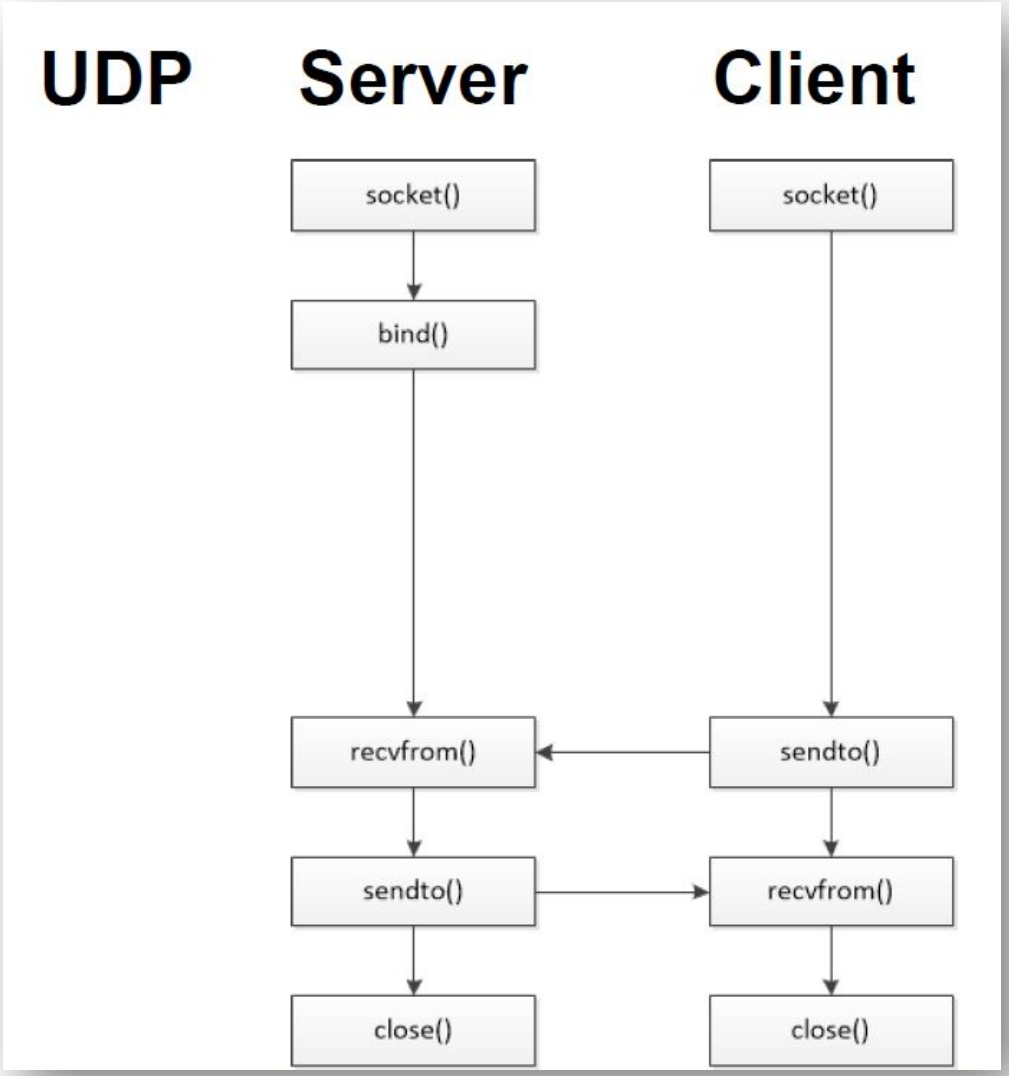
Számítógépes Hálózatok

5. gyakorlat

UDP


UDP
Server
Client
socket()
socket()
bind()
recvfrom()
sendto()
sendto()
recvfrom()
close()
close()

UDP

- socket

```
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

- recvfrom()

```
data, address = sock.recvfrom(4096)
```

- sendto()

```
sent = sock.sendto(data, address)
```

Feladatok

Készítsünk egy kliens-szerver alkalmazást, amely UDP protokollt használ. A kliens küldje a 'Hello Server' üzenetet a szervernek, amely válaszolja a 'Hello Kliens' üzenetet.

Netmask

- Alhálózat címeinek leírása.

Address (Host or Network) **Netmask** (i.e. 24) **Netmask** for sub/supernet (optional)

192.168.0.1 / 16 move to:

Calculate Help

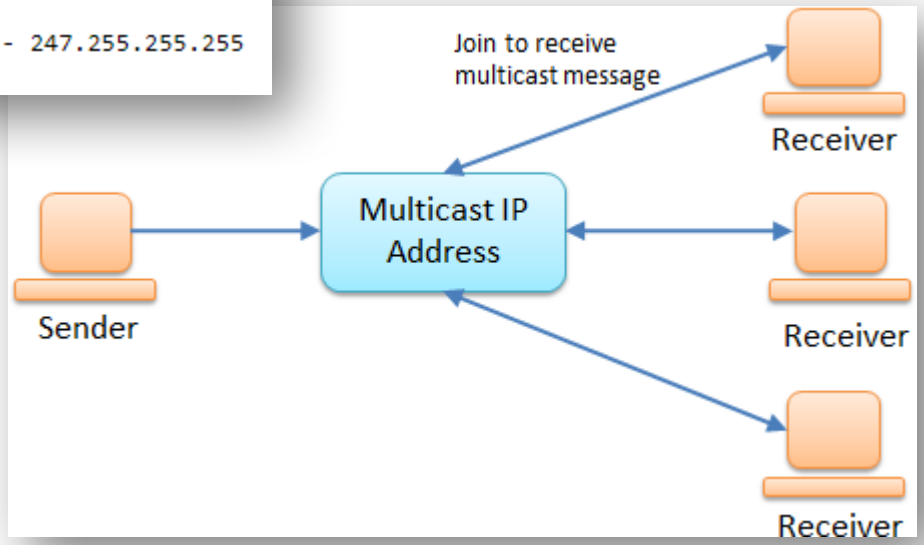
```
Address:   192.168.0.1          11000000.10101000 .00000000.00000001
Netmask:   255.255.0.0 = 16     11111111.11111111 .00000000.00000000
Wildcard:  0.0.255.255          00000000.00000000 .11111111.11111111
=>
Network:   192.168.0.0/16       11000000.10101000 .00000000.00000000 (Class C)
Broadcast: 192.168.255.255      11000000.10101000 .11111111.11111111
HostMin:   192.168.0.1          11000000.10101000 .00000000.00000001
HostMax:   192.168.255.254      11000000.10101000 .11111111.11111110
Hosts/Net: 65534                (Private Internet)
```

Feladat

- Hány cím elérhető a következő netmaskokkal és adjuk meg a minimális és maximális címet:
  - 188.100.22.12/32
  - 188.100.22.12/20
  - 188.100.22.12/10

Multicast

```
Bit -->  0                           31          Address Range:
        +-+--------------------------+
        |0|       Class A Address    |       0.0.0.0 - 127.255.255.255
        +-+--------------------------+
        +-+-+------------------------+
        |1 0|     Class B Address    |     128.0.0.0 - 191.255.255.255
        +-+-+------------------------+
        +-+-+-+----------------------+
        |1 1 0|   Class C Address    |     192.0.0.0 - 223.255.255.255
        +-+-+-+----------------------+
        +-+-+-+-+--------------------+
        |1 1 1 0|  MULTICAST Address |     224.0.0.0 - 239.255.255.255
        +-+-+-+-+--------------------+
        +-+-+-+-+-+------------------+
        |1 1 1 1 0|     Reserved     |     240.0.0.0 - 247.255.255.255
        +-+-+-+-+-+------------------+
```



Multicast

- setsockopt() (sender)

```
ttl = struct.pack('b', 1)
sock.setsockopt(socket.IPPROTO_IP, socket.IP_MULTICAST_TTL, ttl)
```

- socket hozzávétele a multicast grouphoz (recv)

```
multicast_group = '224.3.29.71'
group = socket.inet_aton(multicast_group)
mreq = struct.pack('4sL', group, socket.INADDR_ANY)
sock.setsockopt(socket.IPPROTO_IP, socket.IP_ADD_MEMBERSHIP, mreq)
```

Udp stream példa

Példa kód a gyakorlat honlapján.

cv2 install:

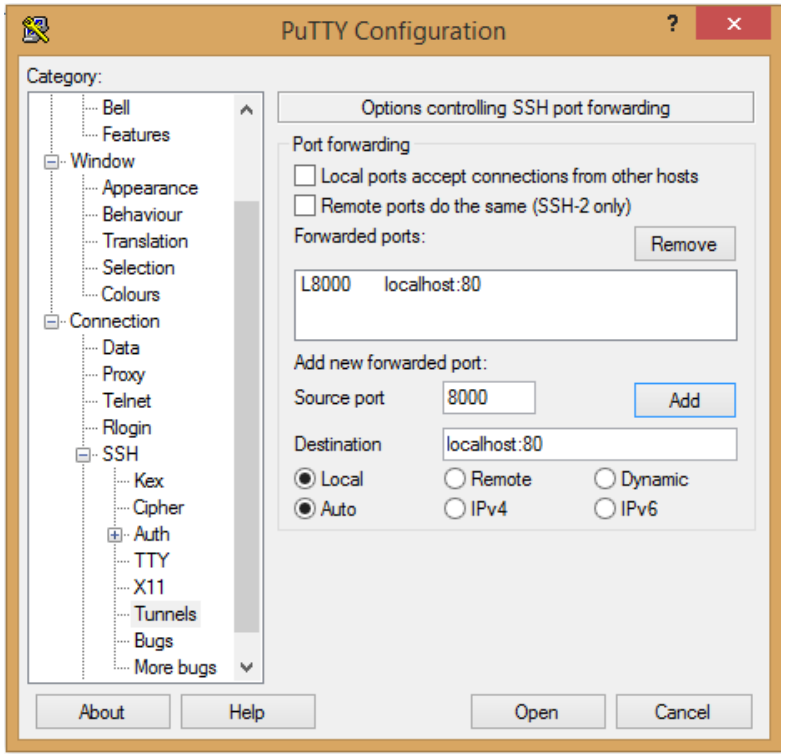
```
py –m pip install --user opencv-python
```

Videó stream működése

- https://medium.com/canal-tech/how-video-streaming-works-on-the-web-an-introduction-7919739f7e1

SSH Tunnel

Windows (putty)



Linux

```
ssh –L 8000:localhost:80 user@hostname
```

Feladatok

1. Készítsünk egy szerver-kliens alkalmazást, ahol a kliens elküld 2 számot és egy operátort a szervernek, amely kiszámolja és visszaküldi az eredményt. A kliens üzenete legyen struktúra.
2. Küldjünk át egy képet UDP segítségével.
   - 200 byte-onként küldjünk
   - Ha vége a filenak akkor küldjünk üres stringet
   - Minden kapott üzenetre OK legyen a válasz

VÉGE