

Tugas : Latihan 5 Thread  
Kelas : IF - 41 - 09  
Kelompok : 7  
Anggota : Aqmarina Alifah Ismahyati / 1301174058 (Tugas 3 )  
: Alvinda Julian Trismadi / 1301174147 (Tugas 2)  
: Jodi Kusuma / 1301174265 (Tugas 1)

## Screenshot Source Code dan Hasil Program

1. Tugas 1 : Pararel Ping
  - a. Source Code

```
01.paralel_ping.py > received_packages
1  # import os, re dan threading
2  import os, re, threading
3
4  # import time
5  import time
6
7  # buat kelas ip_check
8  class ip_check(threading.Thread):
9
10     # fungsi __init__; init untuk assign IP dan hasil respons = -1
11     def __init__(self, ip):
12         threading.Thread.__init__(self)
13         self.ip = ip
14         self.__successful_pings = -1
15
16     # fungsi utama yang dieksekusi ketika thread berjalan
17     def run(self):
18         # lakukan ping dengan perintah ping -n (gunakan os.popen())
19         ping_out = os.popen("ping -n 2 " + self.ip, "r")
20
21         # loop forever
22         while True:
23             # baca hasil respon setiap baris
24             line = ping_out.readline()
25
26             # break jika tidak ada line lagi
27             if not line:
28                 break
29
30             # baca hasil per line dan temukan pola Received = x
31             n_received = re.findall(received_packages, line)
32
33             # tampilkan hasilnya
34             if n_received:
35                 self.__successful_pings = int(n_received[0])
36
37     # fungsi untuk mengetahui status; 0 = tidak ada respon, 1 = hidup tapi ada loss, 2 = hidup
38     def status(self):
```

```

39         # 0 = tidak ada respon
40         if self.__successful_pings == 0:
41             return "Tidak Ada Respon"
42
43         # 1 = ada loss
44         elif self.__successful_pings == 1:
45             return "Ada Loss"
46         # 2 = hidup
47         elif self.__successful_pings == 2:
48             return "Hidup"
49         # -1 = seharusnya tidak terjadi
50         else:
51             return "Seharusnya Tidak Terjadi"
52
53     # buat regex untuk mengetahui isi dari r"Received = (\d)"
54     received_packages = re.compile(r"Received = (\d)")
55
56     # catat waktu awal
57     waktu_awal = time.time()
58
59     # buat list untuk menampung hasil pengecekan
60     check_results = []
61
62     # lakukan ping untuk 20 host
63     for suffix in range(1,20):
64         # tentukan IP host apa saja yang akan di ping
65         ip = "192.168.1." + str(suffix)
66
67         # panggil thread untuk setiap IP
68         thread = ip_check(ip)
69
70         # masukkan setiap IP dalam list
71         check_results.append(thread)
72
73     # jalankan thread
74     thread.start()
75
76     # untuk setiap IP yang ada di list
77     for el in check_results:
78
79         # tunggu hingga thread selesai
80         el.join()
81
82         # dapatkan hasilnya
83         print(el.ip, " : ",el.status())
84
85     # catat waktu berakhir
86     waktu_akhir = time.time()
87
88     # tampilkan selisih waktu akhir dan awal
89     print(waktu_akhir - waktu_awal)
90

```

## b. Output

```

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
PS C:\Users\Nekozawa\Music> & C:/ProgramData/Anaconda3/python.exe c:/Users/Nekozawa/Music/01.paralel_ping.py
192.168.1.1 : Hidup
192.168.1.2 : Hidup
192.168.1.3 : Hidup
192.168.1.4 : Hidup
192.168.1.5 : Hidup
192.168.1.6 : Hidup
192.168.1.7 : Tidak Ada Respon
192.168.1.8 : Hidup
192.168.1.9 : Tidak Ada Respon
192.168.1.10 : Hidup
192.168.1.11 : Hidup
192.168.1.12 : Hidup
192.168.1.13 : Hidup
192.168.1.14 : Hidup
192.168.1.15 : Hidup
192.168.1.16 : Hidup
192.168.1.17 : Hidup
192.168.1.18 : Hidup
192.168.1.19 : Hidup
8.633235454559326
PS C:\Users\Nekozawa\Music>

```

2. Tugas 2 : Multithreaded Server
  - a. Source Code
    - i. Server

```
02.server_thread.py X
02.server_thread.py > start_server
1  # import socket, sys, traceback dan threading
2  import socket
3  import sys
4  import traceback
5  import threading
6
7  # jalankan server
8  def main():
9      start_server()
10
11 # fungsi saat server dijalankan
12 def start_server():
13     # tentukan IP server
14     HOST = "192.168.1.8"
15
16     # tentukan port server
17     PORT = 55555
18
19     # buat socket bertipe TCP
20     soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
21
22     # option socket
23     soc.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
24     print("Socket dibuat")
25
26     # lakukan bind
27     try:
28         soc.bind((HOST,PORT))
29     except:
30         # exit pada saat error
31         print("Bind gagal. Error : " + str(sys.exc_info()))
32         sys.exit()
```

02.server\_thread.py X

02.server\_thread.py > start\_server

```
34     # listen hingga 5 antrian
35     soc.listen(5)
36     print("Socket mendengarkan")
37
38     # infinite loop, jangan reset setiap ada request
39     while True:
40         # terima koneksi
41         koneksi, address = soc.accept()
42
43         # dapatkan IP dan port
44         ip = address[0]
45         port = address[1]
46         print("Connected dengan " + ip + ":" + str(port))
47
48         # jalankan thread untuk setiap koneksi yang terhubung
49         try:
50             threading.Thread(target = client_thread, args = (koneksi, ip, port)).start()
51         except:
52             # print kesalahan jika thread tidak berhasil dijalankan
53             print("Thread tidak berjalan.")
54             traceback.print_exc()
55
56     # tutup socket
57     soc.close()
58
59
60 def client_thread(connection, ip, port, max_buffer_size = 4096):
61     # flag koneksi
62     is_active = True
63
64     # selama koneksi aktif
65     while is_active:
```

```

02.server_thread.py X
L5 > 02.server_thread.py > ...
66
67     # terima pesan dari client
68     client_input = connection.recv(max_buffer_size)
69
70     # dapatkan ukuran pesan
71     client_input_size = sys.getsizeof(client_input)
72
73     # print jika pesan terlalu besar
74     if client_input_size > max_buffer_size:
75         print("The input size is greater than expected {}")
76
77     # dapatkan pesan setelah didecode
78     decoded_input = client_input.decode('utf8').rstrip()
79
80     # jika "quit" maka flag koneksi = false, matikan koneksi
81     if "quit" in decoded_input:
82         # ubah flag
83         is_active = False
84         print("Client meminta keluar")
85
86         # matikan koneksi
87         connection.close()
88         print("Connection " + ip + ":" + str(port) + " ditutup")
89
90     else:
91         # tampilkan pesan dari client
92
93         print("client IP " + ip + ":" + str(port) + " = " + decoded_input)
94 # panggil fungsi utama
95 if __name__ == "__main__":
96     main()

```

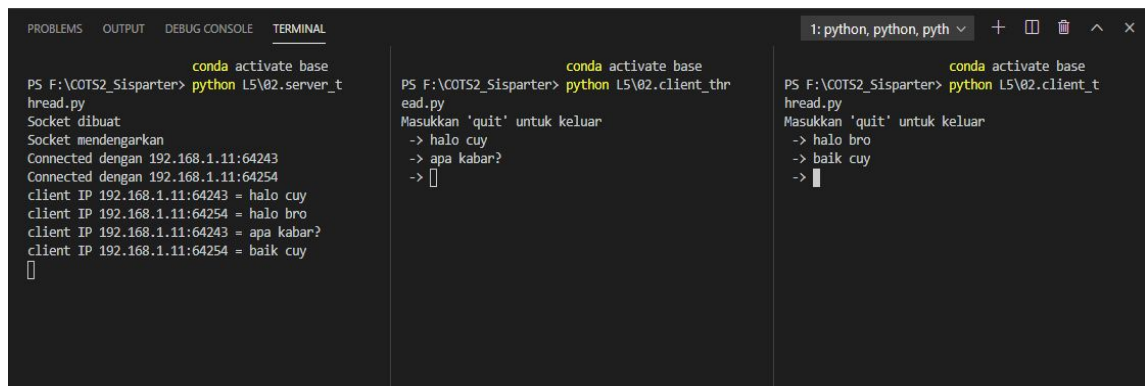
ii. client



```
02.client_thread.py X
02.client_thread.py > main
1  # import socket dan sys
2  import socket
3  import sys
4  # fungsi utama
5  def main():
6      # buat socket bertipe TCP
7      soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8
9      # tentukan IP server target
10     host = "192.168.1.8"
11
12     # tentukan port server
13     port = 55555
14
15     # lakukan koneksi ke server
16     try:
17         soc.connect((host,port))
18     except:
19         # print error
20         print("Koneksi error")
21         # exit
22         sys.exit()
```

```
02.client_thread.py X
02.client_thread.py > main
23
24     # tampilkan menu, enter quit to exit
25     print("Masukkan 'quit' untuk keluar")
26     message = input(" -> ")
27
28     # selama pesan bukan "quit", lakukan loop forever
29     while message != 'quit':
30         # kirimkan pesan yang ditulis ke server
31         soc.sendall(message.encode('utf8'))
32
33         # menu (user interface)
34         message = input(" -> ")
35
36     # send "quit" ke server
37     soc.send(b'--quit--')
38
39     # panggil fungsi utama
40     if __name__ == "__main__":
41         main()
```

## b. Output



```
conda activate base
PS F:\COTS2_Sisparter> python L5\02.server_thread.py
hread.py
Socket mendengarkan
Connected dengan 192.168.1.11:64243
Connected dengan 192.168.1.11:64254
client IP 192.168.1.11:64243 = halo cuy
client IP 192.168.1.11:64254 = halo bro
client IP 192.168.1.11:64243 = apa kabar?
client IP 192.168.1.11:64254 = baik cuy
[]

conda activate base
PS F:\COTS2_Sisparter> python L5\02.client_thread.py
hread.py
Masukkan 'quit' untuk keluar
-> halo cuy
-> apa kabar?
-> []

conda activate base
PS F:\COTS2_Sisparter> python L5\02.client_thread.py
hread.py
Masukkan 'quit' untuk keluar
-> halo bro
-> baik cuy
-> []
```

## 3. Tugas 3 : download\_file.py

### a. Source Code

```
1  # Import os, requests, threading, urllib, time
2  import os
3  import requests
4  import threading
5  import urllib.request, urllib.error, urllib.parse
6  import time
7
8  # Url file yang akan di download
9  url = "https://apod.nasa.gov/apod/image/1901/L0mbradellaTerraFinazzi.jpg"
10
11 # Membagi jumlah data yang akan di download
12 def buildRange(value, numsplits):
13     lst = []
14     """
15     # Perulangan untuk membagi jumlah data
16     for i in range(numsplits):
17         if i == 0:
18             lst.append('%s-%s' % (i, int(round(1 + i * value/(numsplits*1.0) + value/(numsplits*1.0)-1, 0))))
19         else:
20             lst.append('%s-%s' % (int(round(1 + i * value/(numsplits*1.0),0)), int(round(1 + i * value/(numsplits*1.0) +
21 value/(numsplits*1.0)-1, 0))))
22     return lst
23
24 # Membagi buffer dan jumlah thread saat download
25 class SplitBufferThreads(threading.Thread):
26     """ Splits the buffer to any number of threads
27     thereby, concurrently downloading through
28     ny number of threads.
29     """
30
31 # Membagi buffer
32 def __init__(self, url, byteRange):
33     super(SplitBufferThreads, self).__init__()
34     self.__url = url
35     self.__byteRange = byteRange
36     self.req = None
```

```

37     # Request url
38     def run(self):
39         self.req = urllib.request.Request(self.__url, headers={'Range': 'bytes=%s' % self.__byteRange})
40
41     # Mendapatkan file data dan dibaca
42     def getFileData(self):
43         return urllib.request.urlopen(self.req).read()
44
45 # Fungsi utama dengan membagi download menjadi 3 thread
46 def main(url=None, splitBy=3):
47     start_time = time.time()
48
49     # Jika tidak terdapat url
50     if not url:
51         print("Please Enter some url to begin download.")
52         return
53
54     # Jumlah byte yang akan di download
55     fileName = url.split('/')[-1]
56     sizeInBytes = requests.head(url, headers={'Accept-Encoding': 'identity'}).headers.get('content-length', None)
57     print("%s bytes to download." % sizeInBytes)
58     if not sizeInBytes:
59         print("Size cannot be determined.")
60         return
61
62     # Pembagian thread
63     dataLst = []
64     for idx in range(splitBy):
65         byteRange = buildRange(int(sizeInBytes), splitBy)[idx]
66         bufTh = SplitBufferThreads(url, byteRange)
67         bufTh.start()
68         bufTh.join()
69         dataLst.append(bufTh.getFileData())
70
71     content = b''.join(dataLst)

```

```

73     # Mengecek apakah file sudah didownload
74     if dataLst:
75         if os.path.exists(fileName):
76             os.remove(fileName)
77             print("--- %s seconds ---" % str(time.time() - start_time))
78             with open(fileName, 'wb') as fh:
79                 fh.write(content)
80             print("Finished Writing file %s" % fileName)
81
82 if __name__ == '__main__':
83     main(url)

```

## b. Output

```

(base) C:\Users\Aqmarina>python 03.download_file.py
3670260 bytes to download.
--- 81.98187446594238 seconds ---
Finished Writing file LOMbradellaTerraFinazzi.jpg

```



