

TUGAS IMPLEMENTASI KASUS MENGUNAKAN CONCURRENCY PEMROGRAMAN JARINGAN



Nama : Rida Adila
NRP : 05111840000002
Kelas : PROGJAR D

Dosen : Royyana Muslim Ijtihadie, S.Kom., M.Kom., Ph.D.

Departemen Infomatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember (ITS) Surabaya
2021

TUGAS IMPLEMENTASI KASUS MENGGUNAKAN CONCURRENCY

Buatlah program yang mengimplementasikan

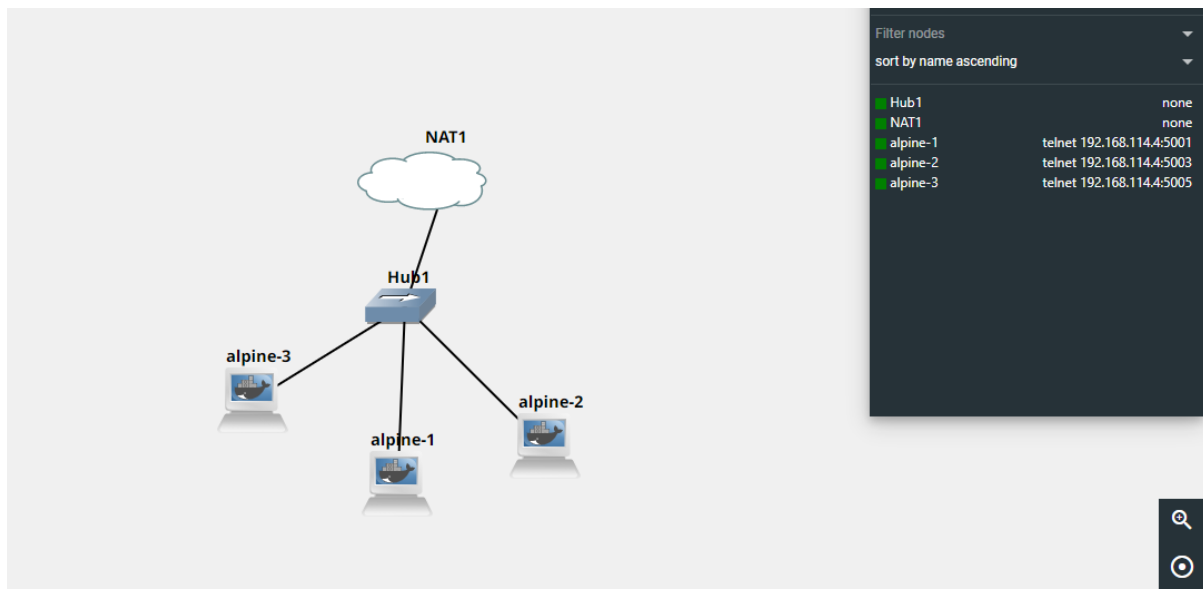
1. multi process
2. multi thread
3. multi process asynchronous
4. multi thread asynchronous

dengan menggunakan protokol transport UDP. kasus dapat didefinisikan sendiri. dan
Buatlah arsitektur jaringan anda sendiri di simulator GNS3

buatlah laporan dalam bentuk PDF yang berisikan screenshot dari

1. deskripsi kasus yang dibuat
2. gambar arsitektur jaringan (dalam simulator GNS3)
3. program yang dibuat (1-4)
4. hasil outputnya

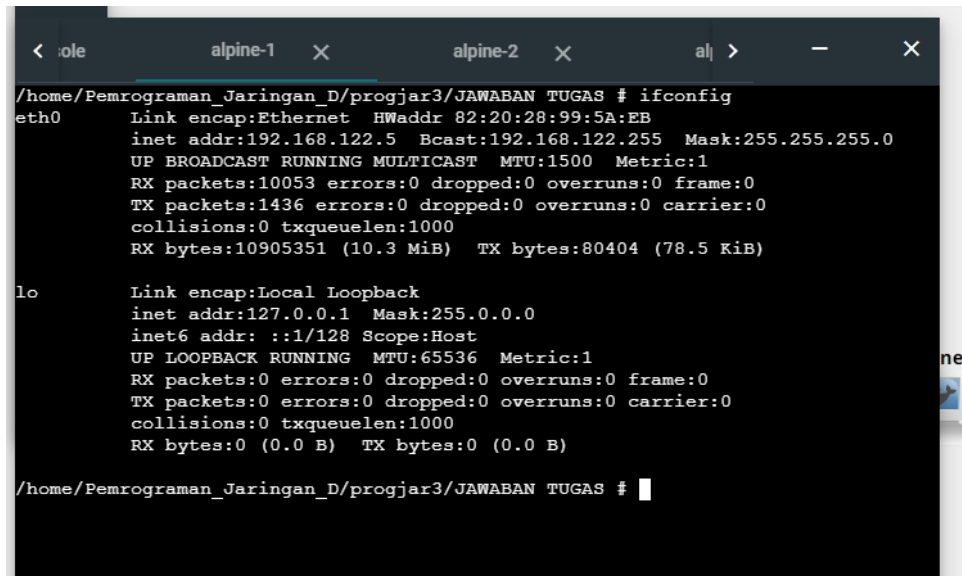
Deskripsi Studi Kasus



Terdapat 2 server dan 1 client. Dimana alpine1 sebagai server1, alpine 2 sebagai server2, dan alpine3 sebagai client. Client akan mengirimkan 2 gambar secara broadcast ke

server 1 dan server yang sudah disediakan tanpa harus melakukan download. Yakni gambar data.jpg dan asteroid.png. Lalu server1 dan server2 akan menerima data tsb.

- Informasi mengenai IP Address dari server1 pada alpine1 :

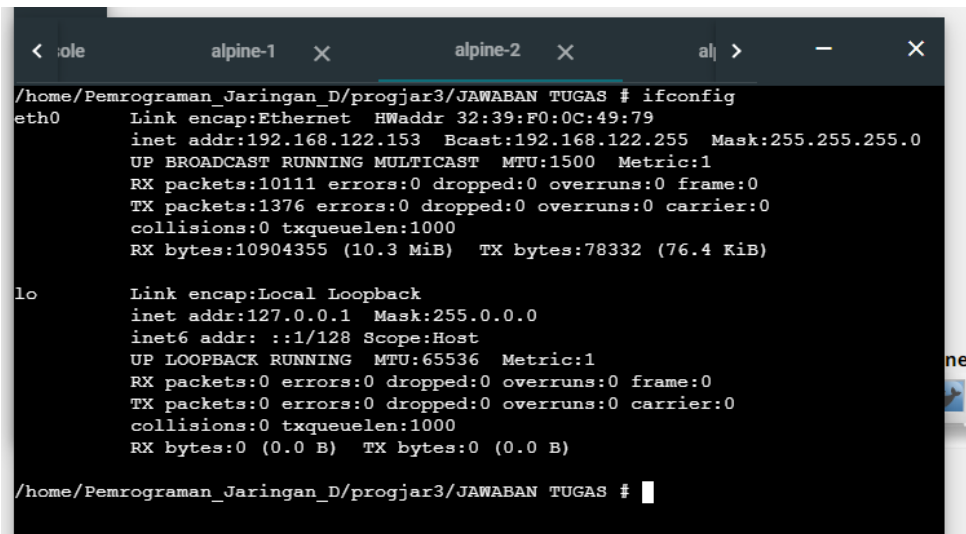


```
< .ole      alpine-1  X      alpine-2  X      al| >  -  X
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ifconfig
eth0      Link encap:Ethernet  HWaddr 82:20:28:99:5A:EB
          inet addr:192.168.122.5  Bcast:192.168.122.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10053 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1436 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10905351 (10.3 MiB)  TX bytes:80404 (78.5 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

- Informasi mengenai IP Address dari server2 pada alpine2 :



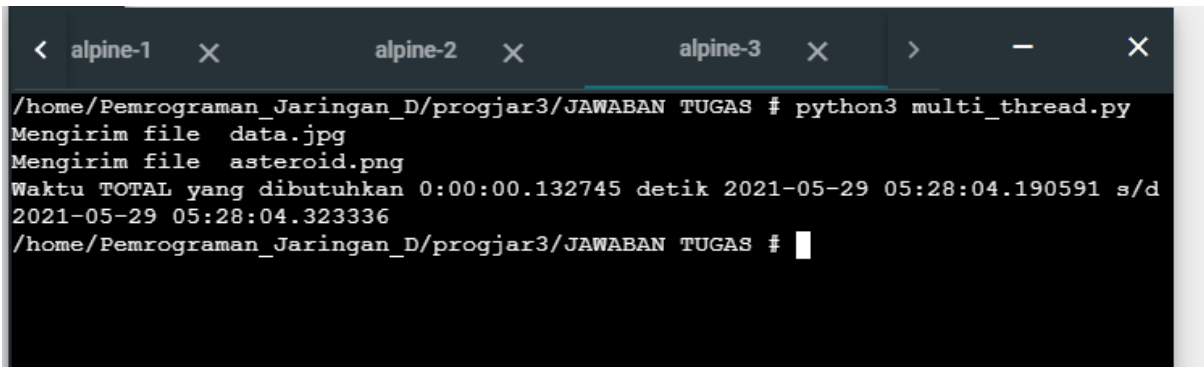
```
< .ole      alpine-1  X      alpine-2  X      al| >  -  X
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ifconfig
eth0      Link encap:Ethernet  HWaddr 32:39:F0:0C:49:79
          inet addr:192.168.122.153  Bcast:192.168.122.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10111 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1376 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10904355 (10.3 MiB)  TX bytes:78332 (76.4 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

1) MULTI THREAD

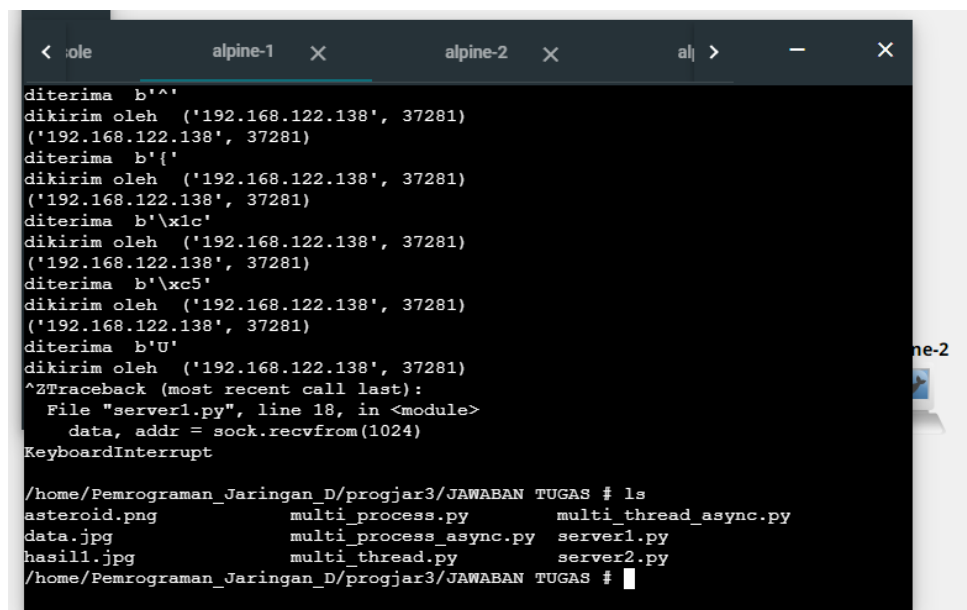
➤ Hasil Client :



```
< alpine-1 x alpine-2 x alpine-3 x > - x
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # python3 multi_thread.py
Mengirim file data.jpg
Mengirim file asteroid.png
Waktu TOTAL yang dibutuhkan 0:00:00.132745 detik 2021-05-29 05:28:04.190591 s/d
2021-05-29 05:28:04.323336
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Pada implementasi kasus multithread, terlihat bahwa 2 gambar yang dikirimkan membutuhkan total waktu sebesar 0.132745 detik

➤ Hasil server 1 :

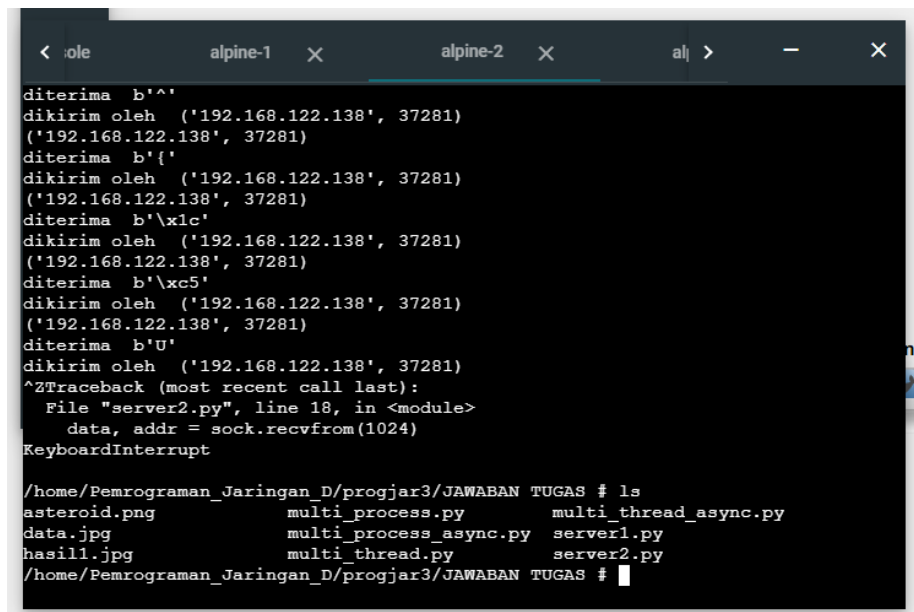


```
< alpine-1 x alpine-2 x alpine-3 x > - x
diterima b'^^'
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b{'
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b'\xlc'
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b'\xc5'
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b'U'
dikirim oleh ('192.168.122.138', 37281)
^ZTraceback (most recent call last):
  File "server1.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png      multi_process.py      multi_thread_async.py
data.jpg          multi_process_async.py server1.py
hasil1.jpg       multi_thread.py       server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 1, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

➤ Hasil server 2 :

A terminal window with three tabs: 'ole', 'alpine-1', and 'alpine-2'. The 'alpine-2' tab is active. It shows the output of a Python script 'server2.py'. The output consists of several lines of data received from a client, including IP addresses and coordinates. It ends with a traceback error: 'KeyboardInterrupt' and a directory listing of the current directory.

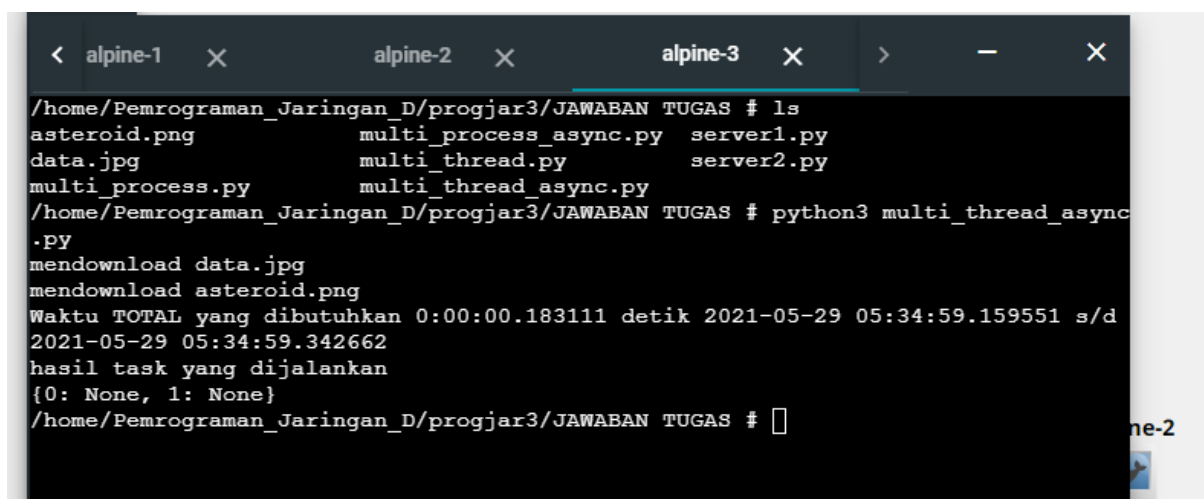
```
< ole alpine-1 alpine-2 alj > - X
diterima b'^'
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b{''
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b'\xlc'
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b'\xc5'
dikirim oleh ('192.168.122.138', 37281)
('192.168.122.138', 37281)
diterima b'U'
dikirim oleh ('192.168.122.138', 37281)
^ZTraceback (most recent call last):
  File "server2.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png      multi_process.py    multi_thread_async.py
data.jpg          multi_process_async.py server1.py
hasil1.jpg        multi_thread.py      server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 2, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

2) MULTI THREAD ASYNC

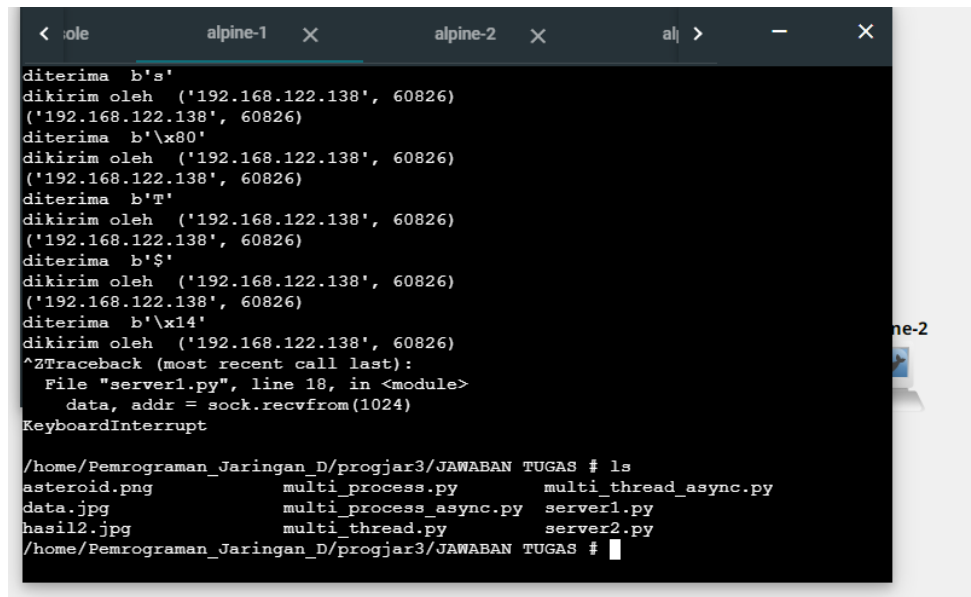
➤ Hasil Client :

A terminal window with three tabs: 'alpine-1', 'alpine-2', and 'alpine-3'. The 'alpine-3' tab is active. It shows the output of a Python script 'multi_thread_async.py'. The output includes a directory listing, the execution of the script, and the time taken to complete the task.

```
< alpine-1 alpine-2 alpine-3 > - X
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png      multi_process_async.py server1.py
data.jpg          multi_thread.py      server2.py
multi_process.py  multi_thread_async.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # python3 multi_thread_async
.py
mendownload data.jpg
mendownload asteroid.png
Waktu TOTAL yang dibutuhkan 0:00:00.183111 detik 2021-05-29 05:34:59.159551 s/d
2021-05-29 05:34:59.342662
hasil task yang dijalankan
{0: None, 1: None}
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Pada implementasi kasus multithread async, terlihat bahwa 2 gambar yang dikirimkan membutuhkan total waktu sebesar 0.183111 detik

➤ Hasil Server 1 :

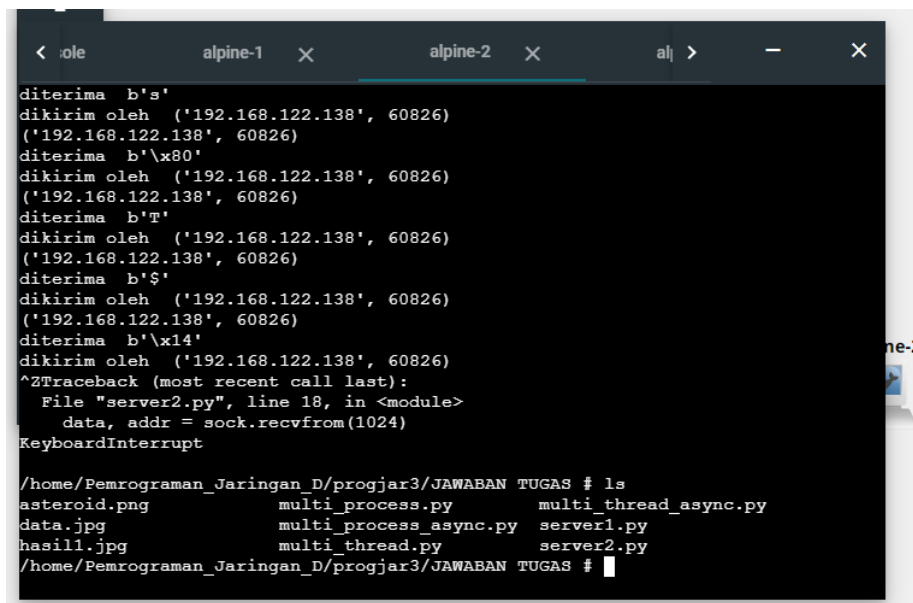


```
< file alpine-1 x alpine-2 x alj > - x
diterima b's'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'\x80'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'T'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'$'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'\x14'
dikirim oleh ('192.168.122.138', 60826)
^ZTraceback (most recent call last):
  File "server1.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png      multi_process.py    multi_thread_async.py
data.jpg          multi_process_async.py  server1.py
hasil2.jpg        multi_thread.py      server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 1, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

➤ Hasil Server 2 :



```
< file alpine-1 x alpine-2 x alj > - x
diterima b's'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'\x80'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'T'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'$'
dikirim oleh ('192.168.122.138', 60826)
('192.168.122.138', 60826)
diterima b'\x14'
dikirim oleh ('192.168.122.138', 60826)
^ZTraceback (most recent call last):
  File "server2.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

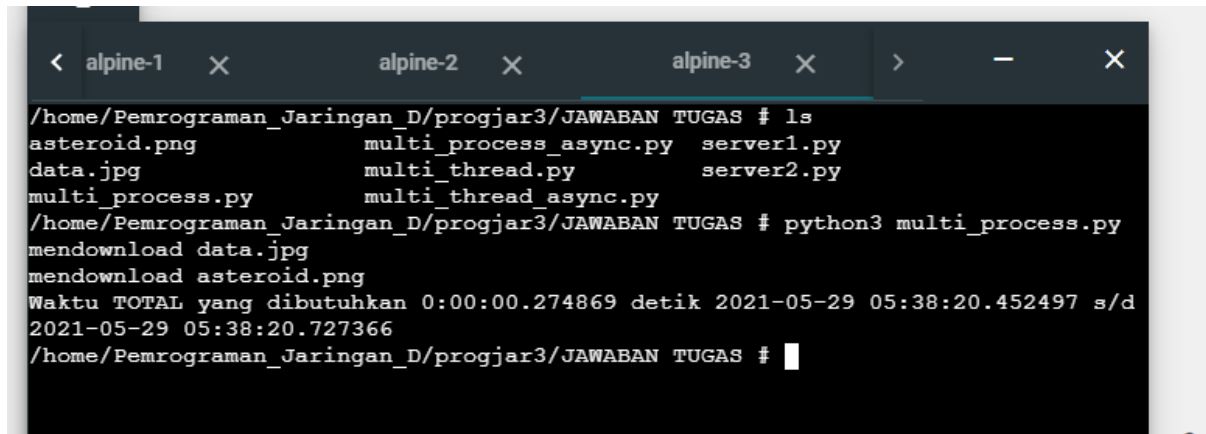
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png      multi_process.py    multi_thread_async.py
data.jpg          multi_process_async.py  server1.py
hasil1.jpg        multi_thread.py      server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 2, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil

gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

3) MULTIPROCESS :

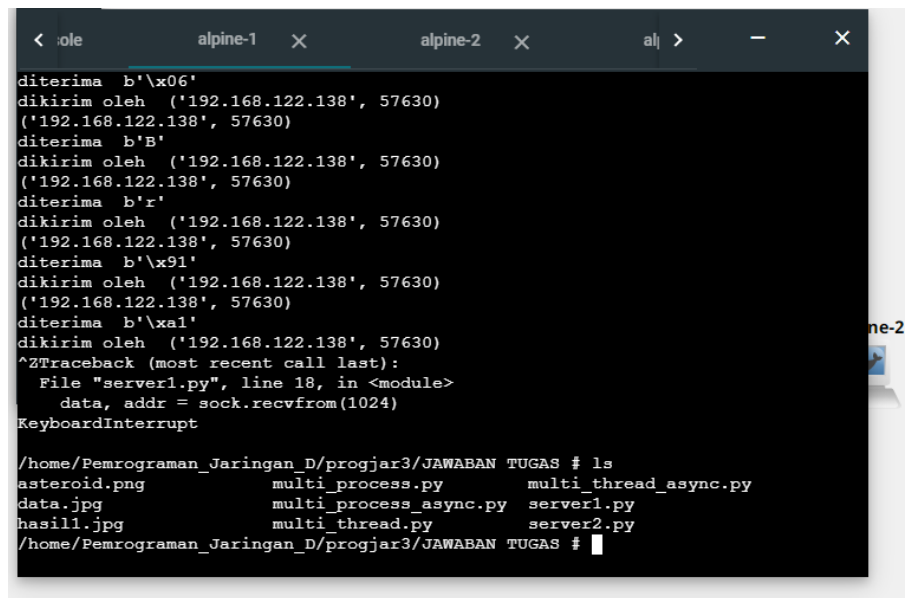
➤ Hasil Client :



```
< alpine-1 x alpine-2 x alpine-3 x > - x
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png          multi_process_async.py  server1.py
data.jpg              multi_thread.py         server2.py
multi_process.py      multi_thread_async.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # python3 multi_process.py
mendownload data.jpg
mendownload asteroid.png
Waktu TOTAL yang dibutuhkan 0:00:00.274869 detik 2021-05-29 05:38:20.452497 s/d
2021-05-29 05:38:20.727366
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Pada implementasi kasus multiprocessing, terlihat bahwa 2 gambar yang dikirimkan membutuhkan total waktu sebesar 0.274869 detik

➤ Hasil Server1 :



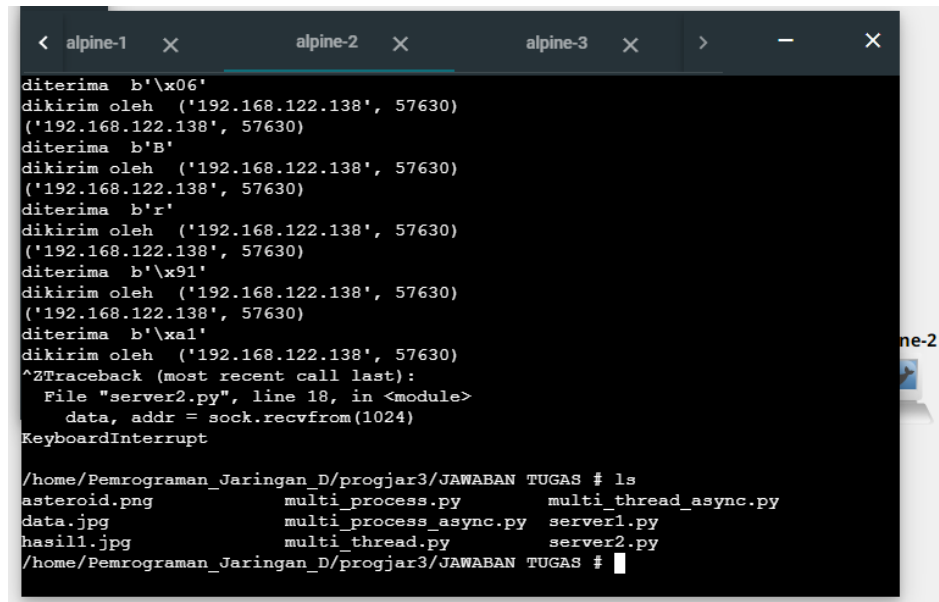
```
< role alpine-1 x alpine-2 x alpine-3 x > - x
diterima b'\x06'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'B'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'r'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'\x91'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'\xa1'
dikirim oleh ('192.168.122.138', 57630)
^ZTraceback (most recent call last):
  File "server1.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png          multi_process.py      multi_thread_async.py
data.jpg              multi_process_async.py  server1.py
hasill.jpg            multi_thread.py        server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 1, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil

gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

Server 2 :



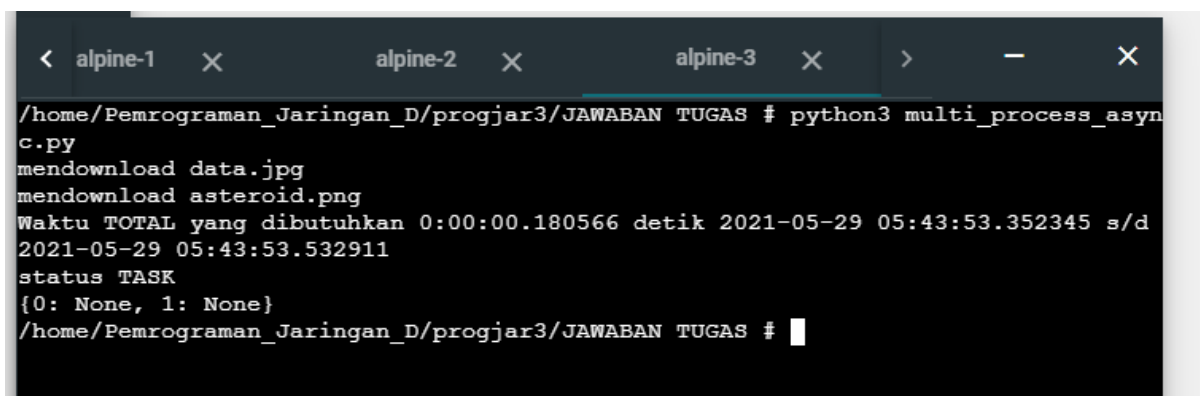
```
< alpine-1 x alpine-2 x alpine-3 x > - x
diterima b'\x06'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'B'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'r'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'\x91'
dikirim oleh ('192.168.122.138', 57630)
('192.168.122.138', 57630)
diterima b'\xa1'
dikirim oleh ('192.168.122.138', 57630)
^ZTraceback (most recent call last):
  File "server2.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png          multi_process.py      multi_thread_async.py
data.jpg              multi_process_async.py server1.py
hasil1.jpg            multi_thread.py        server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 2, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

4) MULTIPROCESS ASYNC :

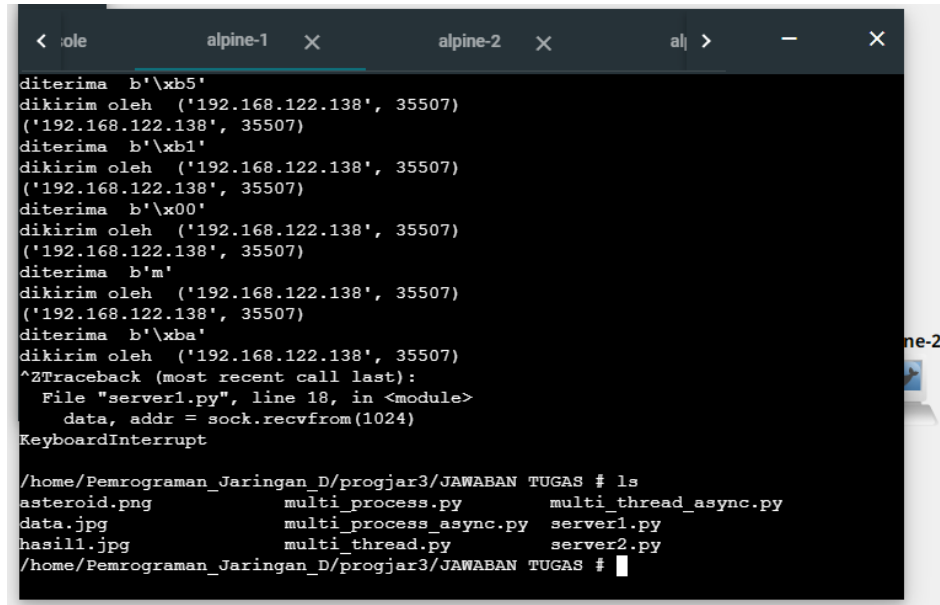
➤ Hasil Client :



```
< alpine-1 x alpine-2 x alpine-3 x > - x
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # python3 multi_process_async.py
mendownload data.jpg
mendownload asteroid.png
Waktu TOTAL yang dibutuhkan 0:00:00.180566 detik 2021-05-29 05:43:53.352345 s/d
2021-05-29 05:43:53.532911
status TASK
{0: None, 1: None}
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```


Pada implementasi kasus multiprocessing async, terlihat bahwa 2 gambar yang dikirimkan membutuhkan total waktu sebesar 0.180566 detik

➤ Hasil Server 1 :

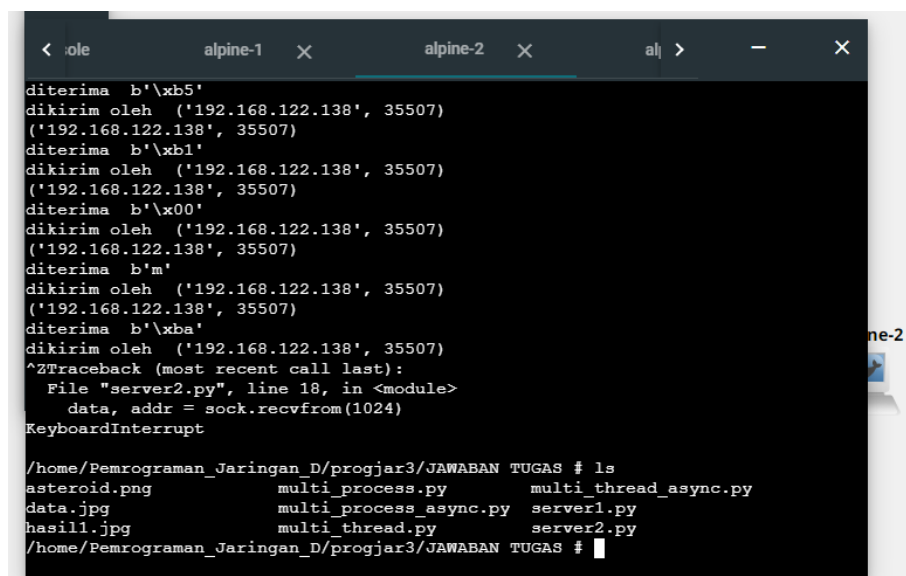


```
< role alpine-1 x alpine-2 x alj > - x
diterima b'\xb5'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'\xb1'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'\x00'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'm'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'\xba'
dikirim oleh ('192.168.122.138', 35507)
^ZTraceback (most recent call last):
  File "server1.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png          multi_process.py      multi_thread_async.py
data.jpg              multi_process_async.py server1.py
hasil1.jpg            multi_thread.py       server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 1, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

➤ Hasil Server 2 :



```
< role alpine-1 x alpine-2 x alj > - x
diterima b'\xb5'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'\xb1'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'\x00'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'm'
dikirim oleh ('192.168.122.138', 35507)
('192.168.122.138', 35507)
diterima b'\xba'
dikirim oleh ('192.168.122.138', 35507)
^ZTraceback (most recent call last):
  File "server2.py", line 18, in <module>
    data, addr = sock.recvfrom(1024)
KeyboardInterrupt

/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS # ls
asteroid.png          multi_process.py      multi_thread_async.py
data.jpg              multi_process_async.py server1.py
hasil1.jpg            multi_thread.py       server2.py
/home/Pemrograman_Jaringan_D/progjar3/JAWABAN TUGAS #
```

Hasil dari server 2, terlihat bahwa hanya ada 1 gambar yang dapat dilakukan write binary file untuk melihat hasilnya. Karena yang dikirim 2 gambar, seharusnya menerima 2 hasil gambar juga. Disini mungkin mohon maaf sebelumnya kodingan server saya masih ada yang belum benar.

KESIMPULAN :

Dari implementasi ke empat jenis concurrency yang telah dilakukan, didapatkan hasil pengiriman 2 gambar masing-masing jenis membutuhkan waktu sebesar :

- 1) Multithread ➔ 0.132745 detik
- 2) Multithread Async ➔ 0.183111 detik

Terlihat bahwa dari praktik yang telah saya lakukan, multithread biasa membutuhkan waktu lebih cepat daripada multithread async dalam melakukan pengiriman 2 gambar.

- 3) Multiprocess ➔ 0.274869 detik
- 4) Multiprocess Async ➔ 0.180566 detik

Dan multiprocess async membutuhkan waktu lebih cepat daripada multiprocess biasa dalam melakukan pengiriman 2 gambar.