Multithreaded client-server socket programming

DNP lab 2 @ Innopolis University, Fall 2022

Task

Create a server and client application using `TCP sockets` and `multithreading`.

1. The client sends the numbers to the server one by one.
2. The server checks if the number is prime or not and sends the result to the client.
3. The client prints the result.

Client

1. Client has a list of numbers to process.
2. Client takes a number, sends it to the server and prints the result.
3. Has a command line argument:
    - `ip-addr`:`port-number` of the server
      Example: `python3 client.py 127.0.0.1:5555`

- If the server is not available at the start of the client, or at any time during data transmission, terminate the client.

Connection must be implemented via `TCP` sockets.

Numbers to process:

```
numbers = [15492781, 15492787, 15492803,
           15492811, 15492810, 15492833,
           15492859, 15502547, 15520301,
           15527509, 15522343, 1550784]
```

Server

1. Main thread creates several worker threads.
2. After that, the main thread continuously listens for connections from clients. The server must be able to stop, when the `KeyboardInterrupt` (Ctrl+C in console) is raised.
3. When a new connection is accepted, main thread passes it to one of worker thread.
4. Worker thread continues the work with the client.
5. When the server stops, all worker threads must be stopped.
6. The server must be able to process several client connections at the same time (at least 2).
7. Has a command line argument:
    - `port-number` to listen on.
      Example: `python3 server.py 5555`

You should use `Thread` from the `threading` module and `Queue` from the `multiprocessing` module.

- When the main thread accepts a connection, it puts a connection socket and an address in the queue.
- Worker thread continuously checks the queue. If it's not empty, it takes an element (connection information) and processes this connection.

Use this function to check if a number is prime:

```
def is_prime(n):
    if n in (2, 3):
        return True
    if n % 2 == 0:
        return False
    for divisor in range(3, n, 2):
        if n % divisor == 0:
            return False
    return True
```

Example output:

Server

```
> python3 server.py 5555
('127.0.0.1', 49688) connected     // First client
('127.0.0.1', 49690) connected     // Second client
('127.0.0.1', 49688) disconnected
('127.0.0.1', 49690) disconnected
^C                                 // KeyboardInterrupt signal
Shutting down                      // Killing all threads
Done
```

Client

```
> python3 client.py 127.0.0.1:5555
Connected to ('127.0.0.1', 5555)
15492781 is prime
15492787 is prime
15492803 is prime
15492811 is prime
15492810 is not prime
15492833 is prime
15492859 is prime
15502547 is prime
15520301 is prime
15527509 is prime
15522343 is not prime
```

```
1550784 is not prime
Completed
```