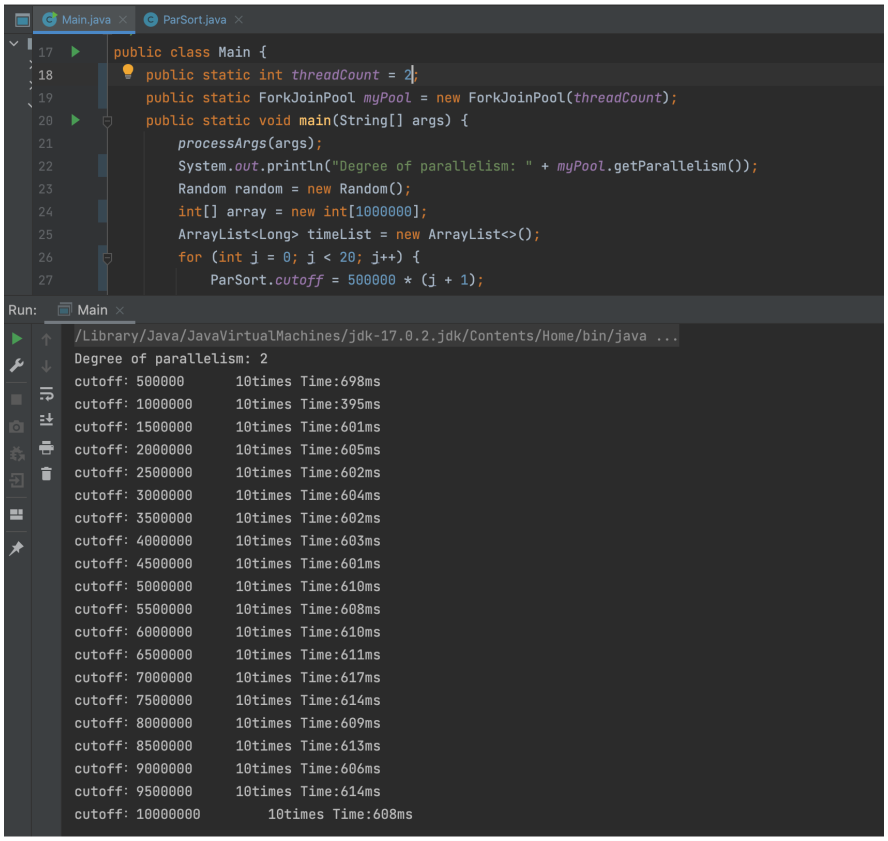


```java
    public static int threadCount = 4;
    public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[1000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 500000 * (j + 1);
            // for (int i = 0; i < array.length; i++) array[i] = random.nextInt(10000000);
```

```
Degree of parallelism: 4
cutoff: 500000      10times Time:563ms
cutoff: 1000000     10times Time:392ms
cutoff: 1500000     10times Time:603ms
cutoff: 2000000     10times Time:600ms
cutoff: 2500000     10times Time:600ms
cutoff: 3000000     10times Time:601ms
cutoff: 3500000     10times Time:600ms
cutoff: 4000000     10times Time:604ms
cutoff: 4500000     10times Time:600ms
cutoff: 5000000     10times Time:600ms
cutoff: 5500000     10times Time:606ms
cutoff: 6000000     10times Time:618ms
cutoff: 6500000     10times Time:608ms
cutoff: 7000000     10times Time:607ms
cutoff: 7500000     10times Time:601ms
cutoff: 8000000     10times Time:597ms
cutoff: 8500000     10times Time:597ms
cutoff: 9000000     10times Time:599ms
cutoff: 9500000     10times Time:601ms
cutoff: 10000000        10times Time:599ms
```

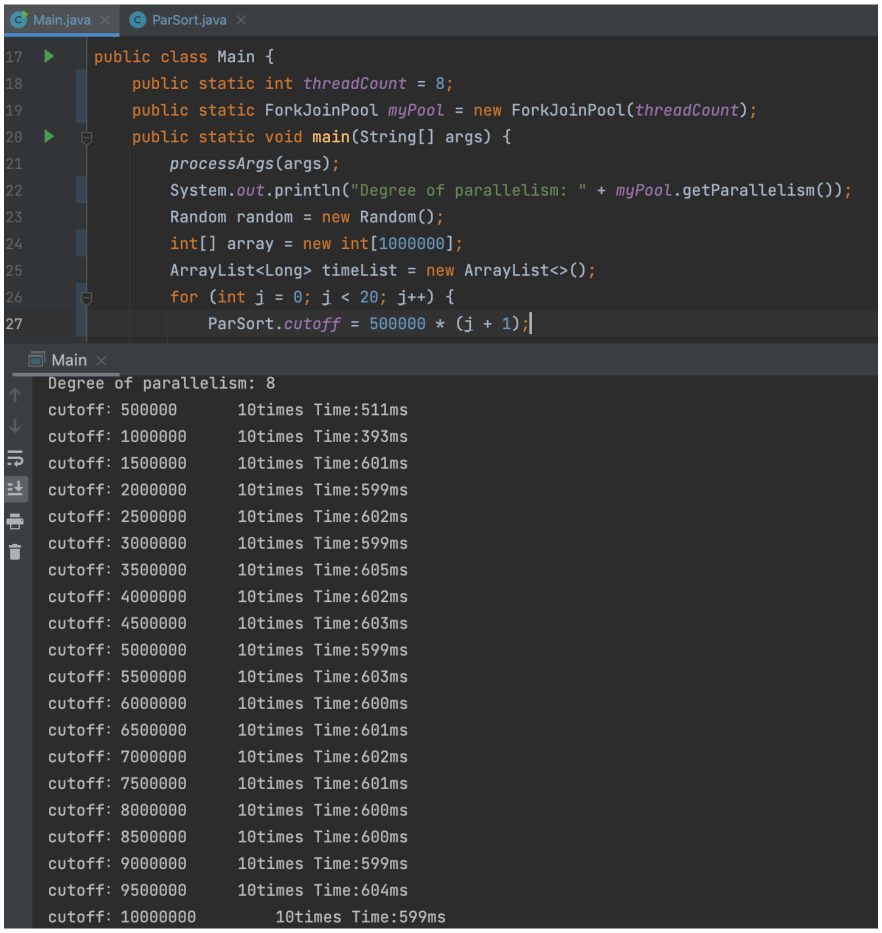
Sample Output for Conclusion 2:

For array size: 1 Million, varying threads from 2 to 16



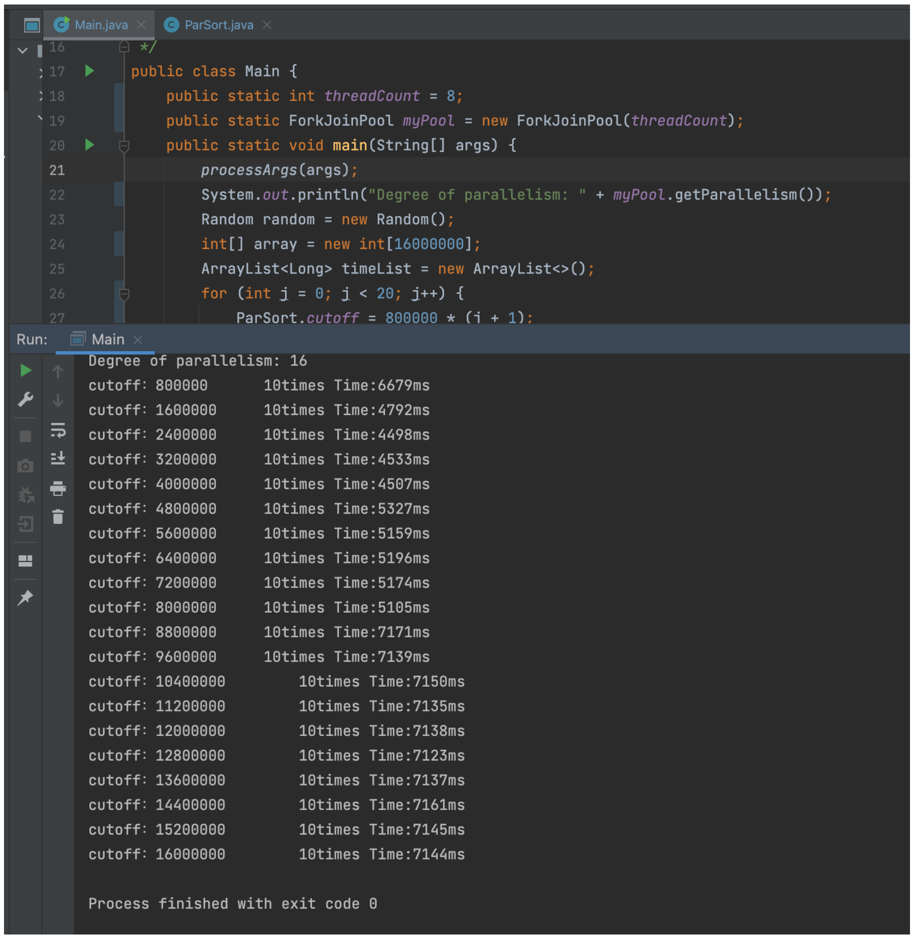
```java
public class Main {
    public static int threadCount = 2;
    public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[1000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 500000 * (j + 1);
```

```
/Library/Java/JavaVirtualMachines/jdk-17.0.2.jdk/Contents/Home/bin/java ...
Degree of parallelism: 2
cutoff: 500000      10times Time:698ms
cutoff: 1000000     10times Time:395ms
cutoff: 1500000     10times Time:601ms
cutoff: 2000000     10times Time:605ms
cutoff: 2500000     10times Time:602ms
cutoff: 3000000     10times Time:604ms
cutoff: 3500000     10times Time:602ms
cutoff: 4000000     10times Time:603ms
cutoff: 4500000     10times Time:601ms
cutoff: 5000000     10times Time:610ms
cutoff: 5500000     10times Time:608ms
cutoff: 6000000     10times Time:610ms
cutoff: 6500000     10times Time:611ms
cutoff: 7000000     10times Time:617ms
cutoff: 7500000     10times Time:614ms
cutoff: 8000000     10times Time:609ms
cutoff: 8500000     10times Time:613ms
cutoff: 9000000     10times Time:606ms
cutoff: 9500000     10times Time:614ms
cutoff: 10000000        10times Time:608ms
```



```java
public class Main {
    public static int threadCount = 8;
    public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[1000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 500000 * (j + 1);
```

```
Degree of parallelism: 8
cutoff: 500000      10times Time:511ms
cutoff: 1000000     10times Time:393ms
cutoff: 1500000     10times Time:601ms
cutoff: 2000000     10times Time:599ms
cutoff: 2500000     10times Time:602ms
cutoff: 3000000     10times Time:599ms
cutoff: 3500000     10times Time:605ms
cutoff: 4000000     10times Time:602ms
cutoff: 4500000     10times Time:603ms
cutoff: 5000000     10times Time:599ms
cutoff: 5500000     10times Time:603ms
cutoff: 6000000     10times Time:600ms
cutoff: 6500000     10times Time:601ms
cutoff: 7000000     10times Time:602ms
cutoff: 7500000     10times Time:601ms
cutoff: 8000000     10times Time:600ms
cutoff: 8500000     10times Time:600ms
cutoff: 9000000     10times Time:599ms
cutoff: 9500000     10times Time:604ms
cutoff: 10000000        10times Time:599ms
```



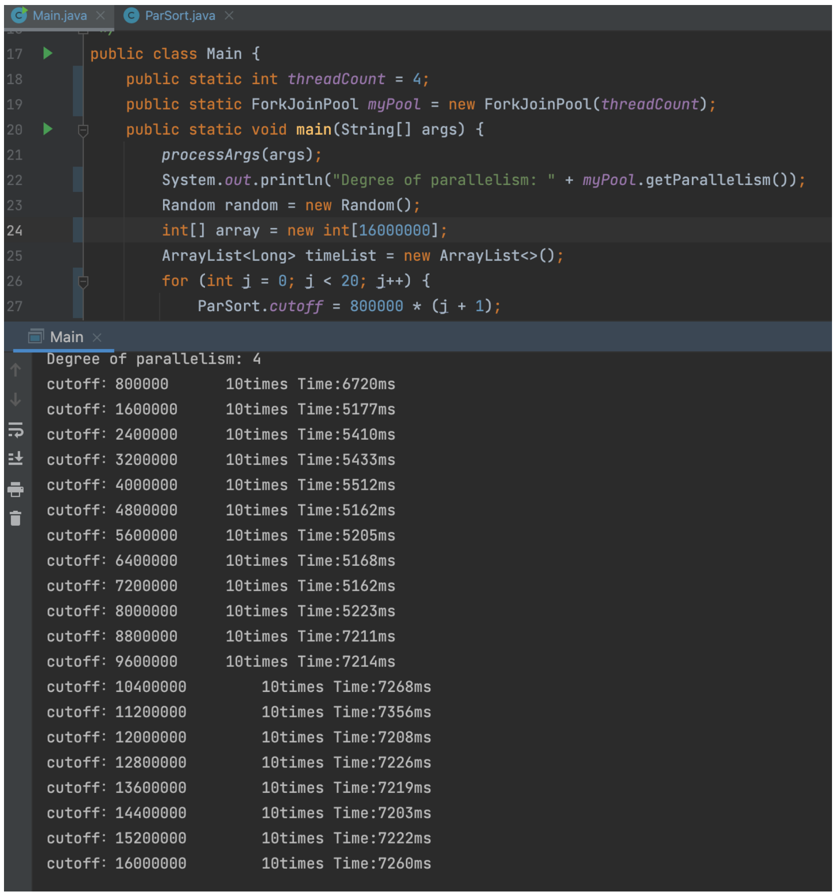
Main.java ParSort.java

```
*/
public class Main {
    public static int threadCount = 8;
    public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[16000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 800000 * (j + 1);
```

Run: Main

```
Degree of parallelism: 16
cutoff: 800000      10times Time:6679ms
cutoff: 1600000     10times Time:4792ms
cutoff: 2400000     10times Time:4498ms
cutoff: 3200000     10times Time:4533ms
cutoff: 4000000     10times Time:4507ms
cutoff: 4800000     10times Time:5327ms
cutoff: 5600000     10times Time:5159ms
cutoff: 6400000     10times Time:5196ms
cutoff: 7200000     10times Time:5174ms
cutoff: 8000000     10times Time:5105ms
cutoff: 8800000     10times Time:7171ms
cutoff: 9600000     10times Time:7139ms
cutoff: 10400000        10times Time:7150ms
cutoff: 11200000        10times Time:7135ms
cutoff: 12000000        10times Time:7138ms
cutoff: 12800000        10times Time:7123ms
cutoff: 13600000        10times Time:7137ms
cutoff: 14400000        10times Time:7161ms
cutoff: 15200000        10times Time:7145ms
cutoff: 16000000        10times Time:7144ms

Process finished with exit code 0
```

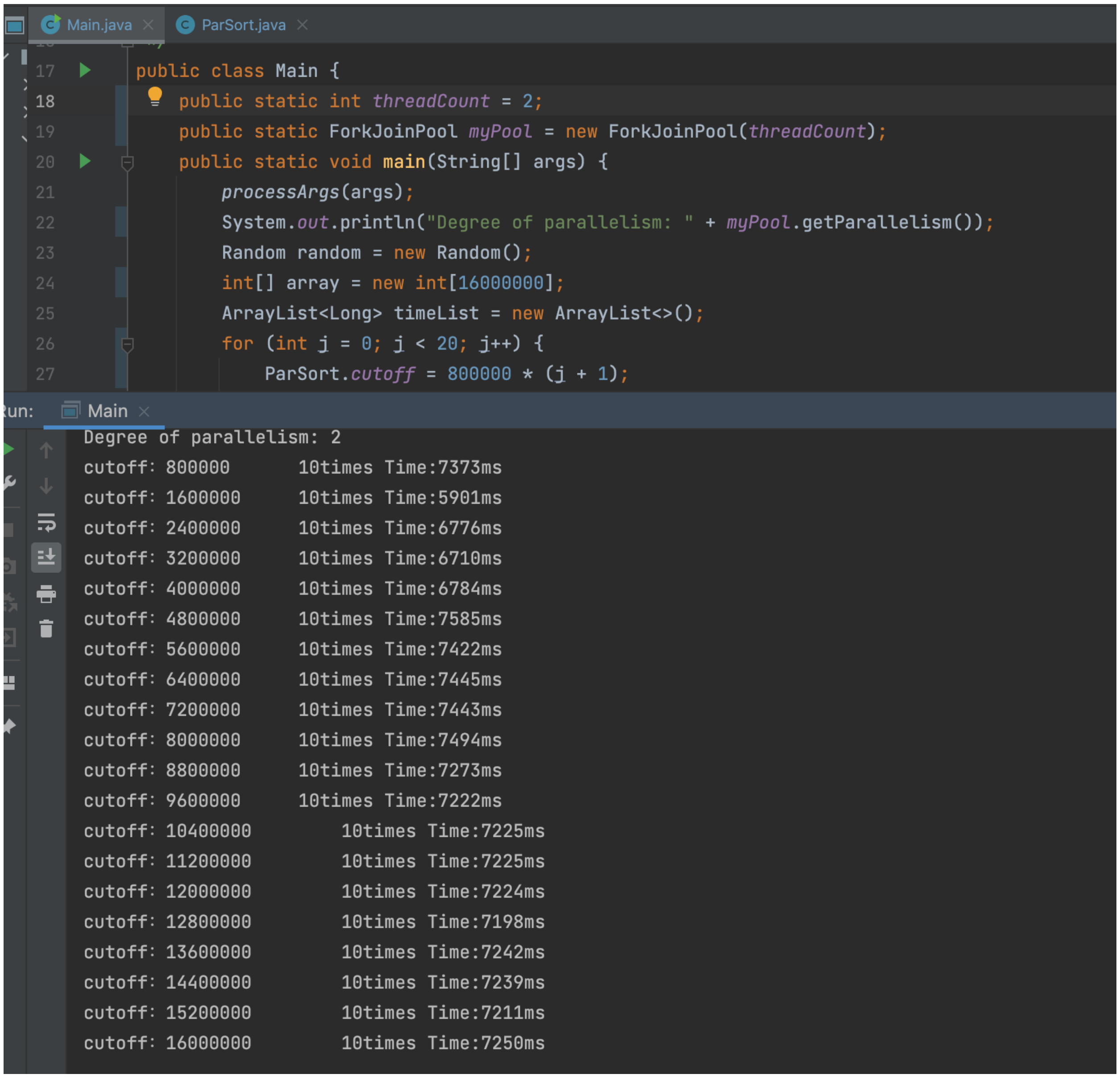


Main.java ParSort.java

```
public class Main {
    public static int threadCount = 4;
    public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[16000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 800000 * (j + 1);
```

Main

```
Degree of parallelism: 4
cutoff: 800000      10times Time:6720ms
cutoff: 1600000     10times Time:5177ms
cutoff: 2400000     10times Time:5410ms
cutoff: 3200000     10times Time:5433ms
cutoff: 4000000     10times Time:5512ms
cutoff: 4800000     10times Time:5162ms
cutoff: 5600000     10times Time:5205ms
cutoff: 6400000     10times Time:5168ms
cutoff: 7200000     10times Time:5162ms
cutoff: 8000000     10times Time:5223ms
cutoff: 8800000     10times Time:7211ms
cutoff: 9600000     10times Time:7214ms
cutoff: 10400000        10times Time:7268ms
cutoff: 11200000        10times Time:7356ms
cutoff: 12000000        10times Time:7208ms
cutoff: 12800000        10times Time:7226ms
cutoff: 13600000        10times Time:7219ms
cutoff: 14400000        10times Time:7203ms
cutoff: 15200000        10times Time:7222ms
cutoff: 16000000        10times Time:7260ms
```



```
public class Main {
    public static int threadCount = 2;
    public static ForkJoinPool myPool = new ForkJoinPool(threadCount);
    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + myPool.getParallelism());
        Random random = new Random();
        int[] array = new int[16000000];
        ArrayList<Long> timeList = new ArrayList<>();
        for (int j = 0; j < 20; j++) {
            ParSort.cutoff = 800000 * (j + 1);
```

```
Degree of parallelism: 2
cutoff: 800000      10times Time:7373ms
cutoff: 1600000      10times Time:5901ms
cutoff: 2400000      10times Time:6776ms
cutoff: 3200000      10times Time:6710ms
cutoff: 4000000      10times Time:6784ms
cutoff: 4800000      10times Time:7585ms
cutoff: 5600000      10times Time:7422ms
cutoff: 6400000      10times Time:7445ms
cutoff: 7200000      10times Time:7443ms
cutoff: 8000000      10times Time:7494ms
cutoff: 8800000      10times Time:7273ms
cutoff: 9600000      10times Time:7222ms
cutoff: 10400000        10times Time:7225ms
cutoff: 11200000        10times Time:7225ms
cutoff: 12000000        10times Time:7224ms
cutoff: 12800000        10times Time:7198ms
cutoff: 13600000        10times Time:7242ms
cutoff: 14400000        10times Time:7239ms
cutoff: 15200000        10times Time:7211ms
cutoff: 16000000        10times Time:7250ms
```