

```

1  import java.util.ArrayList;
2
3  public class Factor {
4      public static int total = 600000;
5      public static int numChild = 2;
6      public static int range = total / numChild;
7      public static int begin = 0;
8
9      public static void main(String args []) throws
      InterruptedException {
10         // variable to keep track of thread count
11         int i;
12         long start_s = System.nanoTime();
13         System.err.println("Run Factor " + total +
14             ":" + numChild);
15         //Array list to keep track of threads
16         ArrayList<Thread> threads = new ArrayList
17             <>();
18         //loop to run thru threads
19         for (i = 0; i < numChild; i++) {
20             // Creates new thread
21             Thread thread = new Thread(new
22                 childFactor(begin, begin + range));
23             //adds new thread with factor to
24             ArrayList
25                 threads.add(thread);
26             //starts threads
27             thread.start();
28         }
29         for (Thread each :
30             threads) {
31             each.join();
32         }
33         long stop_s = System.nanoTime();
34         System.err.println("time: " + (stop_s -
35             start_s)/1000000000.0 + " seconds");
36     }
37
38     private static class childFactor implements
39         Runnable{
40         // starting value for factoring
41         int begin;
42         int end;

```

```
38
39     // Constructor
40     public childFactor(int start, int stop){
41         this.begin = start;
42         this.end = stop;
43     }
44
45     @Override
46     public void run(){
47         //long start_s = System.nanoTime();
48         int val, i;
49         for(val = begin; val<end; val++){
50             for(i=2; i<= val/2; i++){
51                 if(val % i == 0){
52                     break;
53                 }
54                 if(i== val / 2){
55                     System.out.println("F:" +
56 val);
57                 }
58             }
59             //long stop_s = System.nanoTime();
60             //System.err.println("time: " + (stop_s
61 - start_s) + " seconds");
62         }
63     }
64 }
```

```

1  import java.util.ArrayList;
2
3  public class SquareRoot {
4      public static int total = 600000;
5      public static int numChild = 8;
6      public static int range = total / numChild;
7      public static int begin = 0;
8      public static double global = 0.0;
9      public static double memAttr;
10
11
12     public static void main(String args []) throws
    InterruptedException {
13         // variable to keep track of thread count
14         int i;
15         long start_s = System.nanoTime();
16         System.err.println("Run Factor " + total +
    ":" + numChild);
17
18         //Array list to keep track of threads
19         ArrayList<Thread> threadList = new
    ArrayList<>();
20         //loop to run thru threads
21         for (i = 0; i < numChild; i++) {
22             // Creates new thread
23             Thread thread = new Thread(new
    childSquareRoot(begin, begin + range));
24             //adds new thread with factor to
    ArrayList
25             threadList.add(thread);
26             //starts threads
27             thread.start();
28         }
29         for (Thread each :
30             threadList) {
31             each.join();
32         }
33         long stop_s = System.nanoTime();
34         System.err.println("time: " + (stop_s -
    start_s)/1000000000.0 + " seconds");
35
36     }
37
38     private static class childSquareRoot implements

```

```

38  Runnable{
39      // starting value for factoring
40      int begin;
41      int end;
42
43      // Constructor
44      public childSquareRoot(int start, int stop
45  ){
46          this.begin = start;
47          this.end = stop;
48      }
49      @Override
50      public void run(){
51          System.err.println("CPU: " + Runtime.
52          getRuntime().availableProcessors());
53          //long start_s = System.nanoTime();
54          double totalSum = 0.0;
55          for (int local = (int) begin; local <
56          end; local++) {
57              double root = Math.sqrt(local);
58              //revise lines to do prints
59              System.out.println("Number: " +
60              local + " : SquareRoot of Number: " + root + " ");
61              if (local % 5 == 0) {
62                  totalSum += root;
63              }
64          }
65          System.err.println("    totalSum = " +
66          totalSum + "    global = " + ++global + " memAttr = "
67          + memAttr);
68          //long stop_s = System.nanoTime();
69          //System.err.println("time: " + (stop_s
70          - start_s) + " seconds");
71      }
72  }
73  }

```