

Assignment2 report

1. Design

This project includes 4 classes. Then I will introduce these 4 classes and explain how to realize each requirements by implementing this project.

1. Class Client

Client class simulates as a client that provides a IU interaction interface guiding the user to use this system, connects to the Server, transfers the user's information to the Server and receives the Server's respond. In this case, the user input a sequence of natural number in the Client and he can also choose a kind of calculation including $2x$, x^2 , \sqrt{x} , and $\log x$ and a kind of Server.

```
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDE/
-----
|Input a sequence of natural numbers which divided by space.|
-----
1 2 3 4 5
-----
|Input operation you want.|
|      1. 2^x      |
|      2. x^2      |
|      3. sqrt(x)   |
|      4. log(x)    |
-----
1
-----
|          Choose a mode you want.          |
| 1.uniformly dividing the sequence S across the threads|
| 2.one thread for each one of these numbers |
|and 1 thread for all the remaining numbers|
-----
1
Final result is [2, 4, 8, 16, 32] Final time is 8337milis

Process finished with exit code 0
|
```

2. Class Server

Server class simulates as a Server that receives the input from the Client, processes the input and transfers the respond back to the Client. In this case, the Server functions as a calculator which provides 4 kinds of calculations. This server uses multi-Thread to process the input data to increase efficiency. And here we provide two kinds of multi-Thread approaches to make a comparison of each's efficiency.

```
Connect successfully
Start time is 1648822848989
DataGathering1 is implement
mode: 1 Condition: k>length
case1 is working
Operated number: 2
case1 is working
Operated number: 4
Attention!!!! final Result[2, 4]
End time is 1648822852610
exetime is 3621
Attention!!!!!!! final time 3621milis
final result is [2, 4] final time is 3621
```

3. Class calculateThread

I aim to do the calculations in multi-Thread. So I create this class as a thread. Importantly, I use the object Lock to realize synchronized. This class provides 4 kinds of calculations.

```
public class calculateThread extends Thread{}
```

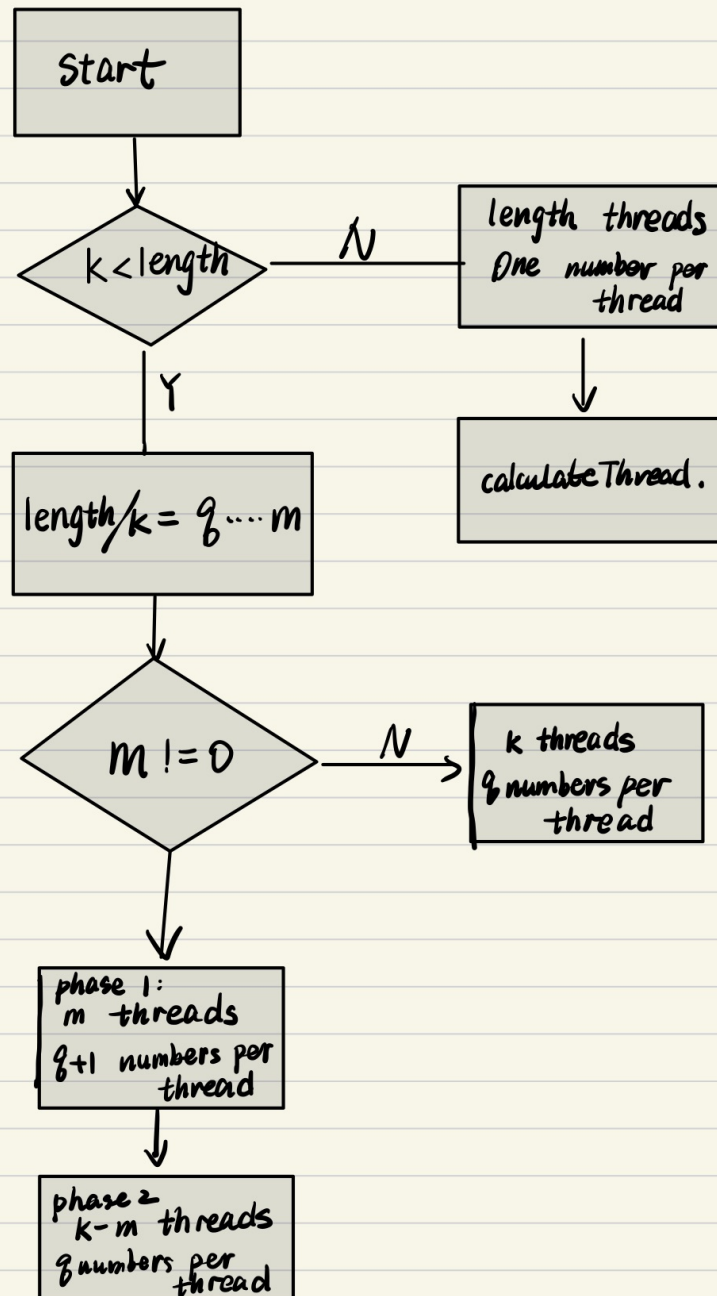
4. Class DataGathering

This class provides two kinds of Servers. (a)One version will be uniformly dividing the sequence S across the threads. (b)One version will be using $k - 1$ threads for the first $k - 1$ numbers of the sequence (one thread for each one of these numbers) and 1 thread for all the remaining numbers of S.

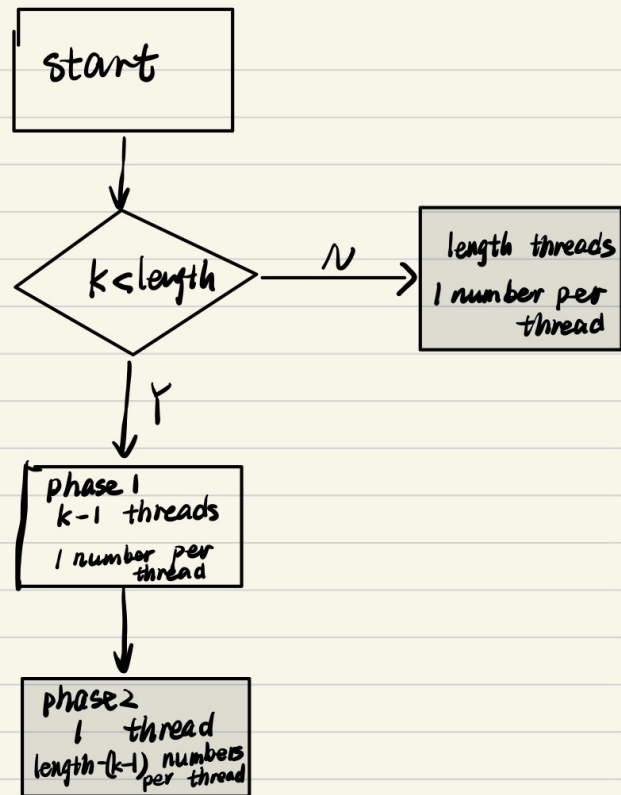
We input a sequence of numbers in the format of an array and a number which represents the calculation we choose.

k - number of threads (pre-set)
length - length of natural number sequence

Version 1:



Version 2



5. Socket Timeout

I use `socket.setTimeout(int timeout)` method to set a pre-determined time T.

```
Socket socket = new Socket(hostName, portNumber);  
socket.setTimeout(5000);
```

```

Client.java x Server.java x calculateThread.java x DataGathering.java x
14
15 }
16
17 public static void main(String[] args) throws IOException {
18
19
20     if (args.length != 2){
21         System.err.println("Usage: java HelloClient <hostname> <port numebr>");
22         System.exit( status: 1);
23     }
24
25     String hostName = args[0];
26     int portNumber = Integer.parseInt(args[1]);
27
28
29
30     try (
31         Socket myClientSocket = new Socket(hostName,portNumber);
32         PrintWriter output = new PrintWriter(myClientSocket.getOutputStream(), autoFlush: true);
33         BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in)); //from user
34         BufferedReader input = new BufferedReader(new InputStreamReader(myClientSocket.getInputStream()));
35     ) {
36
37         myClientSocket.setSoTimeout(5000);
38
39         int len = 10000000;
40         int max = 10;
41         int[] arr = generateArray(len, max);
42
43         String sequence;
44         // System.out.println("-----");
45         // System.out.println("|Input a sequence of natural numbers which divided by space.|");
46         // System.out.println("-----");
47         // sequence = stdIn.readLine();
48
49         sequence = Arrays.toString(arr);
50         sequence = sequence.substring(1, sequence.length()-1);

```

2. Testing

(1) When the user input a sequence of numbers, and choose the operation and Server mode, the Serve will do the calculations and transefer the operated sequence and operation time to the client.

```

Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=
|Input a sequence of natural numbers which divided by space.|
-----
1 2 3 4 5 6 7 8 9 0
-----
|Input operation you want.|
| 1. 2^x |
| 2. x^2 |
| 3. sqrt(x) |
| 4. log(x) |
-----
2
-----
| Choose a mode you want. |
| 1.uniformly dividing the sequence S across the threads|
| 2.one thread for each one of these numbers |
| and 1 thread for all the remaining numbers|
-----
1
Final result is [1, 4, 9, 16, 25, 36, 49, 64, 81, 0] Final time is 11672milis

Process finished with exit code 0

```

```
Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54310:/Applicat
Connect successfully
Start time is 1648827007318
DataGathering1 is implement
mode: 1Condition: k<length
q = 5 m = 0
Condition: mod = 0
case2 is working
case2 is working
Attention!!!! final Result[1, 4, 9, 16, 25, 36, 49, 64, 81, 0]
End time is 1648827018990
exetime is 11672
Attention!!!!!! final time 11672millis
final result is [1, 4, 9, 16, 25, 36, 49, 64, 81, 0] final time is 11672
|
```

(2)To compare the efficiency difference of two versions of Server, I wrote a method to generate a sequence randomly (the size of the sequence can be pre-set).

In case 1: k = 2

a) Set the size of the sequence to 1000

Final execution time of Server 1 is:

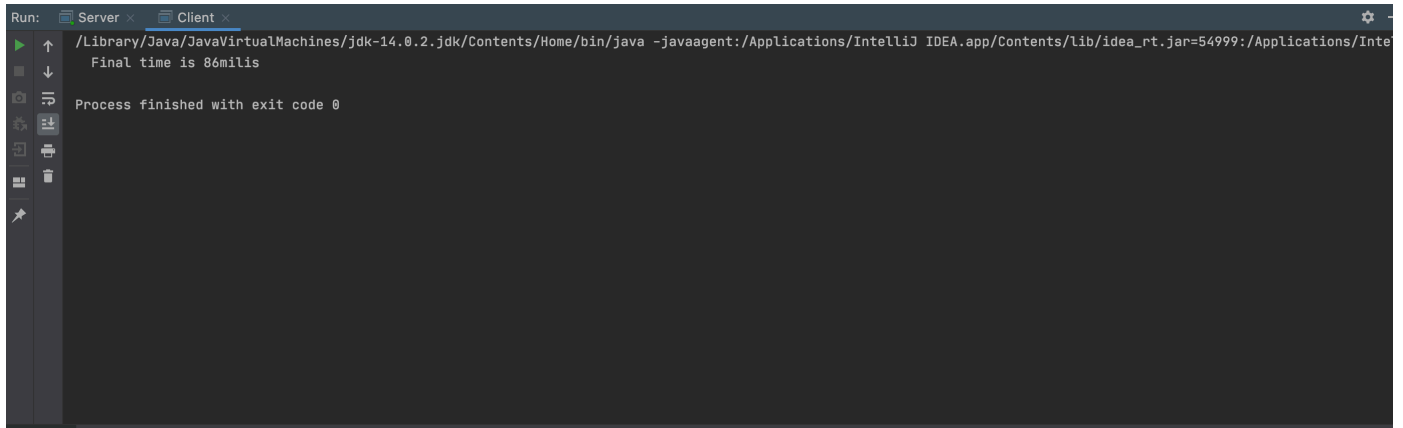
```
Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54980:
Final time is 11milis
Process finished with exit code 0
```

Final execution time of Server 2 is:

```
Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54988:/Applications/Int
Final time is 7milis
Process finished with exit code 0
```

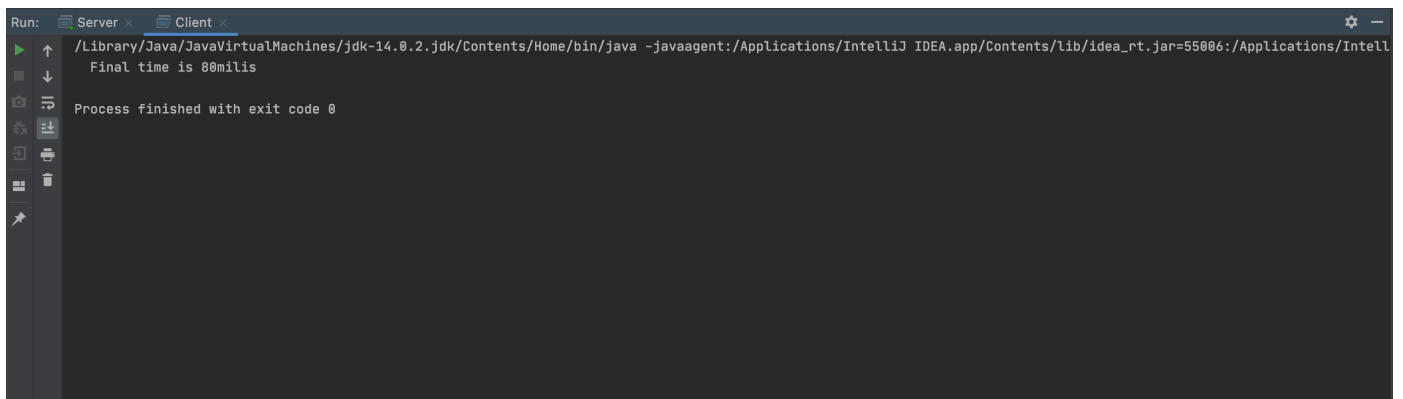
b)Set the size of the sequence to 100000

Final execution time of Server 1 is:



The screenshot shows the Run console in IntelliJ IDEA with two tabs: 'Server' and 'Client'. The 'Server' tab is active, displaying the command: `/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=54999:/Applications/IntelliJ IDEA.app/bin/java`. Below the command, the output is 'Final time is 86milis'. At the bottom, it states 'Process finished with exit code 0'. The left sidebar shows the standard IDE toolbars.

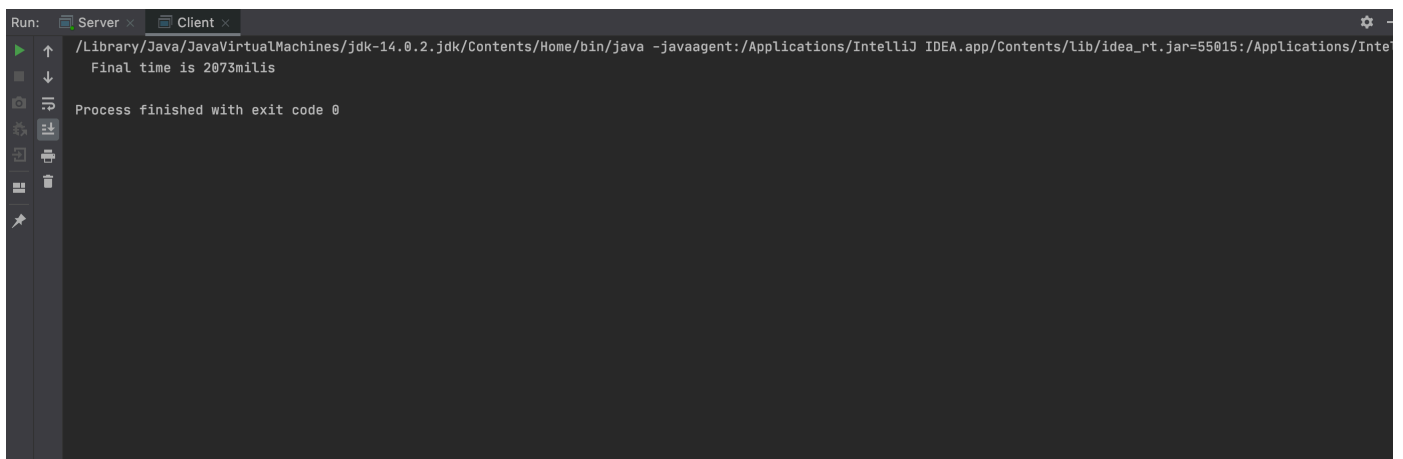
Final execution time of Server 2 is:



The screenshot shows the Run console in IntelliJ IDEA with two tabs: 'Server' and 'Client'. The 'Server' tab is active, displaying the command: `/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55006:/Applications/IntelliJ IDEA.app/bin/java`. Below the command, the output is 'Final time is 80milis'. At the bottom, it states 'Process finished with exit code 0'. The left sidebar shows the standard IDE toolbars.

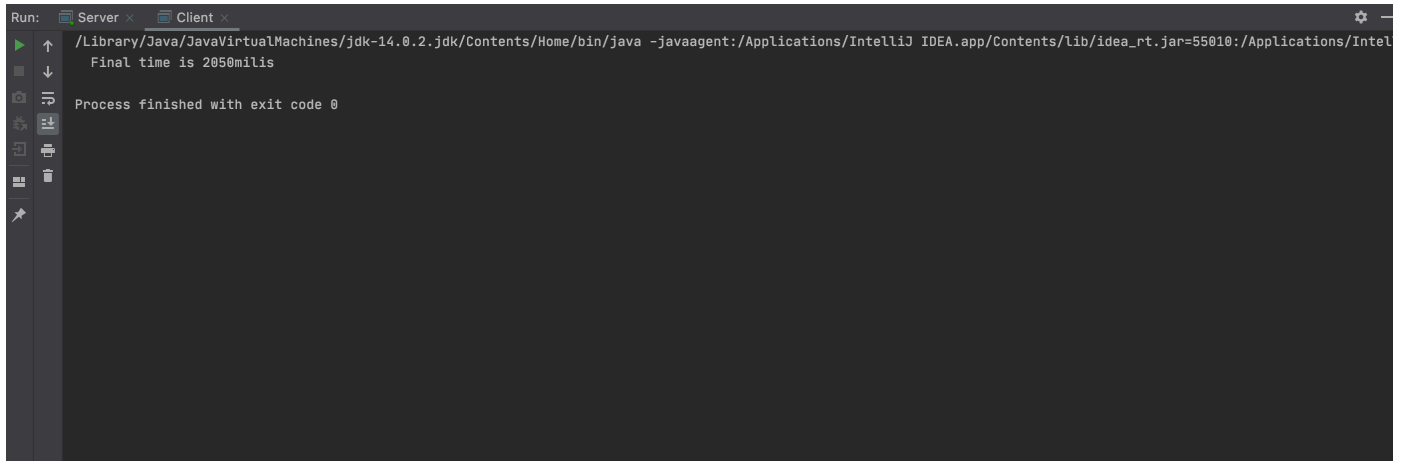
c)Set the size of the sequence to 10000000

Final execution time of Server 1 is:



The screenshot shows the Run console in IntelliJ IDEA with two tabs: 'Server' and 'Client'. The 'Server' tab is active, displaying the command: `/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55015:/Applications/IntelliJ IDEA.app/bin/java`. Below the command, the output is 'Final time is 2073milis'. At the bottom, it states 'Process finished with exit code 0'. The left sidebar shows the standard IDE toolbars.

Final execution time of Server 2 is:

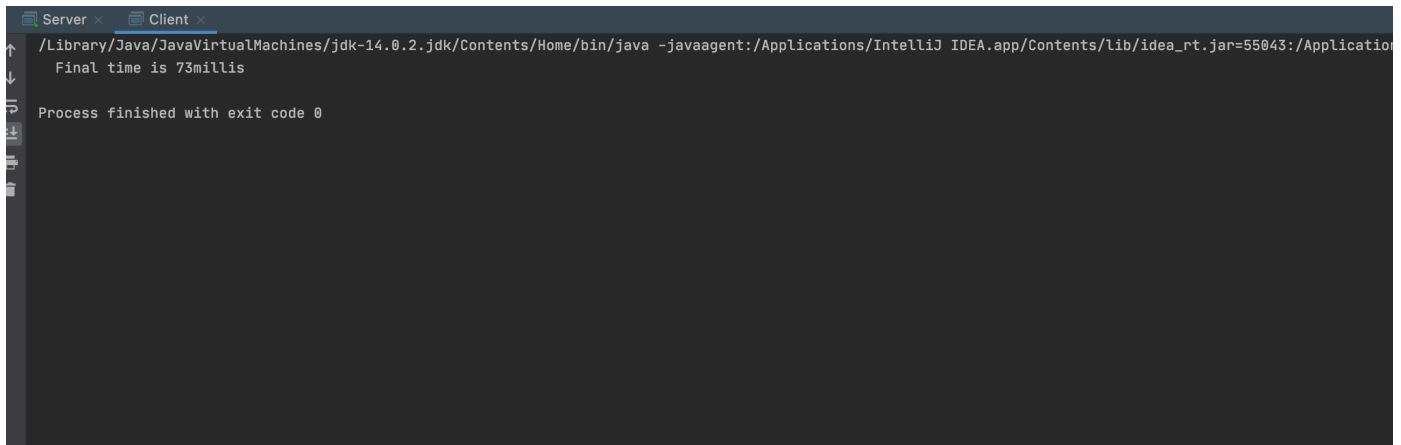


```
Run: Server Client
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55010:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar:
Final time is 2050milis
Process finished with exit code 0
```

In case 2: $k = 50$

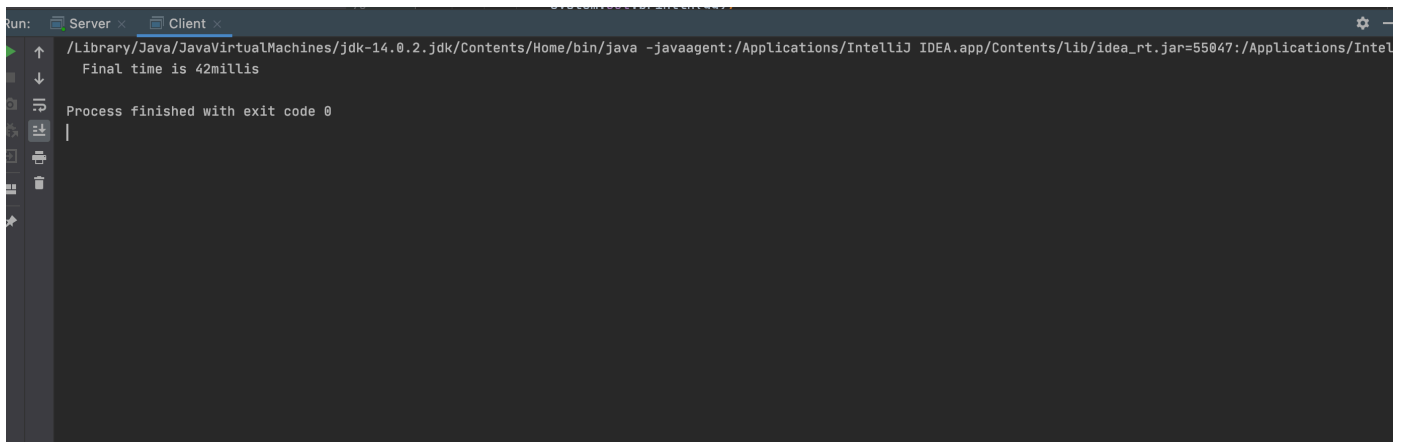
a) Set the size of the sequence to 1000

Final execution time of Server 1 is:



```
Run: Server Client
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55043:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar:
Final time is 73millis
Process finished with exit code 0
```

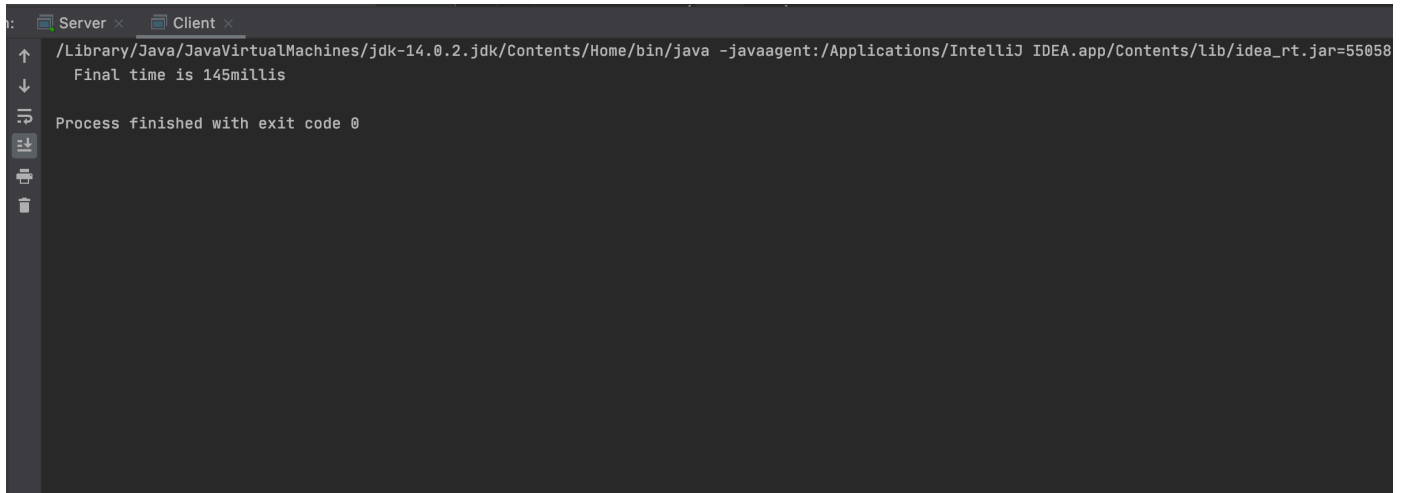
Final execution time of Server 2 is:



```
Run: Server Client
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55047:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar:
Final time is 42millis
Process finished with exit code 0
```

b) Set the size of the sequence to 100000

Final execution time of Server 1 is:



```
Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55058
Final time is 145millis
Process finished with exit code 0
```

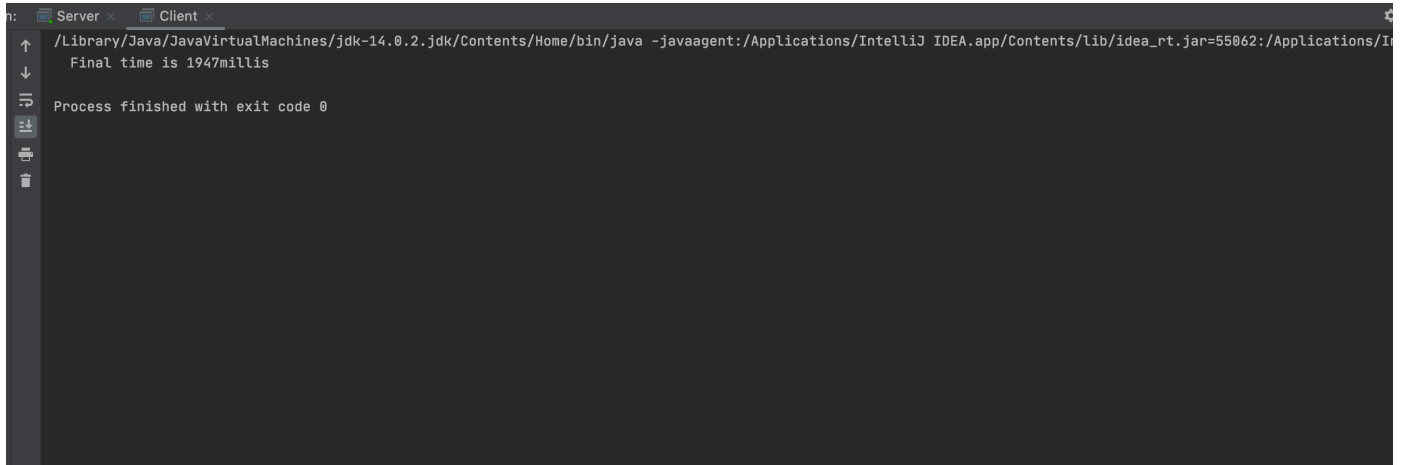
Final execution time of Server 2 is:



```
Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_
Final time is 198millis
Process finished with exit code 0
```

c) Set the size of the sequence to 10000000

Final execution time of Server 1 is:



```
Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=55062:/Applications/I
Final time is 1947millis
Process finished with exit code 0
```

Final execution time of Server 2 is:

```
Server x Client x
/Library/Java/JavaVirtualMachines/jdk-14.0.2.jdk/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt
Final time is 2044millis
Process finished with exit code 0
```

This table shows the execution time (Server1 time\Server2 time)

(Server 1\2)(millis)	1000	100000	10000000
k = 2	11\7	86\80	2073\2050
k = 50	73\42	145\198	1947\2044

3. Conclusion

When number of thread $k = 2$, Server 2 is always faster than Server 1.

When number of thread $k = 50$, the execution time is longer than that when $k = 2$ when the size of sequence is small(1000). However, when the size of sequence is growing larger and larger, the execution time of $k = 50$ will be shorter than that of $k = 2$ gradually.

Multi-thread will increase execution speed than single thread. It can be concluded that more threads are not always better. When there is not a lot of data to process, fewer threads are more efficient. When there is a lot of data to be processed, the more threads, the more efficient it is. Also, when there is not a lot of data, processor 1 takes more time than processor 2 because processor 1 requires more steps to split the sequence. But when there is a lot of data, processor 1 is obviously faster than processor 2 because the time to process the data is mainly determined by the thread that processes the most data.