```java
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;

/**
 * This code has been fleshed out by Ziyao Qiao. Thanks very much.
 * CONSIDER tidy it up a bit.
 */
public class Main {

    public static void main(String[] args) {
        processArgs(args);
        System.out.println("Degree of parallelism: " + ForkJoinPool.getCommonPoolParallelism());
        Random random = new Random();
        int[] array = new int[3000000];
        ArrayList<Long> timeList = new ArrayList<>();

        int thread = (int)Math.pow(2,5);
        ParSort.mPool = new ForkJoinPool(thread);
        System.out.println("Number of threads" + thread);



        for (int j = 1; j < 36; j++) {
            ParSort.cutoff = array.length/200 * (j + 1);
            long time;
            long startTime = System.currentTimeMillis();
```

```
36              for (int t = 0; t < 10; t++) {
37                  for (int i = 0; i < array.length; i
   ++) array[i] = random.nextInt(10000000);
38                  ParSort.sort(array, 0, array.length
   );
39              }
40              long endTime = System.currentTimeMillis
   ();
41              time = (endTime - startTime);
42              timeList.add(time);
43
44
45              System.out.println("cutoff: " + (ParSort
   .cutoff) + "\t\t10times Time:" + time + "ms");
46
47          }
48          try {
49              FileOutputStream fis = new
   FileOutputStream("./src/result.csv");
50              OutputStreamWriter isr = new
   OutputStreamWriter(fis);
51              BufferedWriter bw = new BufferedWriter(
   isr);
52              int j = 0;
53              for (long i : timeList) {
54                  String content = (double) 10000 * (
   j + 1) / 2000000 + "," + (double) i / 10 + "\n";
55                  j++;
56                  bw.write(content);
57                  bw.flush();
58              }
59              bw.close();
60
61          } catch (IOException e) {
62              e.printStackTrace();
63          }
64      }
65
66      private static void processArgs(String[] args
   ) {
67          String[] xs = args;
```

```
68          while (xs.length > 0)
69              if (xs[0].startsWith("-")) xs = processArg(xs);
70      }
71
72      private static String[] processArg(String[] xs) {
73          String[] result = new String[0];
74          System.arraycopy(xs, 2, result, 0, xs.length - 2);
75          processCommand(xs[0], xs[1]);
76          return result;
77      }
78
79      private static void processCommand(String x, String y) {
80          if (x.equalsIgnoreCase("N")) setConfig(x, Integer.parseInt(y));
81          else
82              // TODO sort this out
83              if (x.equalsIgnoreCase("P"))
84                  ForkJoinPool.getCommonPoolParallelism();
85      }
86
87      private static void setConfig(String x, int i) {
88          configuration.put(x, i);
89      }
90
91      @SuppressWarnings("MismatchedQueryAndUpdateOfCollection")
92      private static final Map<String, Integer> configuration = new HashMap<>();
93
94
95  }
96
```