SOFE4790 Distributed Systems Assignment 1 - Luke Hruda - 100654666

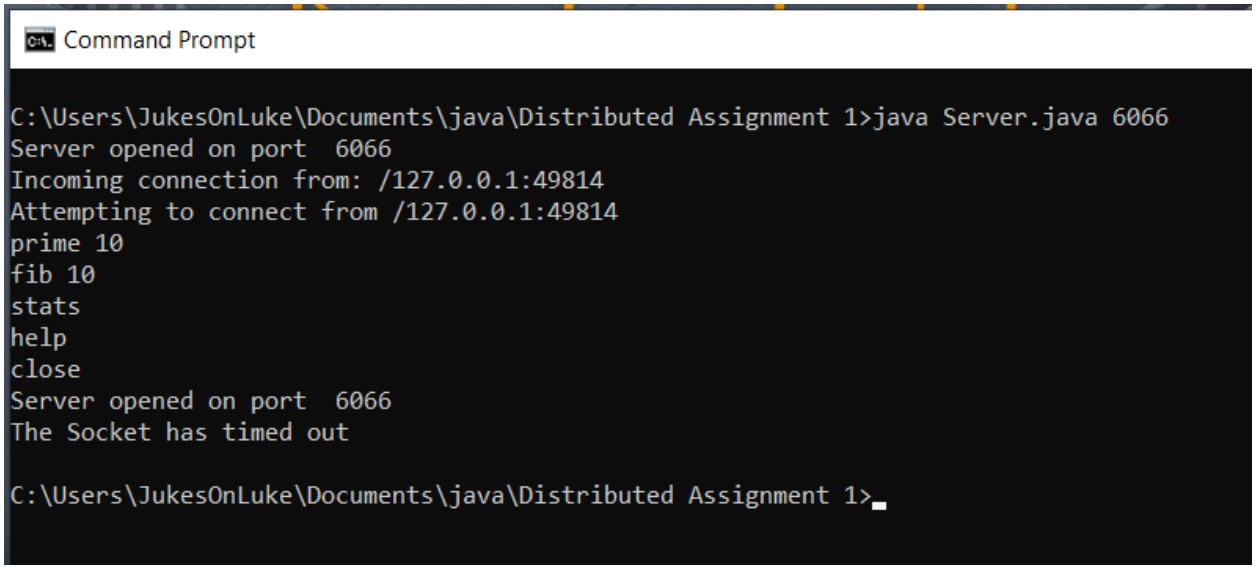
https://github.com/UOITEngineering2/assignment1fall2020-LukeHruda

This report will cover the services and novel functionality of my client server architecture that I am dubbing the number sequence generator. The program is capable of generating two separate sequences of numbers, the fibonacci sequence to a given value and the sequence of prime numbers up to a given number. The fibonacci sequence works when the client calls fib <n>, the server calculates the fibonacci sequence to the n^{th} value in the sequence. So if the client calls fib 10, the server outputs 1 1 2 3 5 8 13 21 34 55, which is the first ten values of the fibonacci set. The prime number sequence works in that it prints all the numbers from two (as one is not a prime number) to the given value that are prime numbers. So if the client calls prime 10, the server outputs 2 3 5 7, which are all the prime numbers from two to ten. There are two main novel features that are included with the server, the first being a time taken calculation, where after a sequence is requested, the server also returns to the client how long it took said sequence to process in seconds. So when a client requests prime 10, in addition to returning 2 3 5 7, it will also return "Operation completed in: 0.007s" based on how long it took. Additionally, the server features a "stats" command in which the server will print to the client how many other clients have connected in the past, how many requests have been processed and how long the server has been running for. Other notable features include the help command, which provides the list of commands to the client, methodized error checking to ensure that the commands are entered properly, a server time out of 1 minute if there are no clients connected to the server and is able to promptly disconnect from one client and connect to another waiting.

The first issue I had was within the prime number sequence generator, in which it was constantly outputting no values as if there were no prime numbers. I was using a boolean flag at the time, setting the flag to true if the value had a factor, making it not a prime number. The issue was I forgot that a prime number is divisible by 1 and itself. So I reworked my method to account for this. I am embarrassed to admit how long this stumped me for. Another issue I had, is in implementing the time taken feature. I was trying to write two lines to the client and have the client read both of them independently. This would cause interesting issues, where the client would read the first message twice, then on the next request would read the time taken from the first request instead of the appropriate response. This persisted despite having an equal number of write and read calls in the client and server. To solve this problem, I ended up coding in more string formatting on the server side, so that only one read and write calls were required, rather than try to fix the timing issue. I initially did try to include a 0.1 second pause after the first string was read from the client, hoping that would allow the second line to come through but this did not fix the issue.

Screenshots of the system working.

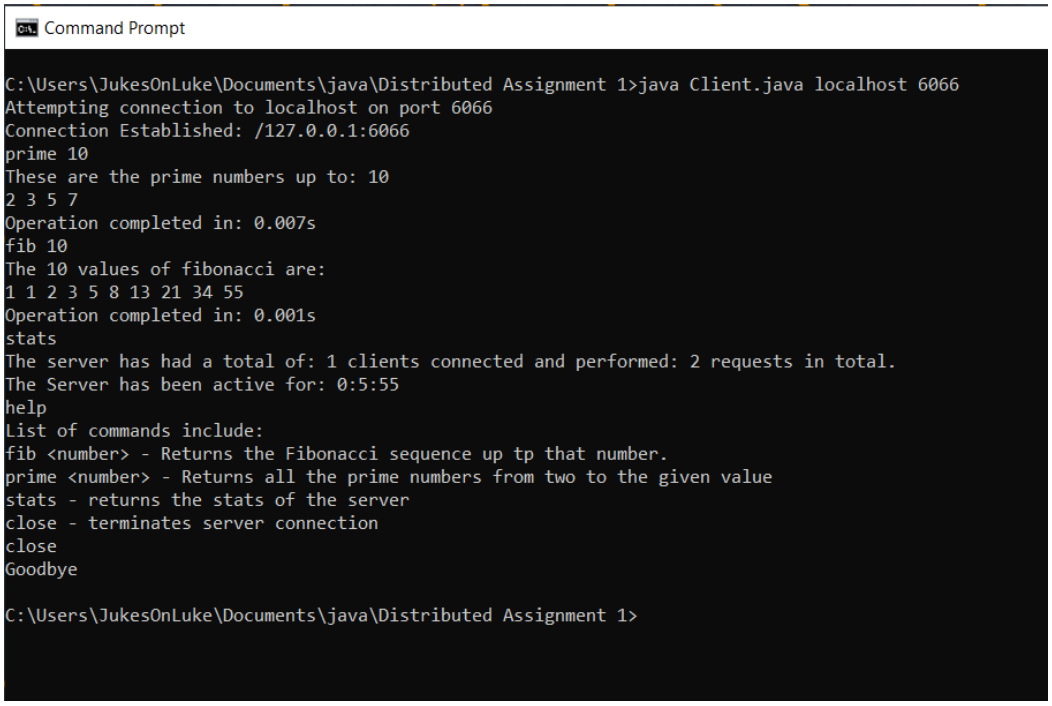
Server:



Line breakdown

1. Call java Server.java <portNumber>
2. Server acknowledges incoming connection.
3. Server attempts to accept connection.
4. Server receives prime 10 request from client
5. Server receives fib 10 request from client
6. Server receives stats requests from client
7. Server receives help request from client
8. Server receives close request from client
9. Server reopens for new client
10. After 1 minute, the server times out.

Client:



Breakdown:

1. Client calls java Client.java <address> <portNum> in this case local host and 6066
2. Client attempts to connect and confirms connection
3. User enters prime 10
4. Client receives the result:

These are the prime numbers up to: 10

2 3 5 7

Operation completed in: 0.007s

5. Client calls fib 10
6. Client receives the result

The 10 values of fibonacci are:

1 1 2 3 5 8 13 21 34 55

Operation completed in: 0.001s

7. Client calls stats and receives the statistics
8. Client calls help and receives the list of commands
9. Client calls close and receives goodbye message from server