Introduction to Computer Networks Homework 1

B07303024 經濟四 李品樺

1. Bonus

- Usage: Under the directory of p1, run the commands.

  ```
  python3 socket_server.py
  python3 socket_client.py bonus
  ```

  Adjust the testcases in the file **p1_testcase_bonus**, and the results are in **b07303024_p1_client_result_bonus.log**

- I implemented 5 functions (fact, power, sqrt, perm, comb), as elaborated below:
  - Factorial: The base cases are x == 1 or x == 0, we return 0. Otherwise, we use the recursive method, to find out the final result of the factorial.

    ```
    def fact(x):
        if x == 1 or x == 0:
            return 1
        else:
            return (x * fact(x-1))
    ```

  - Power: First assign 1 to result, then consider the base (k) and the exponent (n) of the function. If n is greater than 0, we multiply the result by x for n times. If n is smaller than 0, then we divide the result by x for n times.

    ```
    def power(x, n):
        result = 1
        if (n > 0):
            for i in range(n):
                result *= x;
        elif n < 0:
            for i in range(n):
                result /= x
        return result;
    ```

  - Square Root: First we check the boundary condition. Then, we take a guess of the result by iteratively divide the guess by 2. If the guess close enough, we return the result

    ```
    def sqrt(x):
        if x == 0:
            return 0
        if x == 1:
            return 1
        last_guess = x /2.0
        while True:
            guess = (last_guess + x/last_guess)/2
            if abs(guess - last_guess) < 0.0001:
                return guess
            last_guess = guess
    ```

- Permutation & Combination: Permutation and combination can be easily computed by using the previous function `Fact` as the images show.

```
def perm(n, k):
    return fact(n) / fact(n-k)
```

```
def comb(n, k):
    return fact(n) / (fact(k) * fact(n-k))
```

2.

(a)

```
Ready to serve...
('127.0.0.1', 53390)connected
RecvMsg:  GET /index.html HTTP/1.1
Host: 127.0.0.1:4047
Connection: keep-alive
sec-ch-ua: "Microsoft Edge";v="107", "Chromium";v="107", "Not=A?Brand";v="24"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "macOS"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36 Edg/107.0.1
418.24
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,zh-TW;q=0.8,zh;q=0.7,zh-CN;q=0.6


/index.html
```

```
Ready to serve...
('127.0.0.1', 53410)connected
RecvMsg:  GET /helloworld.html HTTP/1.1
Host: 127.0.0.1:4047
Connection: keep-alive
sec-ch-ua: "Microsoft Edge";v="107", "Chromium";v="107", "Not=A?Brand";v="24"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "macOS"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36 Edg/107.0.1
418.24
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1:4047/index.html
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,zh-TW;q=0.8,zh;q=0.7,zh-CN;q=0.6


/helloworld.html
```

```
Ready to serve...
('127.0.0.1', 53411)connected
RecvMsg:  GET /no.html HTTP/1.1
Host: 127.0.0.1:4047
Connection: keep-alive
sec-ch-ua: "Microsoft Edge";v="107", "Chromium";v="107", "Not=A?Brand";v="24"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "macOS"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36 Edg/107.0.1
418.24
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9,zh-TW;q=0.8,zh;q=0.7,zh-CN;q=0.6


/no.html
404 Not Found
```

(b)

- First connect to the local machine with port 4047

```
# Create a socket and bind the socket to the address
# TODO start
HOST = "127.0.0.1"
PORT = 4047
ServerSocket.bind((HOST, PORT))
ServerSocket.listen(0)
# TODO end
```

- Establish the connection using ServerSocket.accept()

```
# TODO start
ConnectionSocket, Addr = ServerSocket.accept()
print(str(Addr) + "connected")
# TODO end
```

- Receive the message and decode it

```
# TODO start
RecvMessage = ConnectionSocket.recv(4096).decode()
print("RecvMsg: ", RecvMessage)
# TODO end
```

- After receiving the file, send the header and content type

```
# TODO start
DataInFile = f.read()
# TODO end

# Send one HTTP header line into socket
# Send HTTP Status to the client
# TODO start
header = b'HTTP/1.1 200 OK\r\n'
ConnectionSocket.send(header)
# TODO end

# Send the Content Type to the client
# TODO start
contentType = b'Content-Type: text/html; charset=UTF-8\n\n'
ConnectionSocket.send(contentType)
# TODO end
```

- If the requested file is not found, then send the status ERROR 404

```
# TODO start
print("404 Not Found")
ConnectionSocket.send(b'HTTP/1.1 404 Not Found\r\n')
ConnectionSocket.send(b'Content-Type: text/html; charset=utf-8\r\n')
ConnectionSocket.send(b'404 Not Found\r\n')
# TODO end

# Close client socket
# TODO start
ConnectionSocket.close()
# TODO end
```

3. Usage: `python3 proxy_server.py 127.0.0.1`

(a)

```
127.0.0.1 8000
Ready to serve...
Received a connection from: ('127.0.0.1', 54190)
Recv Message:  ws.mailtrack.io:8282
Filename:
FileToUse: /
host name is
try to connect to the web_server
connected successfully
GET http:/// HTTP/1.1

get the file successfully
Buffer:  ['HTTP/1.1 404 Not Found\n', 'Content-Type: text/html; charset=utf-8\n', '404 Not Found\n']
Illegal request
Ready to serve...
Received a connection from: ('127.0.0.1', 54192)
Recv Message:  /index.html
Filename: index.html
FileToUse: /index.html
host name is index.html
try to connect to the web_server
connected successfully
GET http:///index.html HTTP/1.1

get the file successfully
Buffer:  ['HTTP/1.1 200 OK\n', 'Content-Type: text/html; charset=UTF-8\n', '\n', '<!DOCTYPE html>\n', '<html lang="en">\n', '    <head>\n', '
        <meta charset="UTF-8" />\n', '        <meta name="viewport" content="width=device-width, initial-scale=1.0" />\n', '        <title>In
dex</title>\n', '    </head>\n', '    <body>\n', '        <h1>B07303024 李品樺</h1>\n', '        <a href="./helloworld.html">Click here to He
llo World Page!</a>\n', '    </body>\n', '</html>\n', '\n']
open the file successfully
Done
```

```
Ready to serve...
Received a connection from: ('127.0.0.1', 54196)
Recv Message:  /
Filename:
FileToUse: /
host name is
try to connect to the web_server
connected successfully
GET http:/// HTTP/1.1

get the file successfully
Buffer:  ['HTTP/1.1 404 Not Found\n', 'Content-Type: text/html; charset=utf-8\n', '404 Not Found\n']
Illegal request
```

(b)

- First connect to the proxy server and its port 8000

```python
# TODO start
# proxy server ip and port
HOST, PORT = sys.argv[1], 8000
print(HOST, PORT)
TCPServerSocket.bind((HOST, PORT))
TCPServerSocket.listen(100)
# TODO end
```

- Second, receive the request from the client

```python
# Strat receiving data from the client
print('Ready to serve...')
# TODO start
TCPClientSocket, Addr = TCPServerSocket.accept()
# TODO end
print('Received a connection from:', Addr)

# Step1: Receive request from the client
# TODO start
RecvMessage = TCPClientSocket.recv(4096).decode()
# TODO end
```

- If we find the requested file in the cache, send the encoded data.

```
# TODO start
for i in range(0, len(DataInFile)):
    TCPClientSocket.send(DataInFile[i].encode())
TCPClientSocket.send("\r\n".encode())
# TODO end
```

- On the other hand, if we do not find the file in cache, we need to connect to the web server

```
# Create a socket on the proxy server
# TODO start
SocketOnProxyServer = socket(AF_INET, SOCK_STREAM)
# TODO end
HostName = Filename.replace("www.", "", 1)
print("host name is " + HostName)
try:
    print("try to connect to the web_server")
    # Connect the socket to the web server port
    # TODO: start
    SocketOnProxyServer.connect(("127.0.0.1", 4047))
    # TODO end
    print("connected successfully")
```

- Finally, we read the response to the buffer, send the HTTP response, and write the cache file

```
# Read the response into buffer
# TODO start
buffer = FileObject.readlines()
print("Buffer: ", buffer)
# TODO end

# Create a new copy in the cache for the requested file
# Also send the response back to client socket and the correspond
TmpFile = open(Filename, "w")
print("open the file successfully")
# TODO start
for i in range(len(buffer)):
    if i <= 1:
        TCPClientSocket.send(buffer[i].encode())
    else:
        TmpFile.write(buffer[i])
TmpFile.close()
print("Done")
# TODO end
```