REPORT

#problem2_code documents

과목명	멀티코어컴퓨팅 01분반
교수명	손봉수
학번	20206319
학과	소프트웨어학부 소프트웨어전공
이름	김가연

(a) What environment (e.g. CPU type, number of cores, memory size, OS type) the experimentation was performed.

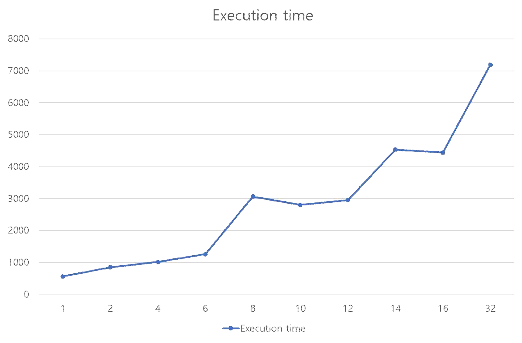
The CPU type of this machine is Intel Core i5-5200U.
It has two cores, four logical processors(when hyperthreading on), and clock speed is 2.20 GHz.
The memory size is 8.0 GB and is DDR3. The speed of the memory is 1600 MHz.
The OS type is 64-bit operating system, x64 based processor.
Experiments were conducted in the same environment as above. Unfortunately, it did not perform on PCs with more than the recommended 4 cores.

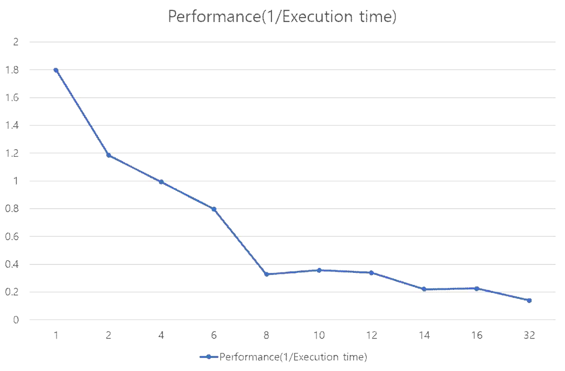
(b) Tables and graphs that show the execution time (unit: ms) for the number of entire threads = {1,2,4,6,8,10,12,14,16,32}.

*The unit of execution time is million seconds.
*The performance is expressed up to 3 decimal places.

	1	2	4	6	8	10	12	14	16	32
exec time	556	844	1008	1255	3062	2801	2950	4531	4439	7122

	1	2	4	6	8	10	12	14	16	32
performace (1/exec time)	1.799	1.185	0.992	0.797	0.327	0.357	0.339	0.221	0.225	0.140





(c) Explanation/analysis on the results and why such results are obtained with sufficient details.

As a result of my code, the execution time increases as the number of threads increases. Performance, which is in inverse proportion to execution time, is on the decline. The reason for this result was considered as follows.

1. The cost of context switching increases with more threads. This is due to the additional overhead required by the CPU to switch from one thread to another.

2. In multi-threaded programming, contention for shared resources can occur. Using synchronization to control this introduces latency between threads, which can cause performance degradation.

3. A cross-thread dependency arises when the data generated by the

previous thread is used by the next thread. Because of this, continuity must be maintained between threads, which can increase overall execution time due to the latency introduced when threads execute.

(d) Entire JAVA source code and explain my code, screen capture image of program execution and output, briefly explain how to compile and execute the source code I submit.

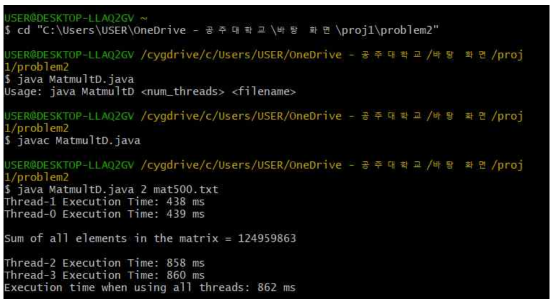
This is how to compile and execute the source. First, turn on the cygwin terminal. After that, move to the directory path where the java file is located. After moving, enter the following command to compile.

$javac java_filename.java

Enter the following command to run it.

$java java_filename.java num_thread txt_filename.txt

Below is an example screen.



```
USER@DESKTOP-LLAQ2GV ~
$ cd "C:\Users\USER\OneDrive - 공주대학교\바탕 화면\proj1\problem2"

USER@DESKTOP-LLAQ2GV /cygdrive/c/Users/USER/OneDrive - 공주대학교/바탕 화면/proj
1/problem2
$ java MatmultD.java
Usage: java MatmultD <num_threads> <filename>

USER@DESKTOP-LLAQ2GV /cygdrive/c/Users/USER/OneDrive - 공주대학교/바탕 화면/proj
1/problem2
$ javac MatmultD.java

USER@DESKTOP-LLAQ2GV /cygdrive/c/Users/USER/OneDrive - 공주대학교/바탕 화면/proj
1/problem2
$ java MatmultD.java 2 mat500.txt
Thread-1 Execution Time: 438 ms
Thread-0 Execution Time: 439 ms

Sum of all elements in the matrix = 124959863

Thread-2 Execution Time: 858 ms
Thread-3 Execution Time: 860 ms
Execution time when using all threads: 862 ms
```

#MatmultD.java source code

```
package problem2;

import java.util.*;
import java.io.*;

public class MatmultD implements Runnable {
private static Scanner sc = new Scanner(System.in);
private static int[][] A, B, C;
private int start, end;

public MatmultD(int start, int end) {
this.start = start;
this.end = end;
}

public static void main(String[] args) {
int thread_id = 0;
String filename = "";
if (args.length == 2) {
thread_id = Integer.parseInt(args[0]);
filename = args[1];
} else {
thread_id = 1;
System.out.println("Usage: java MatmultD <num_threads> <filename>");
System.exit(1);
}

A = readMatrix(filename);
B = readMatrix(filename);

int a = A.length, b = B[0].length;
```

```java
C = new int[a][b];
int size = a / thread_id, start = 0, end = 0;
Thread[] MultiThread = new Thread[thread_id];
long startTime = System.currentTimeMillis();

for (int i = 0; i < thread_id; i++) {
start = end;
end = start + size;
if (i == thread_id - 1) {
end = a;
}
MultiThread[i] = new Thread(new MatmultD(start, end));
MultiThread[i].start();
}
long endTime = System.currentTimeMillis();

for (int i = 0; i < thread_id; i++) {
try {
MultiThread[i].join();
} catch (Exception e) {
e.printStackTrace();
}
}
printMatrix(C);

if (thread_id > 1) {
A = readMatrix(filename);
B = readMatrix(filename);

a = A.length;
b = B[0].length;
C = new int[a][b];
size = a;
start = 0;
end = a;
Thread[] AllThreads = new Thread[thread_id];
startTime = System.currentTimeMillis();

for (int i = 0; i < thread_id; i++) {
AllThreads[i] = new Thread(new MatmultD(start, end));
AllThreads[i].start();
}

for (int i = 0; i < thread_id; i++) {
try {
AllThreads[i].join();
} catch (Exception e) {
e.printStackTrace();
}
}
endTime = System.currentTimeMillis();
System.out.printf("Execution time when using all threads: %d ms\n", endTime - startTime);
}
}

public static int[][] readMatrix(String filename) {
try {
Scanner fileScanner = new Scanner(new File(filename));
int rows = fileScanner.nextInt();
int cols = fileScanner.nextInt();
int[][] arr = new int[rows][cols];
for (int i = 0; i < rows; i++) {
for (int j = 0; j < cols; j++) {
arr[i][j] = fileScanner.nextInt();
}
}
fileScanner.close();
return arr;
} catch (FileNotFoundException e) {
e.printStackTrace();
System.exit(1);
return null;
}
}

public static void printMatrix(int[][] mt) {
int rows = mt.length, cols = mt[0].length, sum = 0;
for (int i = 0; i < rows; i++) {
for (int j = 0; j < cols; j++) {
sum += mt[i][j];
}
}
System.out.println();
System.out.println("Sum of all elements in the matrix = " + sum + "\n");
}
@Override
public void run() {
int a = A[0].length;
long startTime = System.currentTimeMillis();
for (int i = start; i < end; i++) {
for (int j = 0; j < a; j++) {
for (int k = 0; k < a; k++) {
C[i][j] += A[i][k] * B[k][j];
}
}
}
long endTime = System.currentTimeMillis();
System.out.printf("%s Execution Time: %d ms\n", Thread.currentThread().getName(),
endTime - startTime);
}
}
```

1. Main method

- As suggested in the document, Java is executed in the cygwin terminal, and it receives the number of threads (num_threads) and filename (filename) among the commands written in the command line as arguments.
- If num_threads is greater than 1, num_threads number of threads are created.
- Read the two matrices A and B in the file loaded from filename.
- Stores the result of multiplying matrix C by matrix A and B.
- Creates threads according to -num_threads and transfers the start and end values of each thread.
- Start each thread, wait until all threads are finished, and measure the time (startTime, endTime).
- Print C
- All threads are terminated and the total execution time is output.

2. readMatrix method

- When the function is called, it accepts filename as a parameter.
- Use the Scanner class to read the number of rows and columns in a file.
- Based on the number of rows and columns read, create a two dimensional array (arr) of that size.
- Read data from a file and save it to arr and returns an array.
- If the file does not exist, a FileNotFoundException is thrown and the program terminates.

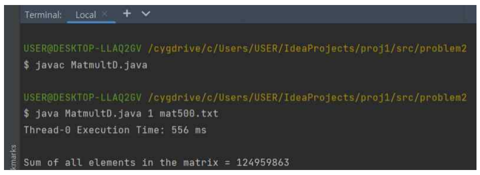
3. printMatrix method

-Calculate the sum of all elements of the passed matrix, and print this value.

<u>4. The run method</u>

- Store the cols of A in a and the current time in startTime. Each element of A and B is multiplied and accumulated in C while executing a loop in the range of rows between start and end.
- After the operation is finished, the current time is stored in endTime, and the name of the current thread and the execution time of the thread are output.

case 1)1 thread

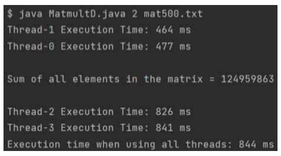


```
USER@DESKTOP-LLAQ2GV /cygdrive/c/Users/USER/IdeaProjects/proj1/src/problem2
$ javac MatmultD.java

USER@DESKTOP-LLAQ2GV /cygdrive/c/Users/USER/IdeaProjects/proj1/src/problem2
$ java MatmultD.java 1 mat500.txt
Thread-0 Execution Time: 556 ms

Sum of all elements in the matrix = 124959863
```

case 2)2 threads

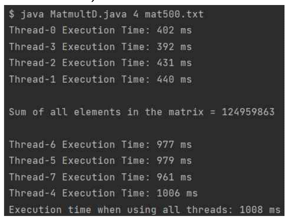


```
$ java MatmultD.java 2 mat500.txt
Thread-1 Execution Time: 464 ms
Thread-0 Execution Time: 477 ms

Sum of all elements in the matrix = 124959863

Thread-2 Execution Time: 826 ms
Thread-3 Execution Time: 841 ms
Execution time when using all threads: 844 ms
```

case 3)4 threads

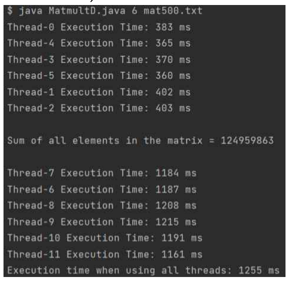


```
$ java MatmultD.java 4 mat500.txt
Thread-0 Execution Time: 402 ms
Thread-3 Execution Time: 392 ms
Thread-2 Execution Time: 431 ms
Thread-1 Execution Time: 440 ms

Sum of all elements in the matrix = 124959863

Thread-6 Execution Time: 977 ms
Thread-5 Execution Time: 979 ms
Thread-7 Execution Time: 961 ms
Thread-4 Execution Time: 1006 ms
Execution time when using all threads: 1008 ms
```

case 4)6 threads

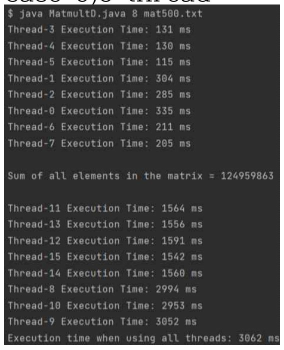


```
$ java MatmultD.java 6 mat500.txt
Thread-0 Execution Time: 383 ms
Thread-4 Execution Time: 365 ms
Thread-3 Execution Time: 370 ms
Thread-5 Execution Time: 360 ms
Thread-1 Execution Time: 402 ms
Thread-2 Execution Time: 403 ms

Sum of all elements in the matrix = 124959863

Thread-7 Execution Time: 1184 ms
Thread-6 Execution Time: 1187 ms
Thread-8 Execution Time: 1208 ms
Thread-9 Execution Time: 1215 ms
Thread-10 Execution Time: 1191 ms
Thread-11 Execution Time: 1161 ms
Execution time when using all threads: 1255 ms
```

case 5)8 thread

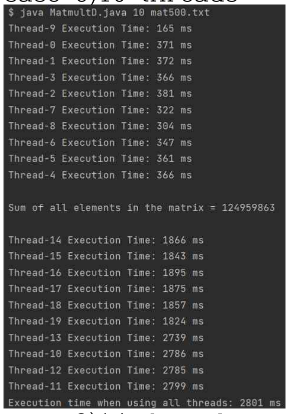


```
$ java MatmultD.java 8 mat500.txt
Thread-3 Execution Time: 131 ms
Thread-4 Execution Time: 130 ms
Thread-5 Execution Time: 115 ms
Thread-1 Execution Time: 304 ms
Thread-2 Execution Time: 285 ms
Thread-0 Execution Time: 335 ms
Thread-6 Execution Time: 211 ms
Thread-7 Execution Time: 205 ms

Sum of all elements in the matrix = 124959863

Thread-11 Execution Time: 1564 ms
Thread-13 Execution Time: 1556 ms
Thread-12 Execution Time: 1591 ms
Thread-15 Execution Time: 1542 ms
Thread-14 Execution Time: 1560 ms
Thread-8 Execution Time: 2994 ms
Thread-10 Execution Time: 2953 ms
Thread-9 Execution Time: 3052 ms
Execution time when using all threads: 3062 ms
```

case 6)10 threads

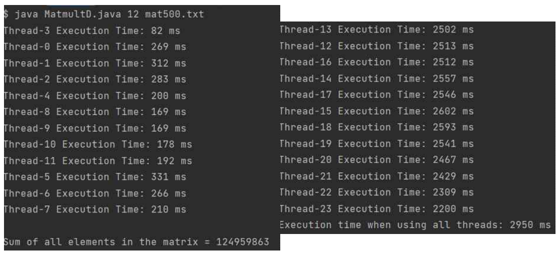


```
$ java MatmultD.java 10 mat500.txt
Thread-9 Execution Time: 165 ms
Thread-0 Execution Time: 371 ms
Thread-1 Execution Time: 372 ms
Thread-3 Execution Time: 366 ms
Thread-2 Execution Time: 381 ms
Thread-7 Execution Time: 322 ms
Thread-8 Execution Time: 304 ms
Thread-6 Execution Time: 347 ms
Thread-5 Execution Time: 361 ms
Thread-4 Execution Time: 366 ms

Sum of all elements in the matrix = 124959863

Thread-14 Execution Time: 1866 ms
Thread-15 Execution Time: 1843 ms
Thread-16 Execution Time: 1895 ms
Thread-17 Execution Time: 1875 ms
Thread-18 Execution Time: 1857 ms
Thread-19 Execution Time: 1824 ms
Thread-13 Execution Time: 2739 ms
Thread-10 Execution Time: 2786 ms
Thread-12 Execution Time: 2785 ms
Thread-11 Execution Time: 2799 ms
Execution time when using all threads: 2801 ms
```

case 7)12 thread

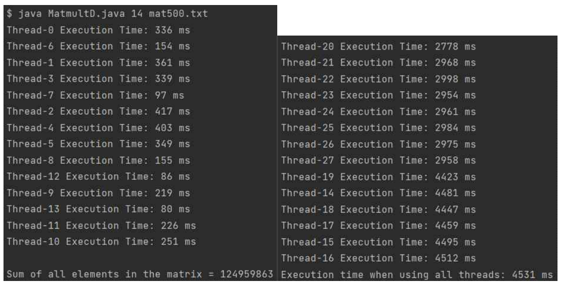


```
$ java MatmultD.java 12 mat500.txt
Thread-3 Execution Time: 82 ms
Thread-0 Execution Time: 269 ms
Thread-1 Execution Time: 312 ms
Thread-2 Execution Time: 283 ms
Thread-4 Execution Time: 200 ms
Thread-8 Execution Time: 169 ms
Thread-9 Execution Time: 169 ms
Thread-10 Execution Time: 178 ms
Thread-11 Execution Time: 192 ms
Thread-5 Execution Time: 331 ms
Thread-6 Execution Time: 266 ms
Thread-7 Execution Time: 210 ms

Sum of all elements in the matrix = 124959863
```

```
Thread-13 Execution Time: 2502 ms
Thread-12 Execution Time: 2513 ms
Thread-16 Execution Time: 2512 ms
Thread-14 Execution Time: 2557 ms
Thread-17 Execution Time: 2546 ms
Thread-15 Execution Time: 2602 ms
Thread-18 Execution Time: 2593 ms
Thread-19 Execution Time: 2541 ms
Thread-20 Execution Time: 2467 ms
Thread-21 Execution Time: 2429 ms
Thread-22 Execution Time: 2309 ms
Thread-23 Execution Time: 2200 ms
Execution time when using all threads: 2950 ms
```

case 8)14 threads

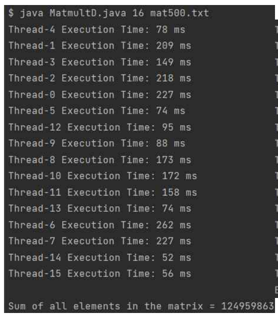


```
$ java MatmultD.java 14 mat500.txt
Thread-0 Execution Time: 336 ms
Thread-6 Execution Time: 154 ms
Thread-1 Execution Time: 361 ms
Thread-3 Execution Time: 339 ms
Thread-7 Execution Time: 97 ms
Thread-2 Execution Time: 417 ms
Thread-4 Execution Time: 403 ms
Thread-5 Execution Time: 349 ms
Thread-8 Execution Time: 155 ms
Thread-12 Execution Time: 86 ms
Thread-9 Execution Time: 219 ms
Thread-13 Execution Time: 80 ms
Thread-11 Execution Time: 226 ms
Thread-10 Execution Time: 251 ms

Sum of all elements in the matrix = 124959863
```

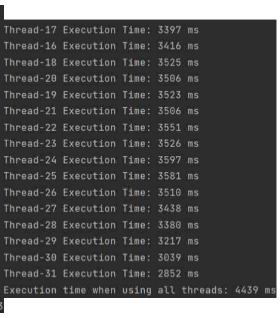
```
Thread-20 Execution Time: 2778 ms
Thread-21 Execution Time: 2968 ms
Thread-22 Execution Time: 2998 ms
Thread-23 Execution Time: 2954 ms
Thread-24 Execution Time: 2961 ms
Thread-25 Execution Time: 2984 ms
Thread-26 Execution Time: 2975 ms
Thread-27 Execution Time: 2958 ms
Thread-19 Execution Time: 4423 ms
Thread-14 Execution Time: 4481 ms
Thread-18 Execution Time: 4447 ms
Thread-17 Execution Time: 4459 ms
Thread-15 Execution Time: 4495 ms
Thread-16 Execution Time: 4512 ms
Execution time when using all threads: 4531 ms
```

case 9)16 thread



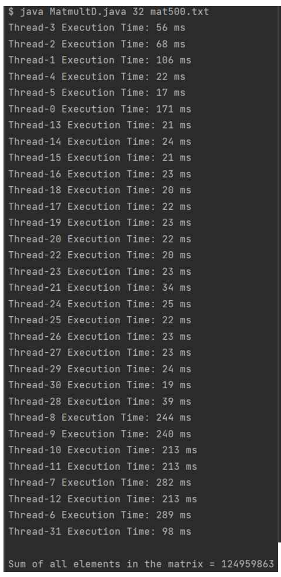
```
$ java MatmultD.java 16 mat500.txt
Thread-4 Execution Time: 78 ms
Thread-1 Execution Time: 209 ms
Thread-3 Execution Time: 149 ms
Thread-2 Execution Time: 218 ms
Thread-0 Execution Time: 227 ms
Thread-5 Execution Time: 74 ms
Thread-12 Execution Time: 95 ms
Thread-9 Execution Time: 88 ms
Thread-8 Execution Time: 173 ms
Thread-10 Execution Time: 172 ms
Thread-11 Execution Time: 158 ms
Thread-13 Execution Time: 74 ms
Thread-6 Execution Time: 262 ms
Thread-7 Execution Time: 227 ms
Thread-14 Execution Time: 52 ms
Thread-15 Execution Time: 56 ms

Sum of all elements in the matrix = 124959863
```



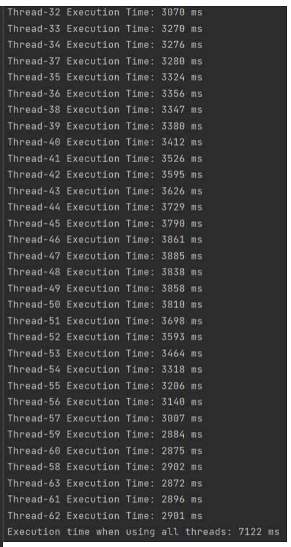
```
Thread-17 Execution Time: 3397 ms
Thread-16 Execution Time: 3416 ms
Thread-18 Execution Time: 3525 ms
Thread-20 Execution Time: 3506 ms
Thread-19 Execution Time: 3523 ms
Thread-21 Execution Time: 3506 ms
Thread-22 Execution Time: 3551 ms
Thread-23 Execution Time: 3526 ms
Thread-24 Execution Time: 3597 ms
Thread-25 Execution Time: 3581 ms
Thread-26 Execution Time: 3510 ms
Thread-27 Execution Time: 3438 ms
Thread-28 Execution Time: 3380 ms
Thread-29 Execution Time: 3217 ms
Thread-30 Execution Time: 3039 ms
Thread-31 Execution Time: 2852 ms
Execution time when using all threads: 4439 ms
```

case 10)32 threads



```
$ java MatmultD.java 32 mat500.txt
Thread-3 Execution Time: 56 ms
Thread-2 Execution Time: 68 ms
Thread-1 Execution Time: 106 ms
Thread-4 Execution Time: 22 ms
Thread-5 Execution Time: 17 ms
Thread-0 Execution Time: 171 ms
Thread-13 Execution Time: 21 ms
Thread-14 Execution Time: 24 ms
Thread-15 Execution Time: 21 ms
Thread-16 Execution Time: 23 ms
Thread-18 Execution Time: 20 ms
Thread-17 Execution Time: 22 ms
Thread-19 Execution Time: 23 ms
Thread-20 Execution Time: 22 ms
Thread-22 Execution Time: 20 ms
Thread-23 Execution Time: 23 ms
Thread-21 Execution Time: 34 ms
Thread-24 Execution Time: 25 ms
Thread-25 Execution Time: 22 ms
Thread-26 Execution Time: 23 ms
Thread-27 Execution Time: 23 ms
Thread-29 Execution Time: 24 ms
Thread-30 Execution Time: 19 ms
Thread-28 Execution Time: 39 ms
Thread-8 Execution Time: 244 ms
Thread-9 Execution Time: 240 ms
Thread-10 Execution Time: 213 ms
Thread-11 Execution Time: 213 ms
Thread-7 Execution Time: 282 ms
Thread-12 Execution Time: 213 ms
Thread-6 Execution Time: 289 ms
Thread-31 Execution Time: 98 ms

Sum of all elements in the matrix = 124959863
```



```
Thread-32 Execution Time: 3070 ms
Thread-33 Execution Time: 3270 ms
Thread-34 Execution Time: 3276 ms
Thread-37 Execution Time: 3280 ms
Thread-35 Execution Time: 3324 ms
Thread-36 Execution Time: 3356 ms
Thread-38 Execution Time: 3347 ms
Thread-39 Execution Time: 3380 ms
Thread-40 Execution Time: 3412 ms
Thread-41 Execution Time: 3526 ms
Thread-42 Execution Time: 3595 ms
Thread-43 Execution Time: 3626 ms
Thread-44 Execution Time: 3729 ms
Thread-45 Execution Time: 3790 ms
Thread-46 Execution Time: 3861 ms
Thread-47 Execution Time: 3885 ms
Thread-48 Execution Time: 3838 ms
Thread-49 Execution Time: 3858 ms
Thread-50 Execution Time: 3810 ms
Thread-51 Execution Time: 3698 ms
Thread-52 Execution Time: 3593 ms
Thread-53 Execution Time: 3464 ms
Thread-54 Execution Time: 3318 ms
Thread-55 Execution Time: 3206 ms
Thread-56 Execution Time: 3140 ms
Thread-57 Execution Time: 3007 ms
Thread-59 Execution Time: 2884 ms
Thread-60 Execution Time: 2875 ms
Thread-58 Execution Time: 2902 ms
Thread-63 Execution Time: 2872 ms
Thread-61 Execution Time: 2896 ms
Thread-62 Execution Time: 2901 ms
Execution time when using all threads: 7122 ms
```