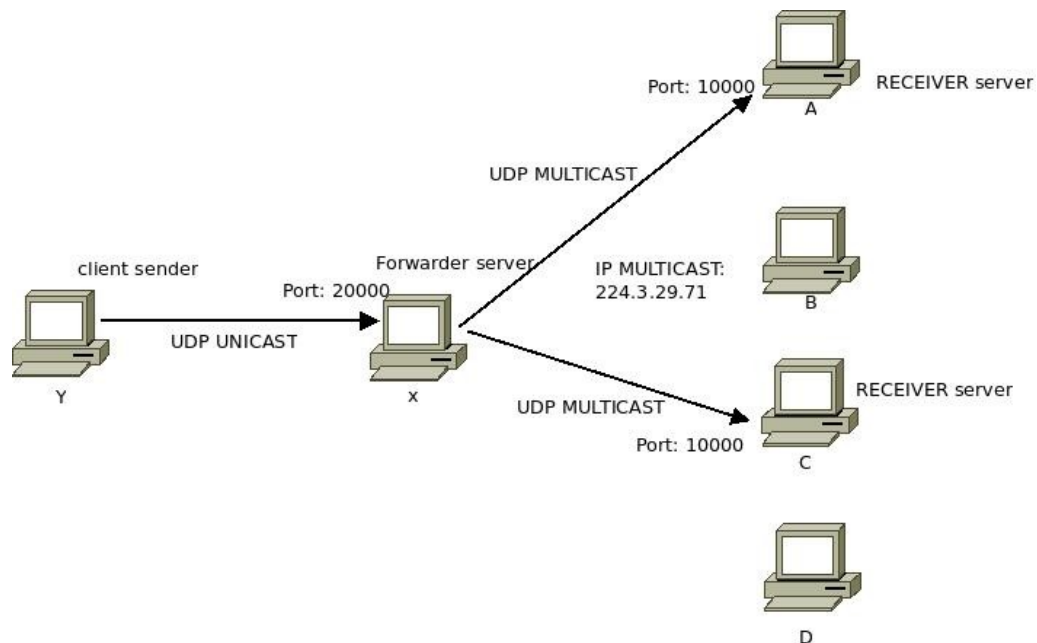


**Muhammad Azhar Rasyad**  
0110217029  
Teknik Informatika  
Pemrograman Sistem dan Jaringan

**Tugas 6 - Forwarder**

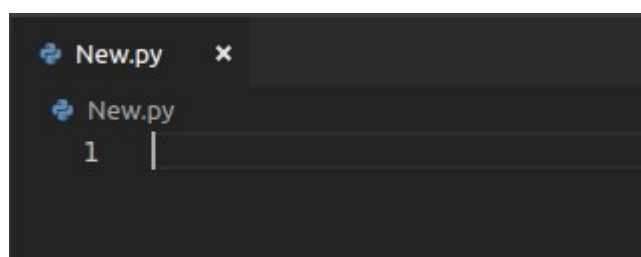


Buatlah kode program untuk mengirimkan pesan dari PC-Y ke PC-X dengan UDP Unicast kemudian PC-X mengirimnya ke PC lain dengan UDP Multicast namun hanya PC yang memiliki IP Multicast yang sama yang dapat menerima pesan dari PC-Y yang telah diforward PC-X

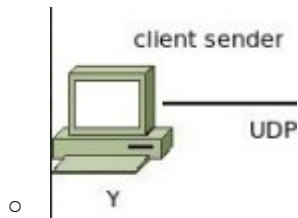
---

Menjawab soal dengan menggunakan alur pengerjaan *step by step*

1. Persiapkan text editor (Saran : Visual Studio Code) dan buat file baru



2. Dari diagram diatas fokus dari *step* awal yaitu fokus pada PC-Y



- PC-Y sebagai **Client UDP Unicast** maka code programnya sebagai berikut

```
from socket import *
import sys
import os

os.system("clear")

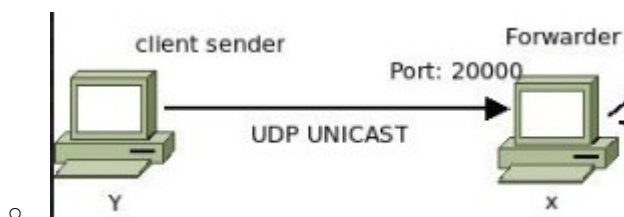
s = socket(AF_INET, SOCK_DGRAM)
print("=====")
print("|Y-PC|")
print("=====")
print()
pesan = input("Ketik Pesan : ")
maxsize = 1024
s.sendto(pesan.encode(), ("127.0.0.1", 20000) )
data, addr = s.recvfrom(maxsize)
print(f"\n{data}")
```

- Tampilannya sebagai berikut



- Selanjutnya ketik pesan dan pastikan ada server yang menyala untuk menerimanya

3. Setelah dari PC-Y maka ikuti garis panah yaitu menunjukkan PC-X



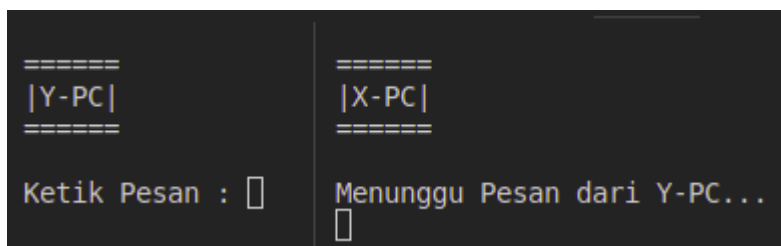
- PC-X sebagai **Server UDP Unicast** maka code programnya sebagai berikut :

```
# Unicast Server
from socket import *
import struct
import sys
import os

s = socket(AF_INET, SOCK_DGRAM)
s.bind(("", 20000))
maxsize = 1024
while True:
    os.system("clear")
    print("=====")
    print("|X-PC|")
    print("=====")
    print("\nMenunggu Pesan dari Y-PC...")
    data, addr = s.recvfrom(maxsize)

    resp = "Pesan Berhasil diKirim ke X-PC"
    s.sendto(resp.encode(), addr)
    print(f"\nPesan dari Y-PC : {data.decode()}")
    input("\nTekan Enter Untuk Forward Pesan...")
```

- Tampilannya sebagai berikut



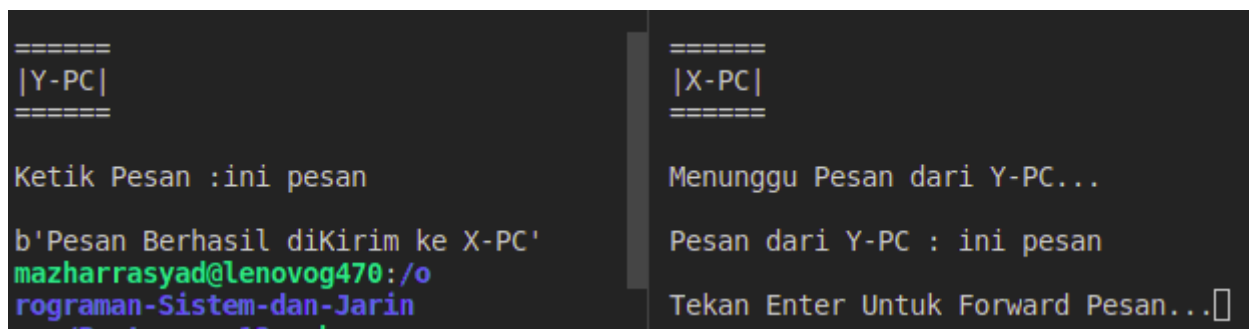
```
=====  
|Y-PC|  
=====
```

```
Ketik Pesan : █
```

```
=====  
|X-PC|  
=====
```

```
Menunggu Pesan dari Y-PC...  
█
```

- Jika dijalankan maka sebagai berikut



```
=====  
|Y-PC|  
=====
```

```
Ketik Pesan :ini pesan
```

```
b'Pesan Berhasil diKirim ke X-PC'
```

```
mazharrasyad@lenovog470:/o
```

```
rograman-Sistem-dan-Jarin
```

```
an/Benteng-12mazharras
```

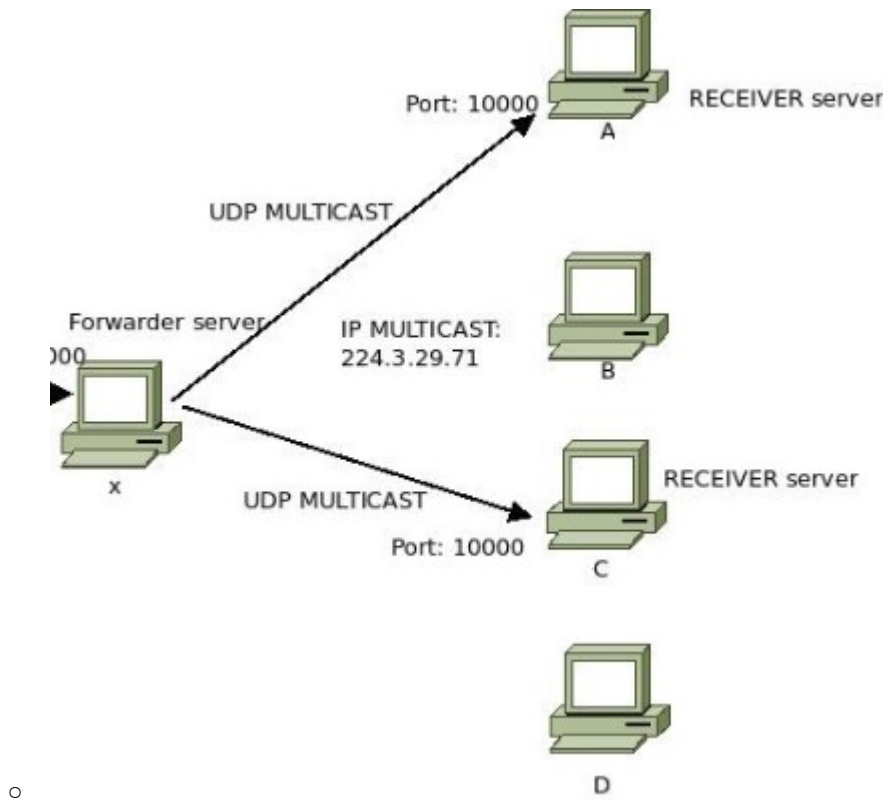
```
=====  
|X-PC|  
=====
```

```
Menunggu Pesan dari Y-PC...
```

```
Pesan dari Y-PC : ini pesan
```

```
Tekan Enter Untuk Forward Pesan...█
```

4. Setelah dari PC-X maka ikuti garis panah yaitu menunjukkan PC-A dan PC-C sementara PC-B dan PC-D tidak termasuk



- PC-X sebagai **Client UDP Multicast** maka code programnya sebagai berikut :

```
# Multicast Client
from socket import *
import struct
import sys
import os

os.system("clear")
print("=====")
print("|X-PC|")
print("=====")
pesan = input('\nKetik Pesan : ')
ip = '224.3.29.71'
port = 10000

multicast_group = (ip, port)

# Create the datagram socket
sock = socket(AF_INET, SOCK_DGRAM)

# Set a timeout so the socket does not block indefinitely when trying
# to receive data.
sock.settimeout(0.2)
```

```
# Set the time-to-live for messages to 1 so they do not go past the
# local network segment.
ttl = struct.pack('b', 3)
sock.setsockopt(IPPROTO_IP, IP_MULTICAST_TTL, ttl)

try:
    # Send data to the multicast group
    sent = sock.sendto(pesan.encode(), multicast_group)

    # Look for responses from all recipients
    while True:
        try:
            data, server = sock.recvfrom(16)
        except timeout:
            break
        else:
            break
finally:
    sock.close()
```

- PC-A-D sebagai **Server UDP Multicast** :

- Code program PC-A :

```
from socket import *
import struct
import sys
import os

multicast_group = '224.3.29.71'
server_address = ('', 10000)

# Create the socket
sock = socket(AF_INET, SOCK_DGRAM)

# Bind to the server address
sock.bind(server_address)
# Tell the operating system to add the socket to the multicast group
# on all interfaces.
group = inet_aton(multicast_group)
mreq = struct.pack('4sL', group, INADDR_ANY)
sock.setsockopt(IPPROTO_IP, IP_ADD_MEMBERSHIP, mreq)
# Receive/respond loop
while True:
    os.system("clear")
    print("=====")
    print("|A-PC|")
    print("=====")
    print('\nMenunggu Pesan dari X-PC...')
    data, address = sock.recvfrom(1024)

    print(f'\nPesan dari X-PC : {data}')
    print(f'Dari IP : {address}')

    sock.sendto(b'ACK', address)
    input("\nTekan Enter Untuk Menerima Pesan Lagi...")
```

- Code program PC-B :

```
from socket import *
import struct
import sys
import os

multicast_group = '224.3.29.71'
server_address = ('', 30000)

# Create the socket
sock = socket(AF_INET, SOCK_DGRAM)

# Bind to the server address
sock.bind(server_address)

# Tell the operating system to add the socket to the multicast group
# on all interfaces.
group = inet_aton(multicast_group)
mreq = struct.pack('4sL', group, INADDR_ANY)
sock.setsockopt(IPPROTO_IP, IP_ADD_MEMBERSHIP, mreq)

# Receive/respond loop
while True:
    os.system("clear")
    print("=====")
    print("|B-PC|")
    print("=====")
    print('\nMenunggu Pesan dari X-PC...')
    data, address = sock.recvfrom(1024)

    print(f'\nPesan dari X-PC : {data}')
    print(f'Dari IP : {address}')

    sock.sendto(b'ACK', address)
    input("\nTekan Enter Untuk Menerima Pesan Lagi...")
```

- Code program PC-C :

```
from socket import *
import struct
import sys
import os

multicast_group = '224.3.29.71'
server_address = ('', 10000)

# Create the socket
sock = socket(AF_INET, SOCK_DGRAM)

# Bind to the server address
sock.bind(server_address)
# Tell the operating system to add the socket to the multicast group
# on all interfaces.
group = inet_aton(multicast_group)
mreq = struct.pack('4sL', group, INADDR_ANY)
sock.setsockopt(IPPROTO_IP, IP_ADD_MEMBERSHIP, mreq)
# Receive/respond loop
while True:
    os.system("clear")
    print("=====")
    print("|C-PC|")
    print("=====")
    print('\nMenunggu Pesan dari X-PC...')
    data, address = sock.recvfrom(1024)

    print(f'\nPesan dari X-PC : {data}')
    print(f'Dari IP : {address}')

    sock.sendto(b'ACK', address)
    input("\nTekan Enter Untuk Menerima Pesan Lagi...")
```



- Code program PC-D :

```
from socket import *
import struct
import sys
import os

multicast_group = '224.3.29.71'
server_address = ('', 30000)

# Create the socket
sock = socket(AF_INET, SOCK_DGRAM)

# Bind to the server address
sock.bind(server_address)
# Tell the operating system to add the socket to the multicast group
# on all interfaces.
group = inet_aton(multicast_group)
mreq = struct.pack('4sL', group, INADDR_ANY)
sock.setsockopt(IPPROTO_IP, IP_ADD_MEMBERSHIP, mreq)
# Receive/respond loop
while True:
    os.system("clear")
    print("=====")
    print("|D-PC|")
    print("=====")
    print('\nMenunggu Pesan dari X-PC...')
    data, address = sock.recvfrom(1024)

    print(f'\nPesan dari X-PC : {data}')
    print(f'Dari IP : {address}')

    sock.sendto(b'ACK', address)
    input("\nTekan Enter Untuk Menerima Pesan Lagi...")
```

- Tampilannya sebagai berikut :

```
=====
|X-PC|
=====

Ketik Pesan : █
```

=====	=====	=====	=====
A-PC	B-PC	C-PC	D-PC
=====	=====	=====	=====
Menunggu Pesan dari X-PC...	Menunggu Pesan dari X-PC...	Menunggu Pesan dari X-PC...	Menunggu Pesan dari X-PC...

- Jika dijalankan maka sebagai berikut :

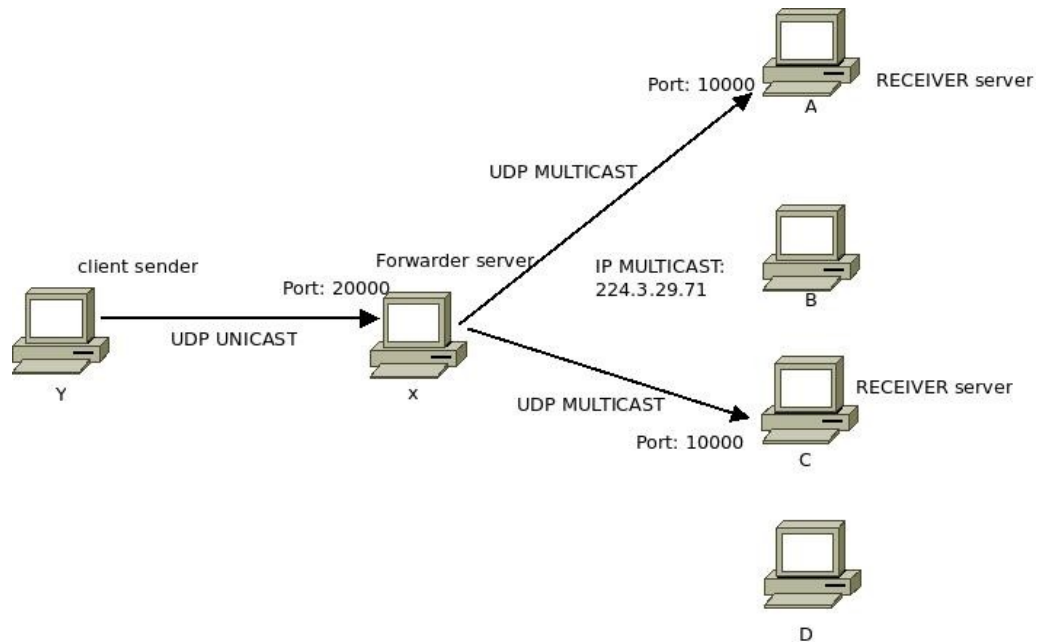
```
=====
|X-PC|
=====

Ketik Pesan : ini pesan
```

=====	=====
A-PC	B-PC
=====	=====
Menunggu Pesan dari X-PC...	Menunggu Pesan dari X-PC...
Pesan dari X-PC : b'ini pesan'	█
Dari IP : ('10.107.141.229', 53915)	
Tekan Enter Untuk Menerima Pesan Lagi... <input type="checkbox"/>	

=====	=====
C-PC	D-PC
=====	=====
Menunggu Pesan dari X-PC...	Menunggu Pesan dari X-PC...
Pesan dari X-PC : b'ini pesan'	<input type="checkbox"/>
Dari IP : ('10.107.141.229', 49256)	
Tekan Enter Untuk Menerima Pesan Lagi... <input type="checkbox"/>	

5. Terakhir hubungkan keseluruhan PC



- o
- o Karena seluruh PC dihubungkan maka perlu adanya penghubung antara PC awal ke PC akhir yaitu PC-X
- o PC-X sebagai perantara menjadi **Client UDP Unicast** dan juga sebagai **Server UDP Multicast**
- o Berikut code program PC-X Client Server :

```
# Unicast Server dan Multicast Client
from socket import *
import struct
import sys
import os

s = socket(AF_INET, SOCK_DGRAM)
s.bind(("", 20000))
maxsize = 1024
while True:
    os.system("clear")
    print("=====")
    print("|X-PC|")
    print("=====")
    print("\nMenunggu Pesan dari Y-PC...")
    data, addr = s.recvfrom(maxsize)

    resp = "Pesan Berhasil diKirim ke X-PC"
    s.sendto(resp.encode(), addr)
    print(f"\nPesan dari Y-PC : {data.decode()}")
    input("\nTekan Enter Untuk Forward Pesan...")
```

```

pesan = data.decode()
ip = '224.3.29.71'
port = 10000

multicast_group = (ip, port)

# Create the datagram socket
sock = socket(AF_INET, SOCK_DGRAM)

# Set a timeout so the socket does not block indefinitely when trying
# to receive data.
sock.settimeout(0.2)

# Set the time-to-live for messages to 1 so they do not go past the
# local network segment.
ttl = struct.pack('b', 3)
sock.setsockopt(IPPROTO_IP, IP_MULTICAST_TTL, ttl)

try:
    # Send data to the multicast group
    print("\n==== Forward =====")
    print(f"Pesan : {pesan}")
    print(f"IP : {ip}")
    print(f"Port : {port}")
    print("=====")
    sent = sock.sendto(pesan.encode(), multicast_group)

    # Look for responses from all recipients
    while True:
        try:
            data, server = sock.recvfrom(16)
        except timeout:
            break
        else:
            break
finally:
    sock.close()
    input("\nTekan Enter Untuk Menerima Pesan Lagi...")

```

6. Berikut hasil tampilan ketika dijalankan

- Saat awal dijalankan

[Y-PC]	[X-PC]	[A-PC]	[B-PC]
Ketik Pesan : █	Menunggu Pesan dari Y-PC...	Menunggu Pesan dari X-PC...	Menunggu Pesan dari X-PC...

- Ketika PC-Y memberikan pesan maka PC-X menerima terlebih dahulu sebelum di forward ke PC lain

[Y-PC]	[X-PC]	[A-PC]	[B-PC]
Ketik Pesan : Yes b'Pesan Berhasil diKirim ke X-PC' mzharrasyad@lenovog470:/opt/Lampp/h	Menunggu Pesan dari Y-PC... Pesan dari Y-PC : Yes Tekan Enter Untuk Forward Pesan...█	Menunggu Pesan dari X-PC... █	Menunggu Pesan dari X-PC... █

- Ketika PC-X sudah menerima maka akan otomatis forward ke PC yang terhubung jaringan multicast yang sama

[Y-PC]	[X-PC]	[A-PC]	[B-PC]
Ketik Pesan : Yes b'Pesan Berhasil diKirim ke X-PC' mzharrasyad@lenovog470:/opt/Lampp/h Jaringan/Pertemuan 13mzharrasyad@l enovog470:/opt/Lampp/htdocs/Pemrogr aman-Sistem-dan-Jaringan/Pertemuan 13\$ █	Menunggu Pesan dari Y-PC... Pesan dari Y-PC : Yes Tekan Enter Untuk Forward Pesan... ==== Forward ==== Pesan : Yes IP : 224.3.29.71 Port : 10000 Tekan Enter Untuk Menerima Pesan Lagi...█	Menunggu Pesan dari X-PC... Pesan dari X-PC : b'Yes' Dari IP : ('10.107.141.229', 39771) Tekan Enter Untuk Menerima Pesan Lagi...█	Menunggu Pesan dari X-PC... █

- Pada contoh diatas PC-X dan PC-A terhubung dalam IP multicast & Port yang sama sementara PC-B tidak sehingga tidak ada pesan yang muncul

----- Selesai -----