```
/Users/zuzannafurtak/Library/Java/JavaVirtualMachines/
corretto-17.0.8/Contents/Home/bin/java -javaagent:/
Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.
jar=57630:/Applications/IntelliJ IDEA.app/Contents/bin
 -Dfile.encoding=UTF-8 -classpath /Users/zuzannafurtak
/studia/las_palmas/big_data/big-data-tasks/assignment-
3/target/classes:/Users/zuzannafurtak/.m2/repository/
org/openjdk/jmh/jmh-core/1.32/jmh-core-1.32.jar:/Users
/zuzannafurtak/.m2/repository/net/sf/jopt-simple/jopt-
simple/4.6/jopt-simple-4.6.jar:/Users/zuzannafurtak/.
m2/repository/org/apache/commons/commons-math3/3.2/
commons-math3-3.2.jar:/Users/zuzannafurtak/.m2/
repository/org/openjdk/jmh/jmh-generator-annprocess/1.
32/jmh-generator-annprocess-1.32.jar com.matrix.
MultiplicationBenchmark
# JMH version: 1.32
# VM version: JDK 17.0.8, OpenJDK 64-Bit Server VM, 17
.0.8+7-LTS
# VM invoker: /Users/zuzannafurtak/Library/Java/
JavaVirtualMachines/corretto-17.0.8/Contents/Home/bin/
java
# VM options: -javaagent:/Applications/IntelliJ IDEA.
app/Contents/lib/idea_rt.jar=57630:/Applications/
IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8
# Blackhole mode: full + dont-inline hint
# Warmup: 5 iterations, 1 s each
# Measurement: 10 iterations, 1 s each
# Timeout: 10 min per iteration
# Threads: 1 thread, will synchronize iterations
# Benchmark mode: Sampling time
# Benchmark: com.matrix.MultiplicationBenchmark.
multiplyParallelBlocking
# Parameters: (n = 1000)

# Run progress: 0.00% complete, ETA 00:00:45
# Fork: 1 of 1
# Warmup Iteration   1: 50.473 ±(99.9%) 9.937 ms/op
# Warmup Iteration   2: 48.568 ±(99.9%) 2.139 ms/op
# Warmup Iteration   3: 47.383 ±(99.9%) 0.726 ms/op
# Warmup Iteration   4: 47.058 ±(99.9%) 0.769 ms/op
# Warmup Iteration   5: 47.138 ±(99.9%) 0.595 ms/op
```

```
Iteration   1: 48.047 ±(99.9%) 0.776 ms/op
                 multiplyParallelBlocking·p0.00:   46.
334 ms/op
                 multiplyParallelBlocking·p0.50:   48.
103 ms/op
                 multiplyParallelBlocking·p0.90:   49.
388 ms/op
                 multiplyParallelBlocking·p0.95:   49.
709 ms/op
                 multiplyParallelBlocking·p0.99:   49.
742 ms/op
                 multiplyParallelBlocking·p0.999:  49.
742 ms/op
                 multiplyParallelBlocking·p0.9999: 49.
742 ms/op
                 multiplyParallelBlocking·p1.00:   49.
742 ms/op

Iteration   2: 48.044 ±(99.9%) 0.698 ms/op
                 multiplyParallelBlocking·p0.00:   46.
531 ms/op
                 multiplyParallelBlocking·p0.50:   48.
234 ms/op
                 multiplyParallelBlocking·p0.90:   49.
165 ms/op
                 multiplyParallelBlocking·p0.95:   49.
748 ms/op
                 multiplyParallelBlocking·p0.99:   49.
807 ms/op
                 multiplyParallelBlocking·p0.999:  49.
807 ms/op
                 multiplyParallelBlocking·p0.9999: 49.
807 ms/op
                 multiplyParallelBlocking·p1.00:   49.
807 ms/op

Iteration   3: 48.796 ±(99.9%) 1.201 ms/op
                 multiplyParallelBlocking·p0.00:   46.
531 ms/op
                 multiplyParallelBlocking·p0.50:   49.
086 ms/op
```

```
                 multiplyParallelBlocking·p0.90:   50.
515 ms/op
                 multiplyParallelBlocking·p0.95:   50.
528 ms/op
                 multiplyParallelBlocking·p0.99:   50.
528 ms/op
                 multiplyParallelBlocking·p0.999:  50.
528 ms/op
                 multiplyParallelBlocking·p0.9999: 50.
528 ms/op
                 multiplyParallelBlocking·p1.00:   50.
528 ms/op

Iteration   4: 47.564 ±(99.9%) 1.241 ms/op
                 multiplyParallelBlocking·p0.00:   46.
268 ms/op
                 multiplyParallelBlocking·p0.50:   47.
022 ms/op
                 multiplyParallelBlocking·p0.90:   50.
941 ms/op
                 multiplyParallelBlocking·p0.95:   51.
315 ms/op
                 multiplyParallelBlocking·p0.99:   51.
315 ms/op
                 multiplyParallelBlocking·p0.999:  51.
315 ms/op
                 multiplyParallelBlocking·p0.9999: 51.
315 ms/op
                 multiplyParallelBlocking·p1.00:   51.
315 ms/op

Iteration   5: 47.894 ±(99.9%) 1.267 ms/op
                 multiplyParallelBlocking·p0.00:   46.
465 ms/op
                 multiplyParallelBlocking·p0.50:   47.
776 ms/op
                 multiplyParallelBlocking·p0.90:   50.
201 ms/op
                 multiplyParallelBlocking·p0.95:   52.
763 ms/op
                 multiplyParallelBlocking·p0.99:   53.
```

```
019 ms/op
                 multiplyParallelBlocking·p0.999:  53.
019 ms/op
                 multiplyParallelBlocking·p0.9999: 53.
019 ms/op
                 multiplyParallelBlocking·p1.00:   53.
019 ms/op

Iteration   6: 50.348 ±(99.9%) 2.065 ms/op
                 multiplyParallelBlocking·p0.00:   47.
317 ms/op
                 multiplyParallelBlocking·p0.50:   50.
135 ms/op
                 multiplyParallelBlocking·p0.90:   54.
054 ms/op
                 multiplyParallelBlocking·p0.95:   56.
001 ms/op
                 multiplyParallelBlocking·p0.99:   56.
099 ms/op
                 multiplyParallelBlocking·p0.999:  56.
099 ms/op
                 multiplyParallelBlocking·p0.9999: 56.
099 ms/op
                 multiplyParallelBlocking·p1.00:   56.
099 ms/op

Iteration   7: 47.979 ±(99.9%) 1.356 ms/op
                 multiplyParallelBlocking·p0.00:   46.
465 ms/op
                 multiplyParallelBlocking·p0.50:   47.
776 ms/op
                 multiplyParallelBlocking·p0.90:   50.
607 ms/op
                 multiplyParallelBlocking·p0.95:   52.
855 ms/op
                 multiplyParallelBlocking·p0.99:   53.
084 ms/op
                 multiplyParallelBlocking·p0.999:  53.
084 ms/op
                 multiplyParallelBlocking·p0.9999: 53.
084 ms/op
```

```
                  multiplyParallelBlocking·p1.00:   53.
084 ms/op

Iteration   8: 50.088 ±(99.9%) 3.667 ms/op
                  multiplyParallelBlocking·p0.00:   47.
579 ms/op
                  multiplyParallelBlocking·p0.50:   48.
169 ms/op
                  multiplyParallelBlocking·p0.90:   57.
462 ms/op
                  multiplyParallelBlocking·p0.95:   64.
546 ms/op
                  multiplyParallelBlocking·p0.99:   65.
274 ms/op
                  multiplyParallelBlocking·p0.999:  65.
274 ms/op
                  multiplyParallelBlocking·p0.9999: 65.
274 ms/op
                  multiplyParallelBlocking·p1.00:   65.
274 ms/op

Iteration   9: 48.559 ±(99.9%) 2.118 ms/op
                  multiplyParallelBlocking·p0.00:   46.
399 ms/op
                  multiplyParallelBlocking·p0.50:   47.
907 ms/op
                  multiplyParallelBlocking·p0.90:   51.
708 ms/op
                  multiplyParallelBlocking·p0.95:   56.
433 ms/op
                  multiplyParallelBlocking·p0.99:   56.
951 ms/op
                  multiplyParallelBlocking·p0.999:  56.
951 ms/op
                  multiplyParallelBlocking·p0.9999: 56.
951 ms/op
                  multiplyParallelBlocking·p1.00:   56.
951 ms/op

Iteration  10: 51.528 ±(99.9%) 6.093 ms/op
                  multiplyParallelBlocking·p0.00:   47.
```

```
514 ms/op
                 multiplyParallelBlocking·p0.50:   48.
464 ms/op
                 multiplyParallelBlocking·p0.90:   63.
970 ms/op
                 multiplyParallelBlocking·p0.95:   75.
340 ms/op
                 multiplyParallelBlocking·p0.99:   75.
891 ms/op
                 multiplyParallelBlocking·p0.999:  75.
891 ms/op
                 multiplyParallelBlocking·p0.9999: 75.
891 ms/op
                 multiplyParallelBlocking·p1.00:   75.
891 ms/op



Result "com.matrix.MultiplicationBenchmark.
multiplyParallelBlocking":
  N = 209
  mean =     48.859 ±(99.9%) 0.732 ms/op

  Histogram, ms/op:
    [40.000, 42.500) = 0
    [42.500, 45.000) = 0
    [45.000, 47.500) = 58
    [47.500, 50.000) = 111
    [50.000, 52.500) = 26
    [52.500, 55.000) = 6
    [55.000, 57.500) = 4
    [57.500, 60.000) = 1
    [60.000, 62.500) = 0
    [62.500, 65.000) = 1
    [65.000, 67.500) = 1
    [67.500, 70.000) = 0
    [70.000, 72.500) = 0
    [72.500, 75.000) = 0
    [75.000, 77.500) = 1

  Percentiles, ms/op:
```

```
       p(0.0000) =     46.268 ms/op
      p(50.0000) =     48.103 ms/op
      p(90.0000) =     51.249 ms/op
      p(95.0000) =     53.543 ms/op
      p(99.0000) =     65.235 ms/op
      p(99.9000) =     75.891 ms/op
      p(99.9900) =     75.891 ms/op
      p(99.9990) =     75.891 ms/op
      p(99.9999) =     75.891 ms/op
     p(100.0000) =     75.891 ms/op


# JMH version: 1.32
# VM version: JDK 17.0.8, OpenJDK 64-Bit Server VM, 17
.0.8+7-LTS
# VM invoker: /Users/zuzannafurtak/Library/Java/
JavaVirtualMachines/corretto-17.0.8/Contents/Home/bin/
java
# VM options: -javaagent:/Applications/IntelliJ IDEA.
app/Contents/lib/idea_rt.jar=57630:/Applications/
IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8
# Blackhole mode: full + dont-inline hint
# Warmup: 5 iterations, 1 s each
# Measurement: 10 iterations, 1 s each
# Timeout: 10 min per iteration
# Threads: 1 thread, will synchronize iterations
# Benchmark mode: Sampling time
# Benchmark: com.matrix.MultiplicationBenchmark.
multiplyParallelBlocking
# Parameters: (n = 1500)

# Run progress: 33.33% complete, ETA 00:00:31
# Fork: 1 of 1
# Warmup Iteration   1: 383.255 ±(99.9%) 524.506 ms/op
# Warmup Iteration   2: 389.371 ±(99.9%) 509.404 ms/op
# Warmup Iteration   3: 357.739 ±(99.9%) 52.680 ms/op
# Warmup Iteration   4: 366.477 ±(99.9%) 695.880 ms/op
# Warmup Iteration   5: 358.963 ±(99.9%) 177.747 ms/op
Iteration   1: 355.118 ±(99.9%) 30.747 ms/op
                 multiplyParallelBlocking·p0.00:   353
.894 ms/op
```

```
                  multiplyParallelBlocking·p0.50:   354
.419 ms/op
                  multiplyParallelBlocking·p0.90:   357
.040 ms/op
                  multiplyParallelBlocking·p0.95:   357
.040 ms/op
                  multiplyParallelBlocking·p0.99:   357
.040 ms/op
                  multiplyParallelBlocking·p0.999:  357
.040 ms/op
                  multiplyParallelBlocking·p0.9999: 357
.040 ms/op
                  multiplyParallelBlocking·p1.00:   357
.040 ms/op

Iteration   2: 363.157 ±(99.9%) 295.174 ms/op
                  multiplyParallelBlocking·p0.00:   351
.797 ms/op
                  multiplyParallelBlocking·p0.50:   355
.992 ms/op
                  multiplyParallelBlocking·p0.90:   381
.682 ms/op
                  multiplyParallelBlocking·p0.95:   381
.682 ms/op
                  multiplyParallelBlocking·p0.99:   381
.682 ms/op
                  multiplyParallelBlocking·p0.999:  381
.682 ms/op
                  multiplyParallelBlocking·p0.9999: 381
.682 ms/op
                  multiplyParallelBlocking·p1.00:   381
.682 ms/op

Iteration   3: 355.467 ±(99.9%) 25.307 ms/op
                  multiplyParallelBlocking·p0.00:   354
.419 ms/op
                  multiplyParallelBlocking·p0.50:   354
.943 ms/op
                  multiplyParallelBlocking·p0.90:   357
.040 ms/op
                  multiplyParallelBlocking·p0.95:   357
```

```
.040 ms/op
                 multiplyParallelBlocking·p0.99:   357
.040 ms/op
                 multiplyParallelBlocking·p0.999:  357
.040 ms/op
                 multiplyParallelBlocking·p0.9999: 357
.040 ms/op
                 multiplyParallelBlocking·p1.00:   357
.040 ms/op

Iteration   4: 359.137 ±(99.9%) 132.881 ms/op
                 multiplyParallelBlocking·p0.00:   354
.419 ms/op
                 multiplyParallelBlocking·p0.50:   355
.467 ms/op
                 multiplyParallelBlocking·p0.90:   367
.526 ms/op
                 multiplyParallelBlocking·p0.95:   367
.526 ms/op
                 multiplyParallelBlocking·p0.99:   367
.526 ms/op
                 multiplyParallelBlocking·p0.999:  367
.526 ms/op
                 multiplyParallelBlocking·p0.9999: 367
.526 ms/op
                 multiplyParallelBlocking·p1.00:   367
.526 ms/op

Iteration   5: 364.205 ±(99.9%) 224.386 ms/op
                 multiplyParallelBlocking·p0.00:   354
.419 ms/op
                 multiplyParallelBlocking·p0.50:   360
.186 ms/op
                 multiplyParallelBlocking·p0.90:   378
.012 ms/op
                 multiplyParallelBlocking·p0.95:   378
.012 ms/op
                 multiplyParallelBlocking·p0.99:   378
.012 ms/op
                 multiplyParallelBlocking·p0.999:  378
.012 ms/op
```

```
                 multiplyParallelBlocking·p0.9999: 378
.012 ms/op
                 multiplyParallelBlocking·p1.00:   378
.012 ms/op

Iteration   6: 353.894 ±(99.9%) 41.693 ms/op
                 multiplyParallelBlocking·p0.00:   351
.273 ms/op
                 multiplyParallelBlocking·p0.50:   354
.943 ms/op
                 multiplyParallelBlocking·p0.90:   355
.467 ms/op
                 multiplyParallelBlocking·p0.95:   355
.467 ms/op
                 multiplyParallelBlocking·p0.99:   355
.467 ms/op
                 multiplyParallelBlocking·p0.999:  355
.467 ms/op
                 multiplyParallelBlocking·p0.9999: 355
.467 ms/op
                 multiplyParallelBlocking·p1.00:   355
.467 ms/op

Iteration   7: 353.370 ±(99.9%) 50.613 ms/op
                 multiplyParallelBlocking·p0.00:   351
.273 ms/op
                 multiplyParallelBlocking·p0.50:   352
.322 ms/op
                 multiplyParallelBlocking·p0.90:   356
.516 ms/op
                 multiplyParallelBlocking·p0.95:   356
.516 ms/op
                 multiplyParallelBlocking·p0.99:   356
.516 ms/op
                 multiplyParallelBlocking·p0.999:  356
.516 ms/op
                 multiplyParallelBlocking·p0.9999: 356
.516 ms/op
                 multiplyParallelBlocking·p1.00:   356
.516 ms/op
```

```
Iteration   8: 352.671 ±(99.9%) 14.611 ms/op
                 multiplyParallelBlocking·p0.00:   351
.797 ms/op
                 multiplyParallelBlocking·p0.50:   352
.846 ms/op
                 multiplyParallelBlocking·p0.90:   353
.370 ms/op
                 multiplyParallelBlocking·p0.95:   353
.370 ms/op
                 multiplyParallelBlocking·p0.99:   353
.370 ms/op
                 multiplyParallelBlocking·p0.999:  353
.370 ms/op
                 multiplyParallelBlocking·p0.9999: 353
.370 ms/op
                 multiplyParallelBlocking·p1.00:   353
.370 ms/op

Iteration   9: 360.361 ±(99.9%) 161.286 ms/op
                 multiplyParallelBlocking·p0.00:   350
.224 ms/op
                 multiplyParallelBlocking·p0.50:   364
.380 ms/op
                 multiplyParallelBlocking·p0.90:   366
.477 ms/op
                 multiplyParallelBlocking·p0.95:   366
.477 ms/op
                 multiplyParallelBlocking·p0.99:   366
.477 ms/op
                 multiplyParallelBlocking·p0.999:  366
.477 ms/op
                 multiplyParallelBlocking·p0.9999: 366
.477 ms/op
                 multiplyParallelBlocking·p1.00:   366
.477 ms/op

Iteration  10: 355.118 ±(99.9%) 105.793 ms/op
                 multiplyParallelBlocking·p0.00:   349
.700 ms/op
                 multiplyParallelBlocking·p0.50:   354
.419 ms/op
```

```
                  multiplyParallelBlocking·p0.90:   361
.234 ms/op
                  multiplyParallelBlocking·p0.95:   361
.234 ms/op
                  multiplyParallelBlocking·p0.99:   361
.234 ms/op
                  multiplyParallelBlocking·p0.999:  361
.234 ms/op
                  multiplyParallelBlocking·p0.9999: 361
.234 ms/op
                  multiplyParallelBlocking·p1.00:   361
.234 ms/op



Result "com.matrix.MultiplicationBenchmark.
multiplyParallelBlocking":
  N = 30
  mean =    357.250 ±(99.9%) 5.056 ms/op

  Histogram, ms/op:
    [340.000, 345.000) = 0
    [345.000, 350.000) = 1
    [350.000, 355.000) = 16
    [355.000, 360.000) = 6
    [360.000, 365.000) = 3
    [365.000, 370.000) = 2
    [370.000, 375.000) = 0
    [375.000, 380.000) = 1
    [380.000, 385.000) = 1

  Percentiles, ms/op:
      p(0.0000) =    349.700 ms/op
     p(50.0000) =    354.681 ms/op
     p(90.0000) =    367.421 ms/op
     p(95.0000) =    379.663 ms/op
     p(99.0000) =    381.682 ms/op
     p(99.9000) =    381.682 ms/op
     p(99.9900) =    381.682 ms/op
     p(99.9990) =    381.682 ms/op
     p(99.9999) =    381.682 ms/op
```

```
    p(100.0000) =    381.682 ms/op


# JMH version: 1.32
# VM version: JDK 17.0.8, OpenJDK 64-Bit Server VM, 17
.0.8+7-LTS
# VM invoker: /Users/zuzannafurtak/Library/Java/
JavaVirtualMachines/corretto-17.0.8/Contents/Home/bin/
java
# VM options: -javaagent:/Applications/IntelliJ IDEA.
app/Contents/lib/idea_rt.jar=57630:/Applications/
IntelliJ IDEA.app/Contents/bin -Dfile.encoding=UTF-8
# Blackhole mode: full + dont-inline hint
# Warmup: 5 iterations, 1 s each
# Measurement: 10 iterations, 1 s each
# Timeout: 10 min per iteration
# Threads: 1 thread, will synchronize iterations
# Benchmark mode: Sampling time
# Benchmark: com.matrix.MultiplicationBenchmark.
multiplyParallelBlocking
# Parameters: (n = 2000)

# Run progress: 66.67% complete, ETA 00:00:16
# Fork: 1 of 1
# Warmup Iteration   1: 1038.090 ms/op
# Warmup Iteration   2: 997.720 ms/op
# Warmup Iteration   3: 979.370 ms/op
# Warmup Iteration   4: 987.759 ms/op
# Warmup Iteration   5: 930.087 ms/op
Iteration   1: 944.767 ms/op
                 multiplyParallelBlocking·p0.00:   940
.573 ms/op
                 multiplyParallelBlocking·p0.50:   944
.767 ms/op
                 multiplyParallelBlocking·p0.90:   948
.961 ms/op
                 multiplyParallelBlocking·p0.95:   948
.961 ms/op
                 multiplyParallelBlocking·p0.99:   948
.961 ms/op
                 multiplyParallelBlocking·p0.999:  948
```

```
.961 ms/op
                 multiplyParallelBlocking·p0.9999: 948
.961 ms/op
                 multiplyParallelBlocking·p1.00:   948
.961 ms/op

Iteration   2: 944.243 ms/op
                 multiplyParallelBlocking·p0.00:   927
.990 ms/op
                 multiplyParallelBlocking·p0.50:   944
.243 ms/op
                 multiplyParallelBlocking·p0.90:   960
.496 ms/op
                 multiplyParallelBlocking·p0.95:   960
.496 ms/op
                 multiplyParallelBlocking·p0.99:   960
.496 ms/op
                 multiplyParallelBlocking·p0.999:  960
.496 ms/op
                 multiplyParallelBlocking·p0.9999: 960
.496 ms/op
                 multiplyParallelBlocking·p1.00:   960
.496 ms/op

Iteration   3: 919.077 ms/op
                 multiplyParallelBlocking·p0.00:   910
.164 ms/op
                 multiplyParallelBlocking·p0.50:   919
.077 ms/op
                 multiplyParallelBlocking·p0.90:   927
.990 ms/op
                 multiplyParallelBlocking·p0.95:   927
.990 ms/op
                 multiplyParallelBlocking·p0.99:   927
.990 ms/op
                 multiplyParallelBlocking·p0.999:  927
.990 ms/op
                 multiplyParallelBlocking·p0.9999: 927
.990 ms/op
                 multiplyParallelBlocking·p1.00:   927
.990 ms/op
```

```
Iteration   4: 948.961 ms/op
                 multiplyParallelBlocking·p0.00:   947
.913 ms/op
                 multiplyParallelBlocking·p0.50:   948
.961 ms/op
                 multiplyParallelBlocking·p0.90:   950
.010 ms/op
                 multiplyParallelBlocking·p0.95:   950
.010 ms/op
                 multiplyParallelBlocking·p0.99:   950
.010 ms/op
                 multiplyParallelBlocking·p0.999:  950
.010 ms/op
                 multiplyParallelBlocking·p0.9999: 950
.010 ms/op
                 multiplyParallelBlocking·p1.00:   950
.010 ms/op

Iteration   5: 963.117 ms/op
                 multiplyParallelBlocking·p0.00:   957
.350 ms/op
                 multiplyParallelBlocking·p0.50:   963
.117 ms/op
                 multiplyParallelBlocking·p0.90:   968
.884 ms/op
                 multiplyParallelBlocking·p0.95:   968
.884 ms/op
                 multiplyParallelBlocking·p0.99:   968
.884 ms/op
                 multiplyParallelBlocking·p0.999:  968
.884 ms/op
                 multiplyParallelBlocking·p0.9999: 968
.884 ms/op
                 multiplyParallelBlocking·p1.00:   968
.884 ms/op

Iteration   6: 956.826 ms/op
                 multiplyParallelBlocking·p0.00:   954
.204 ms/op
                 multiplyParallelBlocking·p0.50:   956
```

```
.826 ms/op
                 multiplyParallelBlocking·p0.90:   959
.447 ms/op
                 multiplyParallelBlocking·p0.95:   959
.447 ms/op
                 multiplyParallelBlocking·p0.99:   959
.447 ms/op
                 multiplyParallelBlocking·p0.999:  959
.447 ms/op
                 multiplyParallelBlocking·p0.9999: 959
.447 ms/op
                 multiplyParallelBlocking·p1.00:   959
.447 ms/op

Iteration   7: 963.641 ms/op
                 multiplyParallelBlocking·p0.00:   961
.544 ms/op
                 multiplyParallelBlocking·p0.50:   963
.641 ms/op
                 multiplyParallelBlocking·p0.90:   965
.738 ms/op
                 multiplyParallelBlocking·p0.95:   965
.738 ms/op
                 multiplyParallelBlocking·p0.99:   965
.738 ms/op
                 multiplyParallelBlocking·p0.999:  965
.738 ms/op
                 multiplyParallelBlocking·p0.9999: 965
.738 ms/op
                 multiplyParallelBlocking·p1.00:   965
.738 ms/op

Iteration   8: 964.690 ms/op
                 multiplyParallelBlocking·p0.00:   957
.350 ms/op
                 multiplyParallelBlocking·p0.50:   964
.690 ms/op
                 multiplyParallelBlocking·p0.90:   972
.030 ms/op
                 multiplyParallelBlocking·p0.95:   972
.030 ms/op
```

```
                  multiplyParallelBlocking·p0.99:   972
.030 ms/op
                  multiplyParallelBlocking·p0.999:  972
.030 ms/op
                  multiplyParallelBlocking·p0.9999: 972
.030 ms/op
                  multiplyParallelBlocking·p1.00:   972
.030 ms/op

Iteration   9: 950.010 ms/op
                  multiplyParallelBlocking·p0.00:   946
.864 ms/op
                  multiplyParallelBlocking·p0.50:   950
.010 ms/op
                  multiplyParallelBlocking·p0.90:   953
.156 ms/op
                  multiplyParallelBlocking·p0.95:   953
.156 ms/op
                  multiplyParallelBlocking·p0.99:   953
.156 ms/op
                  multiplyParallelBlocking·p0.999:  953
.156 ms/op
                  multiplyParallelBlocking·p0.9999: 953
.156 ms/op
                  multiplyParallelBlocking·p1.00:   953
.156 ms/op

Iteration  10: 953.156 ms/op
                  multiplyParallelBlocking·p0.00:   951
.058 ms/op
                  multiplyParallelBlocking·p0.50:   953
.156 ms/op
                  multiplyParallelBlocking·p0.90:   955
.253 ms/op
                  multiplyParallelBlocking·p0.95:   955
.253 ms/op
                  multiplyParallelBlocking·p0.99:   955
.253 ms/op
                  multiplyParallelBlocking·p0.999:  955
.253 ms/op
                  multiplyParallelBlocking·p0.9999: 955
```

```
.253 ms/op
                 multiplyParallelBlocking·p1.00:   955
.253 ms/op



Result "com.matrix.MultiplicationBenchmark.
multiplyParallelBlocking":
  N = 20
  mean =    950.849 ±(99.9%) 12.960 ms/op

  Histogram, ms/op:
    [910.000, 915.000) = 1
    [915.000, 920.000) = 0
    [920.000, 925.000) = 0
    [925.000, 930.000) = 2
    [930.000, 935.000) = 0
    [935.000, 940.000) = 0
    [940.000, 945.000) = 1
    [945.000, 950.000) = 3
    [950.000, 955.000) = 4
    [955.000, 960.000) = 4
    [960.000, 965.000) = 2
    [965.000, 970.000) = 2
    [970.000, 975.000) = 1

  Percentiles, ms/op:
      p(0.0000) =    910.164 ms/op
     p(50.0000) =    953.680 ms/op
     p(90.0000) =    968.570 ms/op
     p(95.0000) =    971.873 ms/op
     p(99.0000) =    972.030 ms/op
     p(99.9000) =    972.030 ms/op
     p(99.9900) =    972.030 ms/op
     p(99.9990) =    972.030 ms/op
     p(99.9999) =    972.030 ms/op
    p(100.0000) =    972.030 ms/op


# Run complete. Total time: 00:01:01
```

```
REMEMBER: The numbers below are just data. To gain
reusable insights, you need to follow up on
why the numbers are the way they are. Use profilers (
see -prof, -lprof), design factorial
experiments, perform baseline and negative tests that
provide experimental control, make sure
the benchmarking environment is safe on JVM/OS/HW
level, ask for reviews from the domain experts.
Do not assume the numbers tell you what you want them
to tell.

Benchmark

                     (n)    Mode  Cnt    Score
Error  Units
MultiplicationBenchmark.multiplyParallelBlocking
                                     1000  sample  209
   48.859 ±  0.732  ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.00    1000  sample
46.268           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.50    1000  sample
48.103           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.90    1000  sample
51.249           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.95    1000  sample
53.543           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.99    1000  sample
65.235           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.999   1000  sample
75.891           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.9999  1000  sample
75.891           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p1.00    1000  sample
```

```
75.891           ms/op
MultiplicationBenchmark.multiplyParallelBlocking
                                    1500  sample   30
357.250 ±  5.056  ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.00    1500  sample
349.700           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.50    1500  sample
354.681           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.90    1500  sample
367.421           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.95    1500  sample
379.663           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.99    1500  sample
381.682           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.999   1500  sample
381.682           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.9999  1500  sample
381.682           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p1.00    1500  sample
381.682           ms/op
MultiplicationBenchmark.multiplyParallelBlocking
                                    2000  sample   20
950.849 ± 12.960  ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.00    2000  sample
910.164           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.50    2000  sample
953.680           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.90    2000  sample
968.570           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
```

```
multiplyParallelBlocking·p0.95    2000  sample
971.873           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.99    2000  sample
972.030           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.999   2000  sample
972.030           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p0.9999  2000  sample
972.030           ms/op
MultiplicationBenchmark.multiplyParallelBlocking:
multiplyParallelBlocking·p1.00    2000  sample
972.030           ms/op

Process finished with exit code 0
```