

```

1  # Python3 program imitating a clock server
2
3  from functools import reduce
4  from dateutil import parser
5  import threading
6  import datetime
7  import socket
8  import time
9
10
11 # datastructure used to store client address and clock data
12 client_data = {}
13
14
15 ''' nested thread function used to receive
16 clock time from a connected client '''
17 def startReceivingClockTime(connector, address):
18
19     while True:
20         # receive clock time
21         clock_time_string = connector.recv(1024).decode()
22         clock_time = parser.parse(clock_time_string)
23         clock_time_diff = datetime.datetime.now() - \
24                             clock_time
25
26         client_data[address] = {
27             "clock_time" : clock_time,
28             "time_difference" : clock_time_diff,
29             "connector" : connector
30         }
31
32         print("Client Data updated with: " + str(address),
33               end = "\n\n")

```

```

34     time.sleep(5)
35
36
37 ''' master thread function used to open portal for
38 accepting clients over given port '''
39 def startConnecting(master_server):
40
41     # fetch clock time at slaves / clients
42     while True:
43         # accepting a client / slave clock client
44         master_slave_connector, addr = master_server.accept()
45         slave_address = str(addr[0]) + ":" + str(addr[1])
46
47         print(slave_address + " got connected successfully")
48
49         current_thread = threading.Thread(
50             target = startReceivingClockTime,
51             args = (master_slave_connector,
52                   slave_address, ))
53         current_thread.start()
54
55
56 # subroutine function used to fetch average clock difference
57 def getAverageClockDiff():
58
59     current_client_data = client_data.copy()
60
61     time_difference_list = list(client['time_difference']
62                                for client_addr, client
63                                in client_data.items())
64
65
66     sum_of_clock_difference = sum(time_difference_list, \

```

```
67         datetime.timedelta(0, 0))
68
69     average_clock_difference = sum_of_clock_difference \
70         / len(client_data)
71
72     return average_clock_difference
73
74
75     ''' master sync thread function used to generate
76     cycles of clock synchronization in the network '''
77 def synchronizeAllClocks():
78
79     while True:
80
81         print("New synchronization cycle started.")
82         print("Number of clients to be synchronized: " + \
83             str(len(client_data)))
84
85         if len(client_data) > 0:
86
87             average_clock_difference = getAverageClockDiff()
88
89             for client_addr, client in client_data.items():
90                 try:
91                     synchronized_time = \
92                         datetime.datetime.now() + \
93                             average_clock_difference
94
95                     client['connector'].send(str(
96                         synchronized_time).encode())
97
98                 except Exception as e:
99                     print("Something went wrong while " + \
```

```
99         print("Something went wrong while " + \
100             "sending synchronized time " + \
101             "through " + str(client_addr))
102
103     else :
104         print("No client data." + \
105             " Synchronization not applicable.")
106
107     print("\n\n")
108
109     time.sleep(5)
110
111
112 # function used to initiate the Clock Server / Master Node
113 def initiateClockServer(port = 8080):
114
115     master_server = socket.socket()
116     master_server.setsockopt(socket.SOL_SOCKET,
117                             socket.SO_REUSEADDR, 1)
118
119     print("Socket at master node created successfully\n")
120
121     master_server.bind(('', port))
122
123     # Start listening to requests
124     master_server.listen(10)
125     print("Clock server started...\n")
126
127     # start making connections
128     print("Starting to make connections...\n")
129     master_thread = threading.Thread(
130         target = startConnecting,
131         args = (master_server, ))
```

```
131         args = (master_server, )
132     master_thread.start()
133
134     # start synchronization
135     print("Starting synchronization parallelly...\n")
136     sync_thread = threading.Thread(
137         target = synchronizeAllClocks,
138         args = ()
139     )
140     sync_thread.start()
141
142
143     # Driver function
144     if __name__ == '__main__':
145
146         # Trigger the Clock Server
147         initiateClockServer(port = 8080)
```

Activate Windows  
Go to Settings to activate Windows

```
166     # Python3 program imitating a client process
167
168     from timeit import default_timer as timer
169     from dateutil import parser
170     import threading
171     import datetime
172     import socket
173     import time
174
175
176     # client thread function used to send time at client side
177     def startSendingTime(slave_client):
178
179         while True:
180             # provide server with clock time at the client
181             slave_client.send(str(
182                 datetime.datetime.now().encode()))
183
184             print("Recent time sent successfully",
185                 end = "\n\n")
186             time.sleep(5)
187
188
189     # client thread function used to receive synchronized time
190     def startReceivingTime(slave_client):
191
192         while True:
193             # receive data from the server
194             Synchronized_time = parser.parse(
195                 slave_client.recv(1024).decode())
196
197             print("Synchronized time at the client is: " + \
198                 str(Synchronized_time),
```

Activate Windows  
Go to Settings to activate Windows

```
197     print("Synchronized time at the client is: " + \
198           str(Synchronized_time),
199           end = "\n\n")
200
201
202     # function used to Synchronize client process time
203     def initiateSlaveClient(port = 8080):
204
205         slave_client = socket.socket()
206
207         # connect to the clock server on local computer
208         slave_client.connect(('127.0.0.1', port))
209
210         # start sending time to server
211         print("Starting to receive time from server\n")
212         send_time_thread = threading.Thread(
213             target = startSendingTime,
214             args = (slave_client, ))
215         send_time_thread.start()
216
217
218         # start receiving synchronized from server
219         print("Starting to receiving " + \
220               "synchronized time from server\n")
221         receive_time_thread = threading.Thread(
222             target = startReceivingTime,
223             args = (slave_client, ))
224         receive_time_thread.start()
225
226
227     # Driver function
228     if __name__ == '__main__':
229
```

Activate  
Go to Settings

```
File Edit Selection View Go Run Terminal Help • berkey.py - WADL Lab Manual - Visual Studio Code
226
227 # Driver function
228 if __name__ == '__main__':
229
230     # initialize the Slave / Client
231     initiateSlaveClient(port = 8080)
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
```

```
activities Terminal Feb 15 14:58 a3-402-12@a3-402-12: -
a3-402-12@a3-402-12:~$ python3
Python 3.8.18 (default, Nov 14 2022, 12:59:47)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> exit()
a3-402-12@a3-402-12:~$ gedit &
[1] 6190
a3-402-12@a3-402-12:~$ python3 berkeleys.py
Socket at master node created successfully
Clock server started...
Starting to make connections...
Starting synchronization parallelly...
New synchronization cycle started.
Number of clients to be synchronized: 0
No client data. Synchronization not applicable.
Starting to receive time from server
127.0.0.1:50476 got connected successfully
Client Data updated with: 127.0.0.1:50476
Starting to receiving synchronized time from server
Recent time sent successfully
Recent time sent successfully
New synchronization cycle started.
Number of clients to be synchronized: 1
Synchronized time at the client is: 2023-02-15 14:57:02.996328
Client Data updated with: 127.0.0.1:50476
Recent time sent successfully
Client Data updated with: 127.0.0.1:50476
```