PSA ASSIGNMENT 5

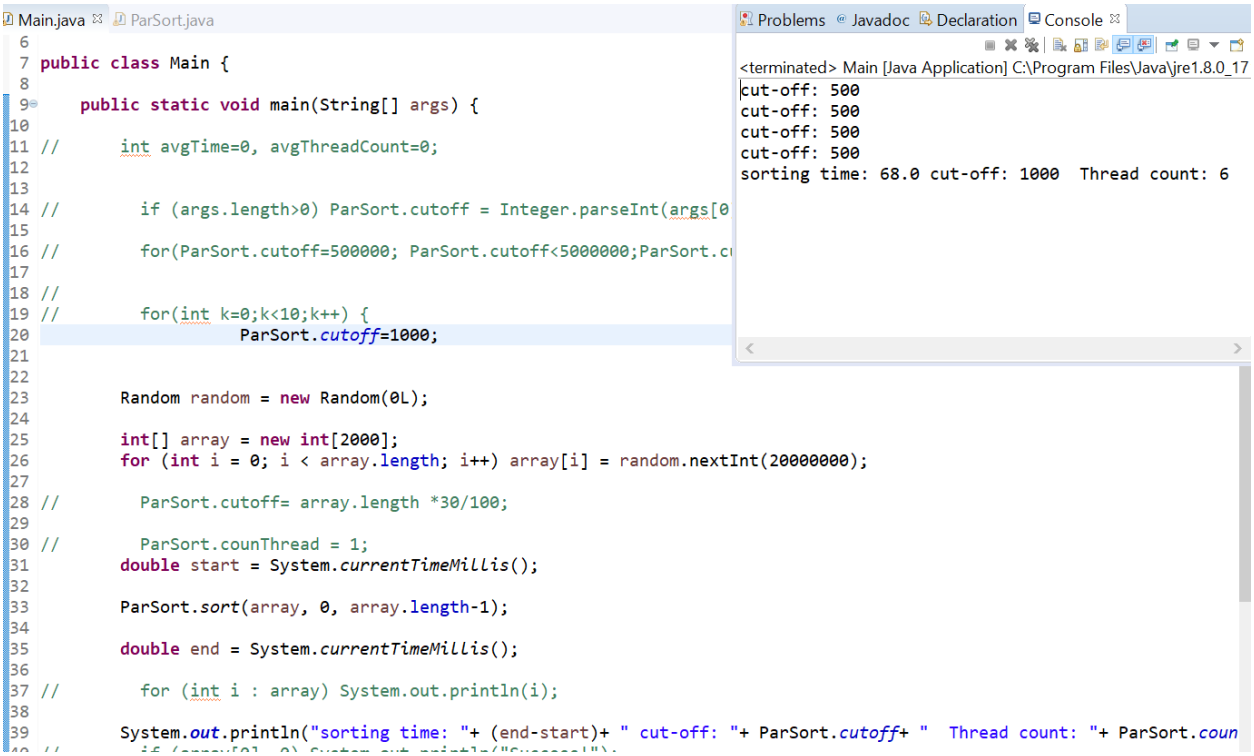
YASH KHOPKAR

CONCLUSION:

We can conclude that as with the increase in the array size performance is increases. Initially when the array size is small system took more time to sort but if be continuously increase the size of the array after certain size we can find that eventually, it will take less time to sort the array and on further increase in the size of the array the time taken to sort the array increases by very small amount.

ARRAY SIZE	CUTOFF	SORTING TIME (MIL SEC)
2000	1000	68.0
4000	500	55.0
4000	1000	63.0
8000	500	68.0
8000	1000	63.0

SCREENSHOTS:



Main.java | ParSort.java

```
 6
 7 public class Main {
 8
 9     public static void main(String[] args) {
10
11 //      int avgTime=0, avgThreadCount=0;
12
13
14 //        if (args.length>0) ParSort.cutoff = Integer.parseInt(args[0
15
16 //        for(ParSort.cutoff=500000; ParSort.cutoff<5000000;ParSort.c
17
18 //
19 //        for(int k=0;k<10;k++) {
20                   ParSort.cutoff=500;
21
22
23         Random random = new Random(0L);
24
25         int[] array = new int[4000];
26         for (int i = 0; i < array.length; i++) array[i] = random.nextInt(20000000);
27
28 //        ParSort.cutoff= array.length *30/100;
29
30 //        ParSort.counThread = 1;
31         double start = System.currentTimeMillis();
32
33         ParSort.sort(array, 0, array.length-1);
34
35         double end = System.currentTimeMillis();
36
37 //        for (int i : array) System.out.println(i);
38
39         System.out.println("sorting time: "+ (end-start)+ " cut-off: "+ ParSort.cutoff+ "  Thread count: "+ ParSort.coun
40 //        if (array[0]==0) System.out.println("Success!");
41 //        System.out.println("Thread count: "+ ParSort.counThread);
```

Problems | Javadoc | Declaration | Console

```
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_17
cut-off: 250
cut-off: 250
cut-off: 250
cut-off: 250
cut-off: 250
cut-off: 250
cut-off: 250
cut-off: 250
cut-off: 250
cut-off: 500
sorting time: 55.0 cut-off: 500  Thread count: 28
```

Main.java | ParSort.java

```
 6
 7 public class Main {
 8
 9     public static void main(String[] args) {
10
11 //      int avgTime=0, avgThreadCount=0;
12
13
14 //        if (args.length>0) ParSort.cutoff = Integer.parseInt(args[0
15
16 //        for(ParSort.cutoff=500000; ParSort.cutoff<5000000;ParSort.c
17
18 //
19 //        for(int k=0;k<10;k++) {
20                   ParSort.cutoff=1000;
21
22
23         Random random = new Random(0L);
24
25         int[] array = new int[4000];
26         for (int i = 0; i < array.length; i++) array[i] = random.nextInt(20000000);
27
28 //        ParSort.cutoff= array.length *30/100;
29
30 //        ParSort.counThread = 1;
31         double start = System.currentTimeMillis();
32
33         ParSort.sort(array, 0, array.length-1);
34
35         double end = System.currentTimeMillis();
36
37 //        for (int i : array) System.out.println(i);
38
39         System.out.println("sorting time: "+ (end-start)+ " cut-off: "+ ParSort.cutoff+ "  Thread count: "+ ParSort.cour
40 //        if (array[0]==0) System.out.println("Success!");
41 //        System.out.println("Thread count: "+ ParSort.counThread);
```

Problems | Javadoc | Declaration | Console

```
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
sorting time: 63.0 cut-off: 1000  Thread count: 14
```

Main.java | ParSort.java

```
10
11 //      int avgTime=0, avgThreadCount=0;
12
13
14 //        if (args.length>0) ParSort.cutoff = Integer.parseInt(args[0
15
16 //        for(ParSort.cutoff=500000; ParSort.cutoff<5000000;ParSort.cu
17
18 //
19 //        for(int k=0;k<10;k++) {
20                   ParSort.cutoff=500;
21
22
23        Random random = new Random(0L);
24
25        int[] array = new int[8000];
26        for (int i = 0; i < array.length; i++) array[i] = random.nextInt(20000000);
27
28 //        ParSort.cutoff= array.length *30/100;
29
30 //        ParSort.counThread = 1;
31        double start = System.currentTimeMillis();
32
33        ParSort.sort(array, 0, array.length-1);
34
35        double end = System.currentTimeMillis();
36
37 //        for (int i : array) System.out.println(i);
38
39        System.out.println("sorting time: "+ (end-start)+ " cut-off: "+ ParSort.cutoff+ "  Thread count: "+ ParSort.coun
```

Problems | Javadoc | Declaration | Console

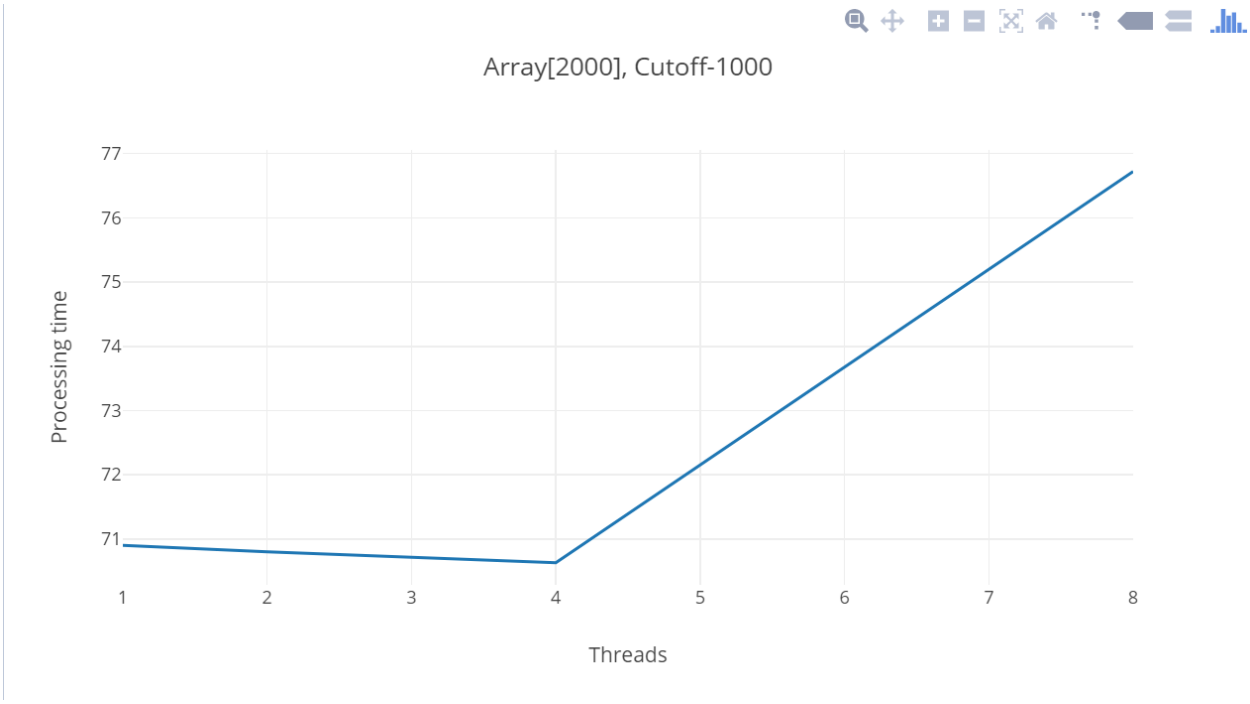
```
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_17
cut-off: 250
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 250
cut-off: 250
cut-off: 1000
cut-off: 1000
cut-off: 250
cut-off: 250
sorting time: 68.0 cut-off: 500  Thread count: 36
```

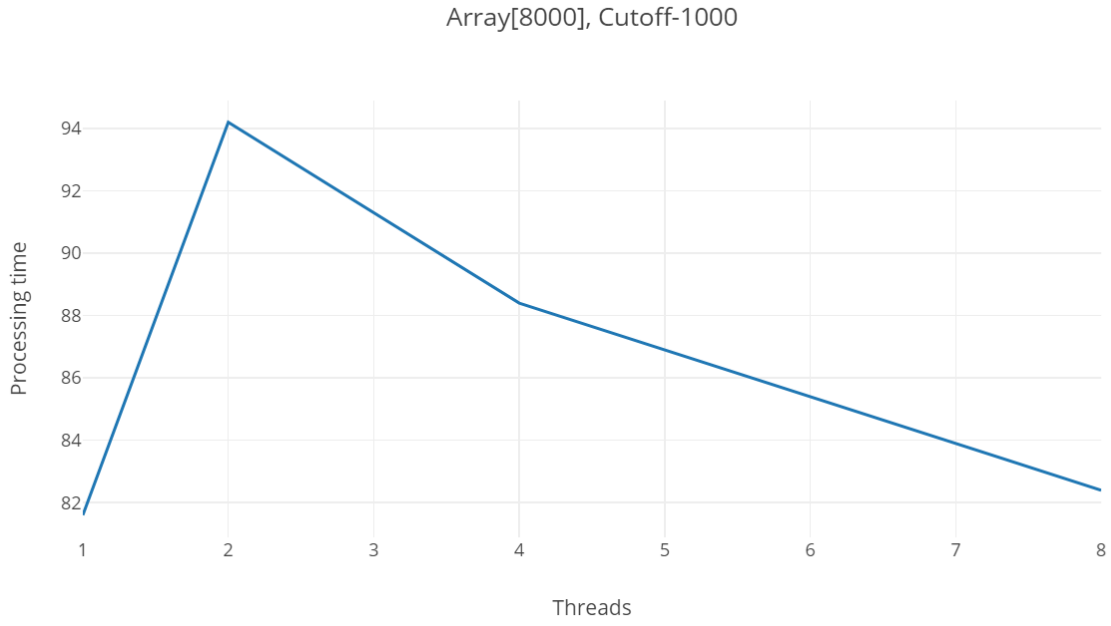
Main.java | ParSort.java

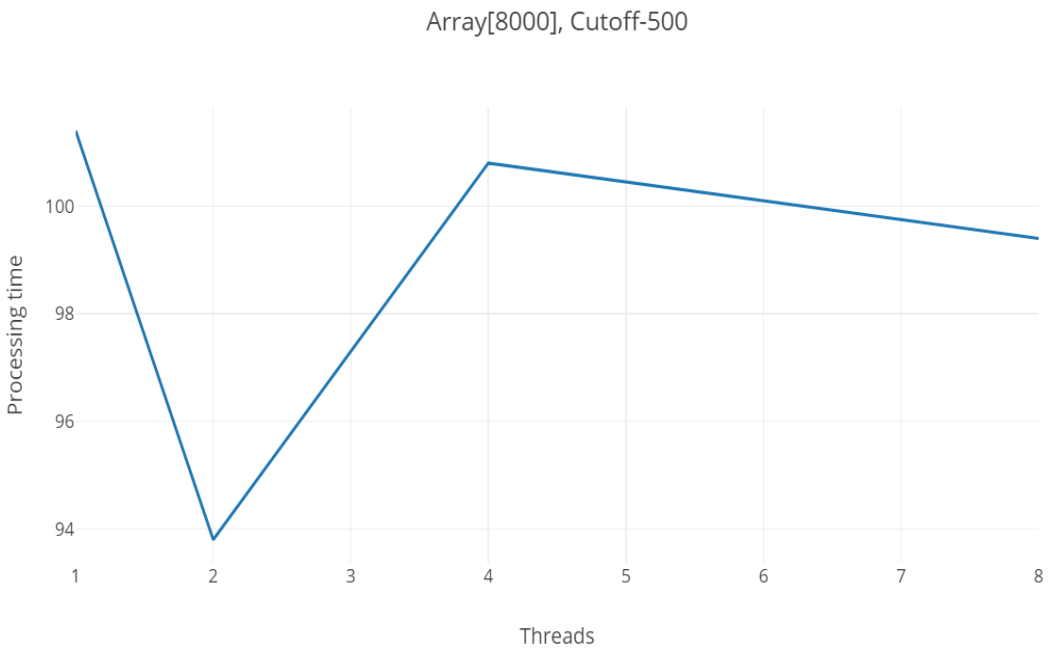
```
10
11 //      int avgTime=0, avgThreadCount=0;
12
13
14 //        if (args.length>0) ParSort.cutoff = Integer.parseInt(args[0
15
16 //        for(ParSort.cutoff=500000; ParSort.cutoff<5000000;ParSort.cu
17
18 //
19 //        for(int k=0;k<10;k++) {
20                   ParSort.cutoff=1000;
21
22
23        Random random = new Random(0L);
24
25        int[] array = new int[8000];
26        for (int i = 0; i < array.length; i++) array[i] = random.nextInt(20000000);
27
28 //        ParSort.cutoff= array.length *30/100;
29
30 //        ParSort.counThread = 1;
31        double start = System.currentTimeMillis();
32
33        ParSort.sort(array, 0, array.length-1);
34
35        double end = System.currentTimeMillis();
36
37 //        for (int i : array) System.out.println(i);
38
39        System.out.println("sorting time: "+ (end-start)+ " cut-off: "+ ParSort.cutoff+ "  Thread count: "+ ParSort.coun
40 //        if (array[0]==0) System.out.println("Success!");
```

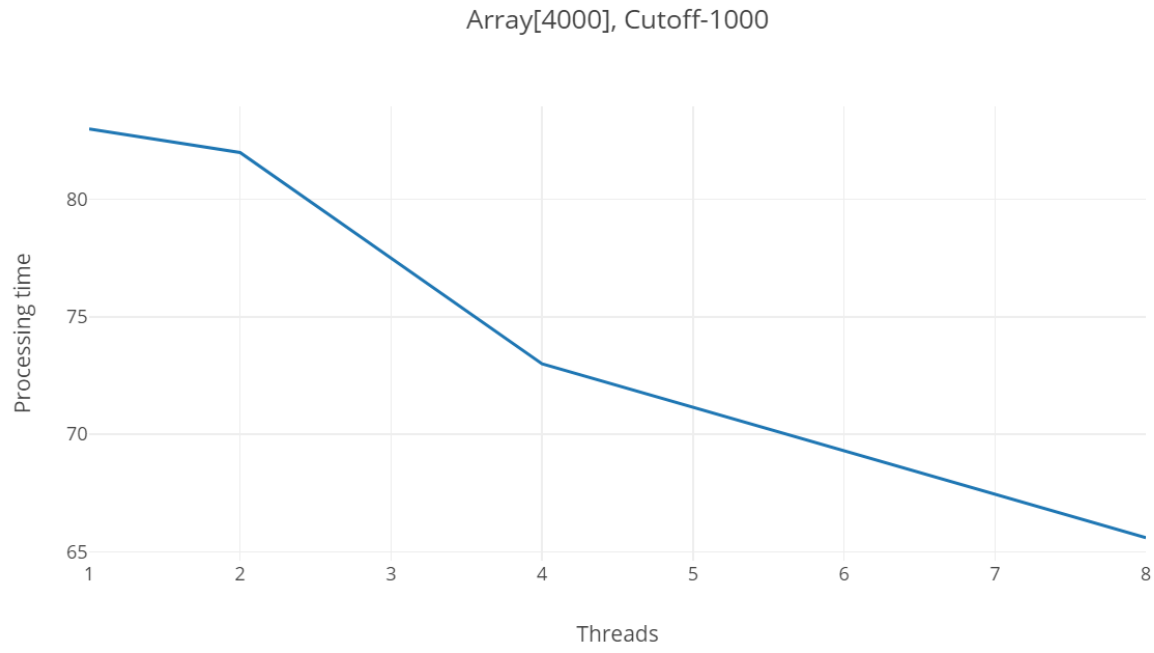
Problems | Javadoc | Declaration | Console

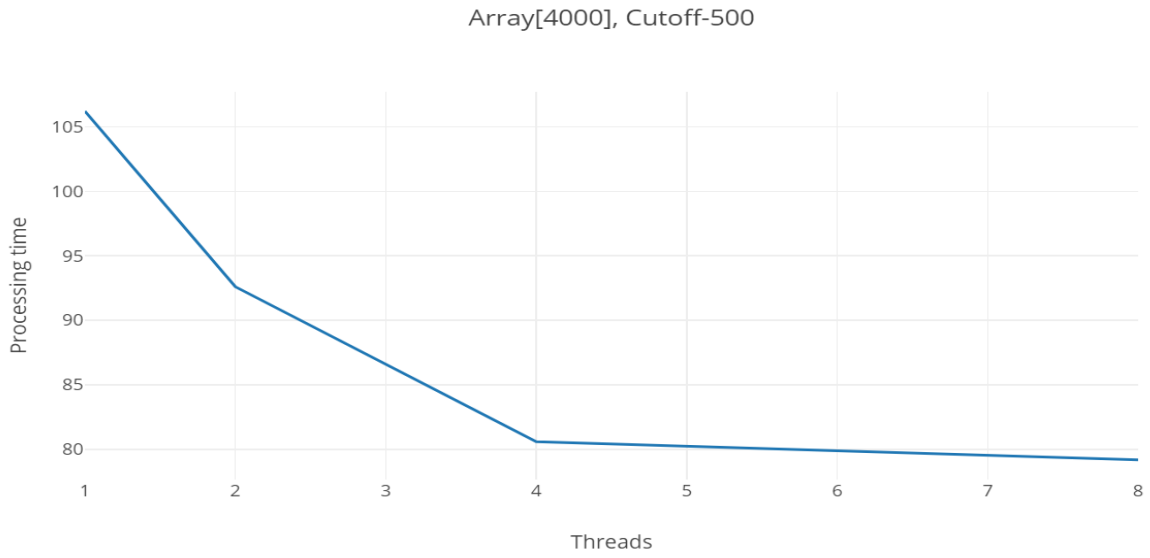
```
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_171
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 500
cut-off: 1000
cut-off: 500
cut-off: 500
sorting time: 63.0 cut-off: 1000  Thread count: 28
```


Array[2000], Cutoff-1000
Processing time
77
76
75
74
73
72
71
1
2
3
4
5
6
7
8
Threads


Array[8000], Cutoff-1000
Processing time
94
92
90
88
86
84
82
1
2
3
4
5
6
7
8
Threads


Array[8000], Cutoff-500
Processing time
100
98
96
94
1
2
3
4
5
6
7
8
Threads


Array[4000], Cutoff-1000
Processing time
80
75
70
65
1
2
3
4
5
6
7
8
Threads


Array[4000], Cutoff-500
Processing time
105
100
95
90
85
80
1
2
3
4
5
6
7
8
Threads