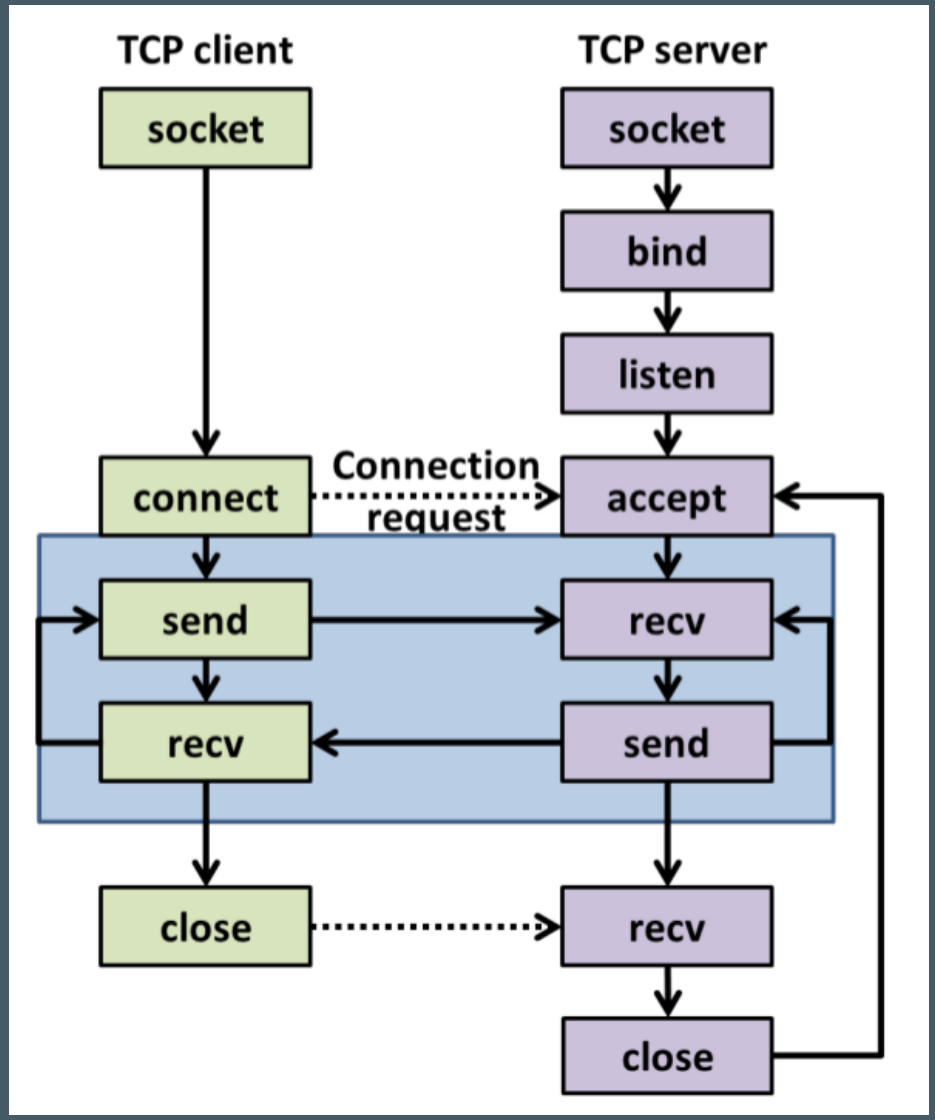
Telecommunications Network

Practice 3

TCP


TCP client
TCP server
socket
socket
bind
listen
connect
Connection request
accept
send
recv
recv
send
close
recv
close

TCP

- socket()

```
import socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

- bind()

```
server_address = ('localhost', 10000)
server.bind(server_address)
```

- listen()

```
server.listen(1)
```

- accept()

```
connection, client_address = sock.accept()
```

TCP

- send(), sendall()

```
connection.sendall(data.encode())
```

- recv()

```
data = connection.recv(16).decode()
```

- close()

```
connection.close()
```

- connect()

```
server_address = (‚localhost’, 10000)
client.connect(server_address)
```

Exercise I.

- Write a simple client-server application where the client writes 'Hello server' to the server and the server responds with 'Hello client'
- Modify the program so that instead of using hardcoded port, we should pass it as a command line argument

Exercise II.

- Write a client-server application where the client sends 2 numbers and a binary operator to the server and the server computes the result of the operation and sends it back to the client. The message of the client should be a struct

Listen - many clients

Write an application where the server's backlog length is 1:

```
server.listen(0)
```

And in the script corresponding to the client, we have 3 clients that all try to connect to the server one after another. Observe what happens!

In Windows we cannot stop the infinite loop server with Ctrl+C but instead with Ctrl+Break. The Break button can change laptop by laptop: for example: Ctrl+Fn+Pause, Ctrl+Fn+B, etc…

Exercise III.

Write a client-server application which plays tic-tac-toe

On the server we have a matrix of size 3x3 :

- It inspects if the game is over or not
- Messages: Win, Loss, Making a move, Wait

Client:

- Connects to the server, accepts server's messages then if it's the starting one:
  - Gets x,y coordinates from std input and sends it to the server
  - Waits for server's answer which has the other player's coordinates
- If it's not the starting one:
  - Waits for the other player's coordinates

Assignment II.

Protocol

- Description

Assignment II.

- Read the binary files given as parameters and write their first lines to the standard output! (The format of the files is going to be different student by student.)
- Write out the different values to the standard output in binary format (the result of pack)! (The values are going to be different student by student) The length of the string is described by the number behind the text!
  Possible struct parameters: f, i, c, ?, Xs (where X is the length of a string, e.g.: 3s)

Submitting: The program should be submitted through the TMS system in .zip format, which contains a single client.py file!
Deadline: See TMS