[Session 2] Network Programming Exercise

Note

You don't need to solve everything, because it is not mandatory and graded, just for practice! We will provide a solution after the real-time class. But, if you have some problem and ask us, then we will give you some help or hints. **Good luck!**

Exercise 1

(1.1) Know yourself

Do you know your IP address? Open your terminal / command prompt. Use **ipconfig** (**ifconfig** for Linux and Mac OS) Can you find successfully?

Exercise 2

The following is a skeleton for client-server socket programming. Of course, you can change the code, if you want.

client.py

```
from socket import *

ip = '255.255.255.255'
port = 65535

my_socket = socket(AF_INET, SOCK_STREAM)
my_socket.connect((ip, port))

while True:
    pass
```

server.py

```
from socket import *

ip = '255.255.255.255'
port = 65535

my_socket = socket(AF_INET, SOCK_STREAM)
my_socket.setsockopt(SOL_SOCKET, SO_REUSEADDR, 1)

my_socket.bind((ip, port))
my_socket.listen()
c_socket, address = my_socket.accept()

while True:
    pass
```

(2.1) Echo

Implement server-client model, with the following instructions.

1. Use loopback(127.0.0.1)
2. After the connection is established, client gets some message from user, by `input()`
3. Then, client sends the message to server.

4. Server receives the message, prints out this. Also, re-send the message to client (echo)
5. Client receives the re-sent message, and print out this.
6. If client send message without content (empty string), then connection is terminated

(2.2) File Downloader

You can load your file as a binary file, so it can be sent / received by using `send() / recv()` method. Implement client-server model that follows the instructions:

- Use loopback.
- From the skeleton, you don't need infinite loop (`while True`), so remove it
- Prepare some text file(.txt) and place it on the project directory
- After the connection is established, server sends the file to client
- Client receives this, and store in the same directory with client, as "downloaded.txt"

(+) You can prepare the file data like this:

```
file = open('Filename', 'rb')
filedata = file.read()
```

Change "filename" for your file, and use "filedata". Also, you can store the **DECODED** filedata like this:

```
open("downloaded.txt", 'w').write(data)
```

Be cautious of the recv's buffer size. Sometimes file is very big...

(2.3) Communicate with others

Repeat (2.1) and (2.2), changing IP / Port part. Run the client and server in different computers. (That's ok to do this with other friend not in the program, as long as Python is installed in his/her computer) This should work if you implemented these correctly.