

PROIECT PROGRAMARE SOCKET

Problema 4.3

Serverul intoarce suma cifrelor din IP-ul clientului.

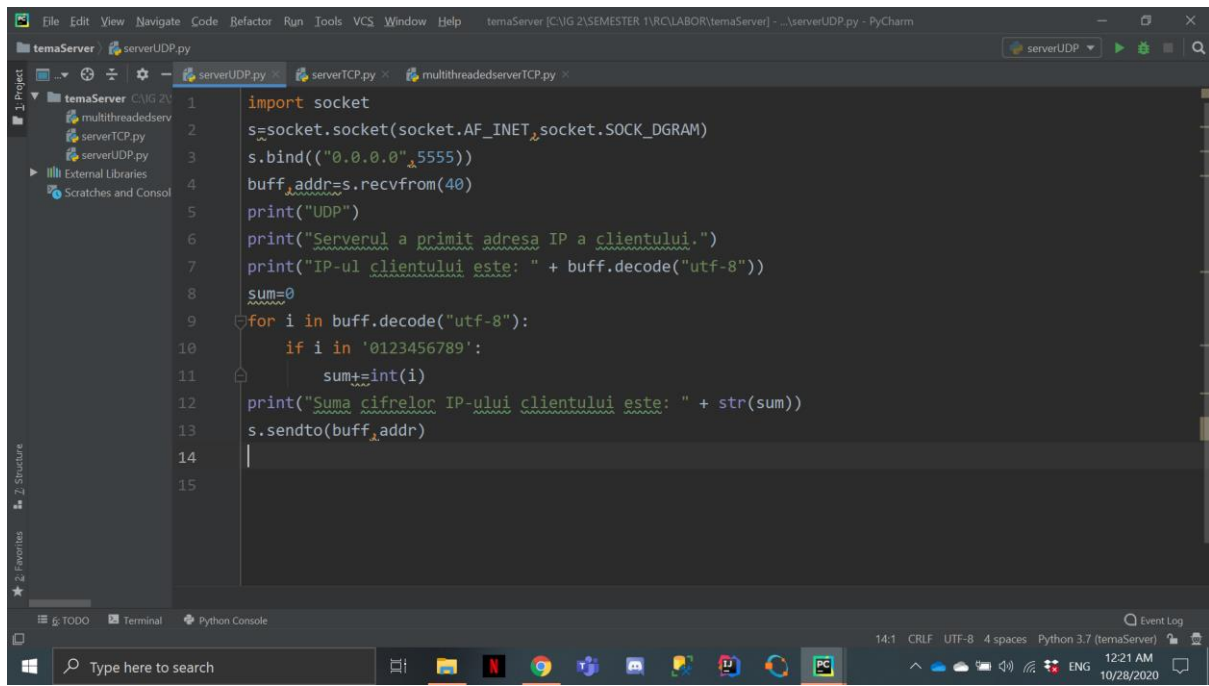
Cum am rezolvat problema?

Am rezolvat problema in UDP, TCP si TCP concurent, utilizand un server multithreaded si avand mai multi clienti. Modul in care am obtinut adresa IP a clientului este urmatorul: in client am obtinut numele host-ului utilizand metoda *gethostname*, iar apoi am obtinut IP-ul cu metoda *gethostbyname*, ulterior am trimis informatia catre server pentru ca acesta sa parcurga adresa IP si sa obtina suma cifrelor din aceasta.

Mai jos am atasat screenshot-uri cu fiecare varianta de rezolvare(UDP, TCP si TCP Multithreaded), dar si ce am obtinut dupa rulara fiecareia.

UDP

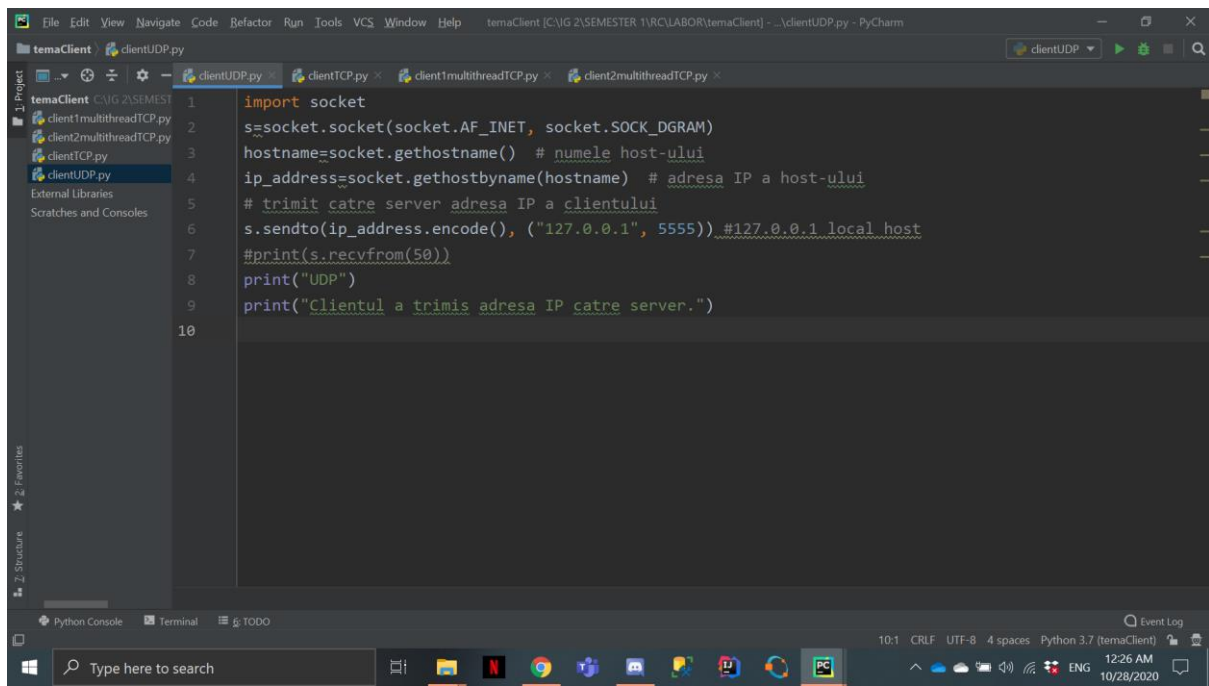
-server UDP Python-



```
1 import socket
2 s=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
3 s.bind(("0.0.0.0",5555))
4 buff,addr=s.recvfrom(4096)
5 print("UDP")
6 print("Serverul a primit adresa IP a clientului.")
7 print("IP-ul clientului este: " + buff.decode("utf-8"))
8 sum=0
9 for i in buff.decode("utf-8"):
10     if i in '0123456789':
11         sum+=int(i)
12 print("Suma cifrelor IP-ului clientului este: " + str(sum))
13 s.sendto(buff,addr)
14
15
```

- in primele doua linii de cod fac un socket(o instanta socket care primeste 2 parametrii: socket.AF_INET, care reprezinta protocolul si socket.SOCK_DGRAM care ii corespunde UDP-ului) * s este un file descriptor pentru socket
- cu bind imi rezerv un port
- serverul primeste informatia de la client, adica adresa IP
- aceasta este stocata intr-un buffer, pe care il parcurg pentru a face suma cifrelor din IP
- astfel, serverul intoarce suma cifrelor din IP-ul clientului

-client UDP Python-

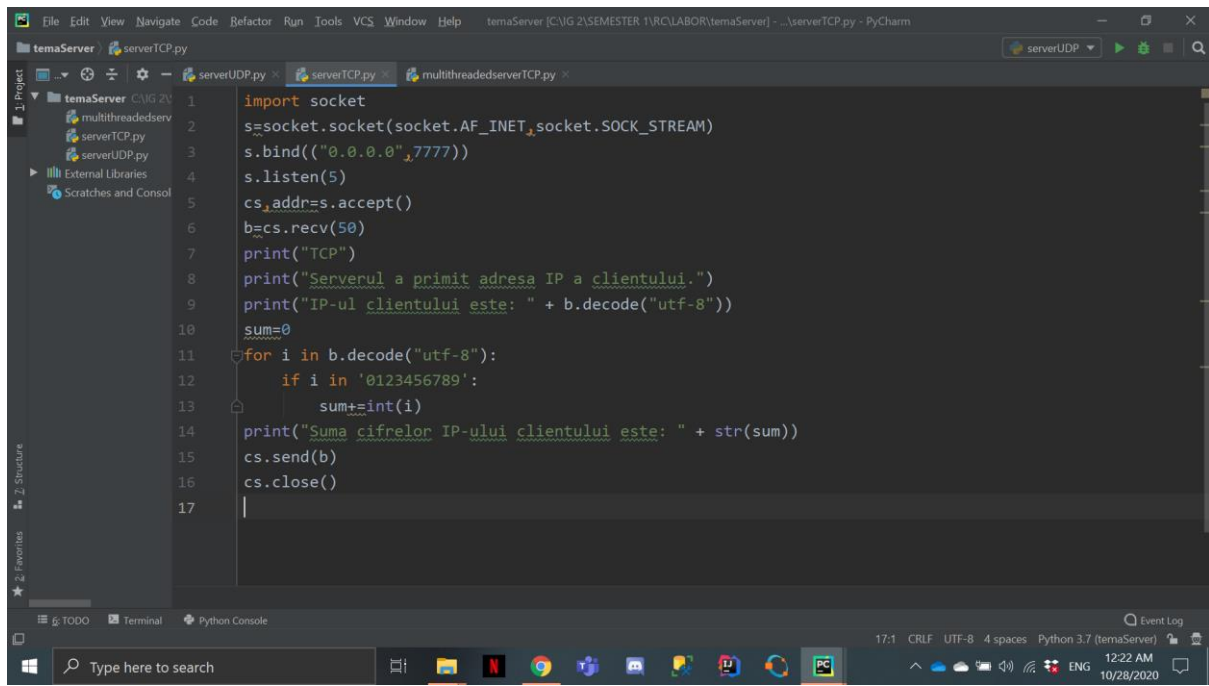


```
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a host-ului
5 # trimite catre server adresa IP a clientului
6 s.sendto(ip_address.encode(), ("127.0.0.1", 5555)) #127.0.0.1 local host
7 #print(s.recvfrom(50))
8 print("UDP")
9 print("Clientul a trimis adresa IP catre server.")
10
```

- in primele doua linii de cod fac un socket(o instanta socket care primeste 2 parametrii: socket.AF_INET, care reprezinta protocolul si socket.SOCK_DGRAM care ii corespunde UDP-ului) * s este un file descriptor pentru socket
- se obtine numele host-ului utilizand metoda gethostname
- se obtine adresa IP utilizand metoda gethostbyname
- clientul trimite IP-ul catre server

TCP

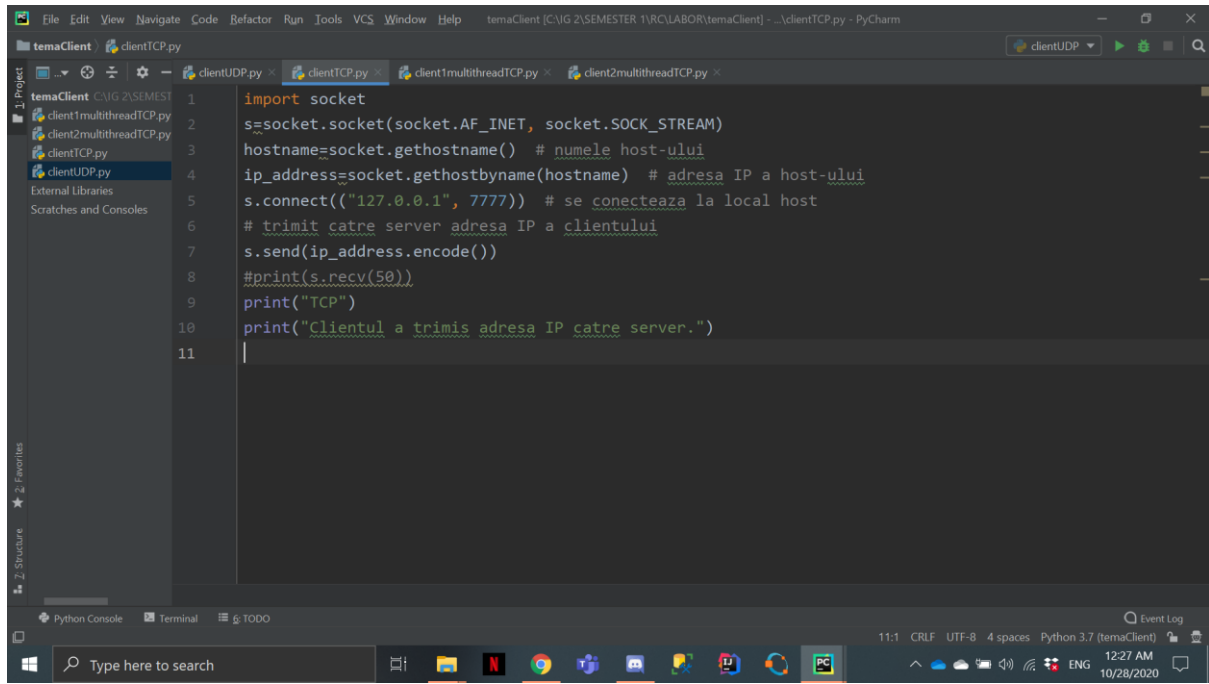
-server TCP Python-



```
1 import socket
2 s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
3 s.bind(("0.0.0.0",7777))
4 s.listen(5)
5 cs_addr=s.accept()
6 b=cs.recv(50)
7 print("TCP")
8 print("Serverul a primit adresa IP a clientului.")
9 print("IP-ul clientului este: " + b.decode("utf-8"))
10 sum=0
11 for i in b.decode("utf-8"):
12     if i in '0123456789':
13         sum+=int(i)
14 print("Suma cifrelor IP-ului clientului este: " + str(sum))
15 cs.send(b)
16 cs.close()
17
```

- in primele doua linii de cod fac un socket(o instanta socket care primeste 2 parametri: socket.AF_INET, care reprezinta protocolul si socket.SOCK_STREAM care ii corespunde TCP-ului)
- cu bind se rezerva un port
- listen(5) reprezinta numarul de conexiuni care sunt tinute in coada pana sunt preluate de sistemul de operare
- cs reprezinta descriptorul de socket
- serverul accepta informatia de la client si anume, adica adresa IP
- aceasta este primita de la client si stocata intr-un buffer, pe care il parcurg pentru a face suma cifrelor din IP
- astfel, serverul intoarce suma cifrelor din IP-ul clientului

-client TCP Python-



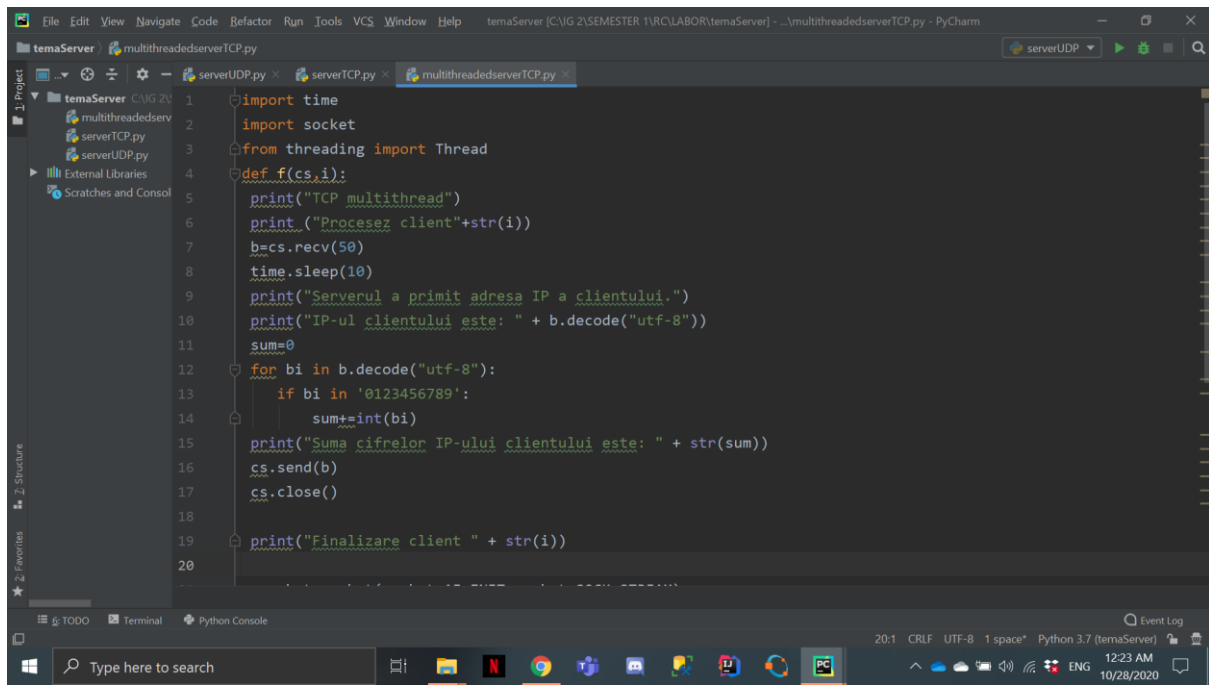
```
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a host-ului
5 s.connect(("127.0.0.1", 7777)) # se conecteaza la local host
6 # trimite catre server adresa IP a clientului
7 s.send(ip_address.encode())
8 #print(s.recv(50))
9 print("TCP")
10 print("Clientul a trimis adresa IP catre server.")
11 |
```

- in primele doua linii de cod fac un socket(o instanta socket care primeste 2 parametrii: socket.AF_INET, care reprezinta protocolul si socket.SOCK_STREAM care ii corespunde TCP-ului) * s este un file descriptor pentru socket
- se obtine numele host-ului utilizand metoda gethostname
- se obtine adresa IP utilizand metoda gethostbyname
- clientul se conecteaza la local host
- clientul trimite IP-ul catre server

TCP Multithread

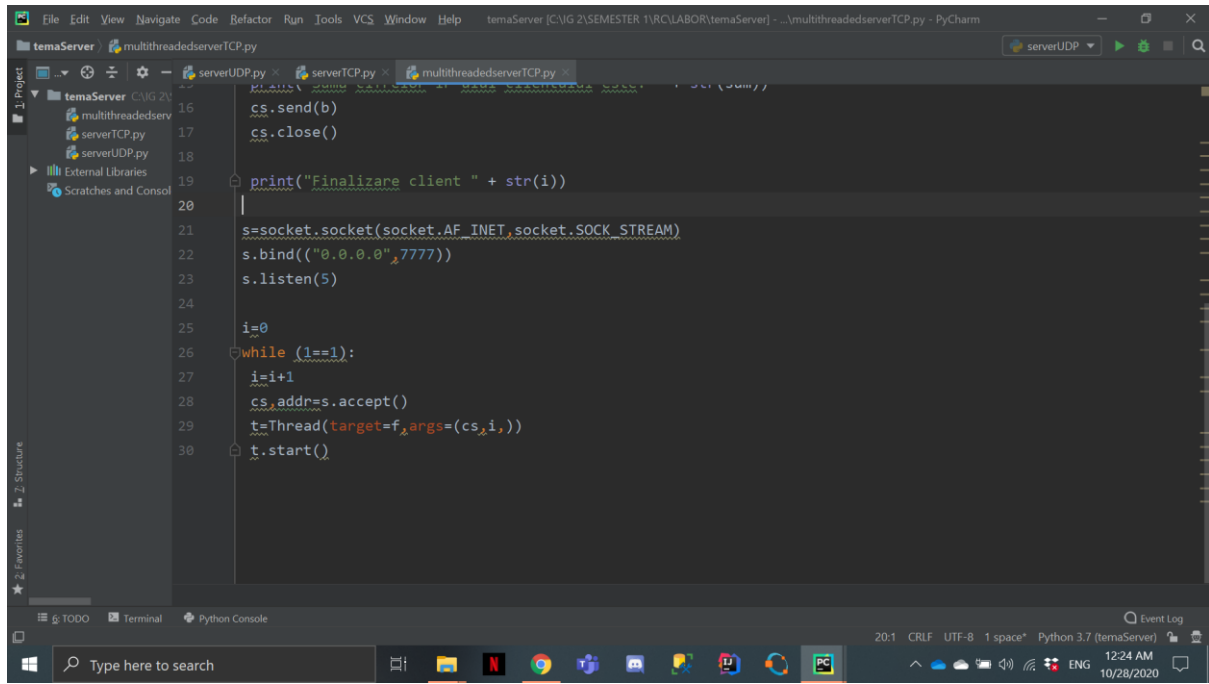
-multithreaded server TCP Python-

- acest server comunica cu mai multi clienti in acelasi timp in aceeasi retea



```
1 import time
2 import socket
3 from threading import Thread
4 def f(cs,i):
5     print("TCP multithread")
6     print("Procesez client"+str(i))
7     b=cs.recv(50)
8     time.sleep(10)
9     print("Serverul a primit adresa IP a clientului.")
10    print("IP-ul clientului este: " + b.decode("utf-8"))
11    sum=0
12    for bi in b.decode("utf-8"):
13        if bi in '0123456789':
14            sum+=int(bi)
15    print("Suma cifrelor IP-ului clientului este: " + str(sum))
16    cs.send(b)
17    cs.close()
18
19    print("Finalizare client " + str(i))
20
```

- in functia de thread stochez intr-un buffer b ceea ce primesc de la client
- cs reprezinta descriptorul de socket
- serverul accepta informatia de la client si anume, adica adresa IP
- aceasta este primita de la client si stocata intr-un buffer, pe care il parcurg pentru a face suma cifrelor din IP
- astfel, serverul intoarce suma cifrelor din IP-ul clientului



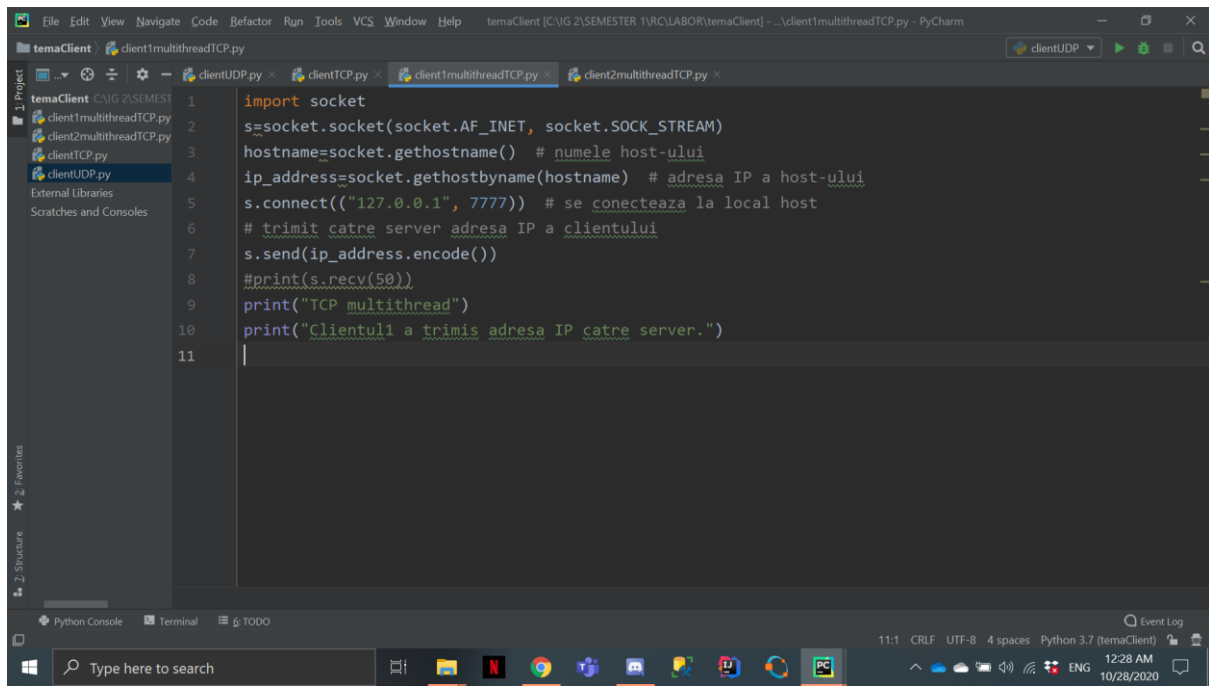
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help temaServer [C:\UG 2\SEMESTER 1\RC\LABOR\temaServer] - \multithreadserverTCP.py - PyCharm
temaServer
├── multithreadserverTCP.py
├── serverTCP.py
├── serverUDP.py
└── External Libraries
  └── Scratches and Console

16 cs.send(b)
17 cs.close()
18
19 print("Finalizare client " + str(i))
20
21 s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
22 s.bind(("0.0.0.0",7777))
23 s.listen(5)
24
25 i=0
26 while (1==1):
27     i=i+1
28     cs,addr=s.accept()
29     t=Thread(target=f,args=(cs,i,))
30     t.start()
```

- fac un socket(o instanta socket care primeste 2 parametrii: socket.AF_INET, care reprezinta protocolul si socket.SOCK_STREAM care ii corespunde TCP-ului)
- cu bind se rezerva un port
- listen(5) reprezinta numarul de conexiuni care sunt tinute in coada pana sunt preluate de sistemul de operare
- while-ul ruleaza la infinit, pana clientul se termina
- cu accept se stabileste conexiunea cu clientul
- se creeaza un thread nou

Clientii sunt exact la fel ca la TCP-ul fara multithreaded server.

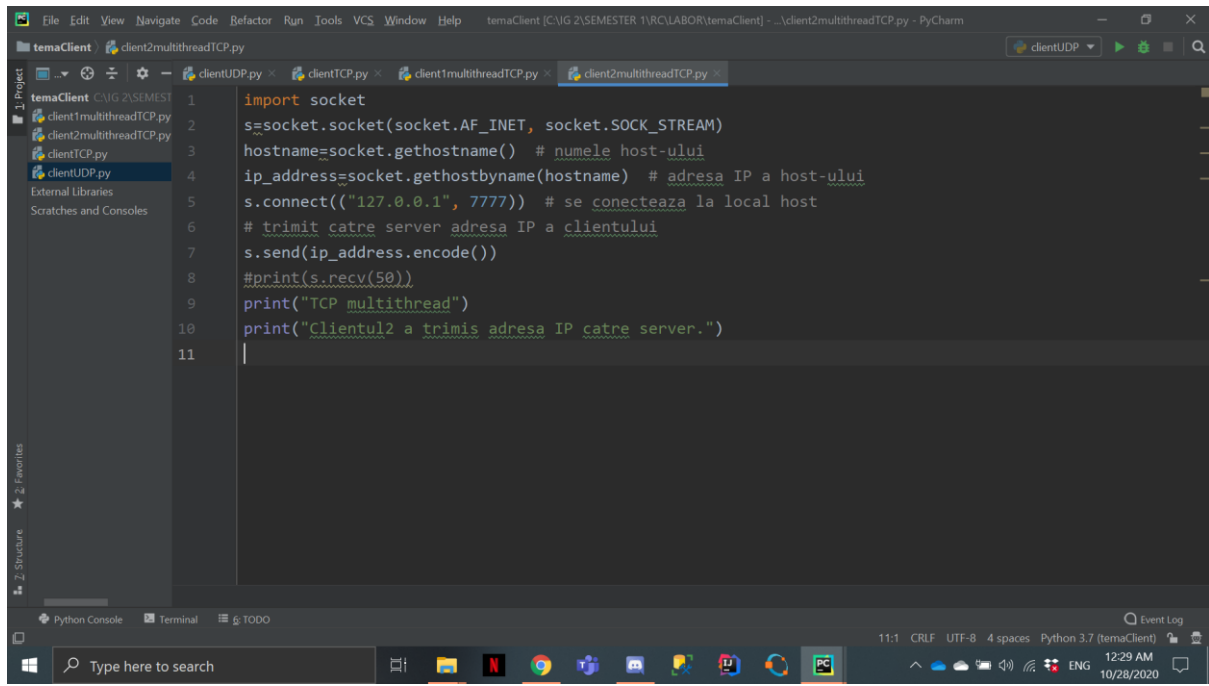
-client 1 TCP Python-



```
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a host-ului
5 s.connect(("127.0.0.1", 7777)) # se conecteaza la local host
6 # trimite catre server adresa IP a clientului
7 s.send(ip_address.encode())
8 #print(s.recv(50))
9 print("TCP multithread")
10 print("Clientul1 a trimis adresa IP catre server.")
11 |
```

- in primele doua linii de cod fac un socket(o instanta socket care primeste 2 parametrii: socket.AF_INET, care reprezinta protocolul si socket.SOCK_STREAM care ii corespunde TCP-ului) * s este un file descriptor pentru socket
- se obtine numele host-ului utilizand metoda gethostname
- se obtine adresa IP utilizand metoda gethostbyname
- clientul se conecteaza la local host
- clientul trimite IP-ul catre server

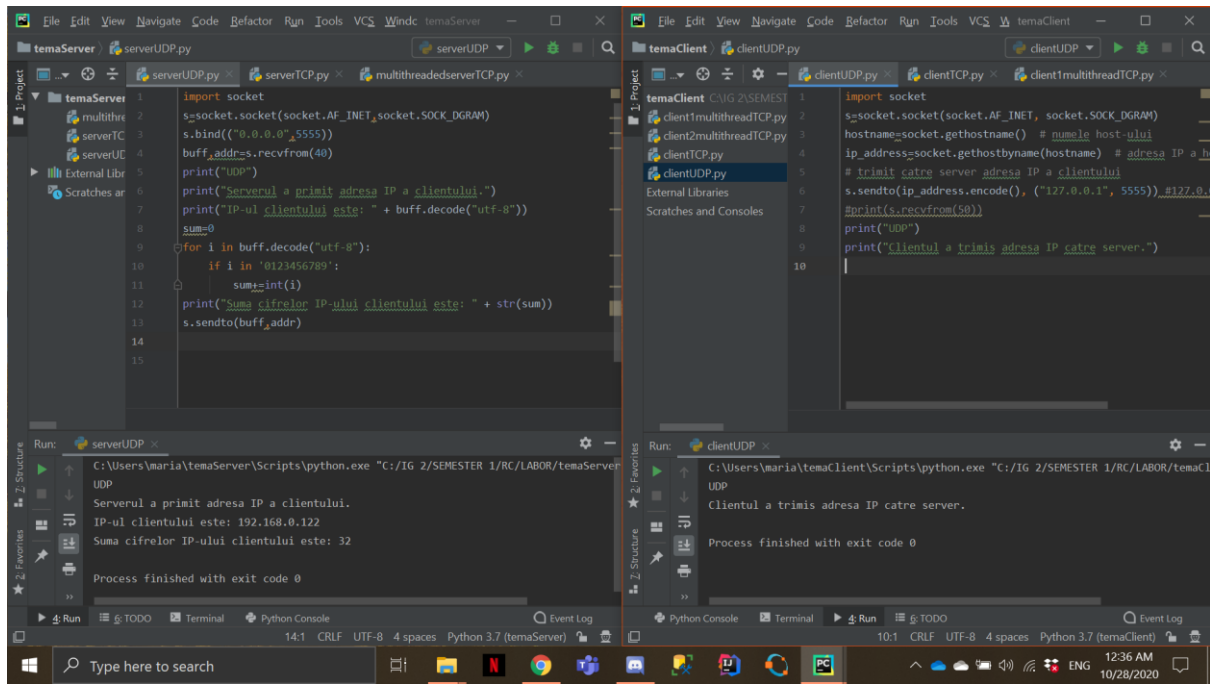
-client 2 TCP Python-



```
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a host-ului
5 s.connect(("127.0.0.1", 7777)) # se conecteaza la local host
6 # trimite catre server adresa IP a clientului
7 s.send(ip_address.encode())
8 #print(s.recv(50))
9 print("TCP multithread")
10 print("Clientul2 a trimis adresa IP catre server.")
11 |
```

RULARE:

-UDP-



The screenshot shows two IDE windows side-by-side. The left window, titled 'temaServer', displays the code for 'serverUDP.py'. The code imports 'socket', binds to '0.0.0.0' on port 5555, and listens for incoming UDP packets. It prints the received IP address and data, and calculates the sum of the digits of the IP address. The right window, titled 'temaClient', displays the code for 'clientUDP.py'. The code imports 'socket', gets the host name and IP address, and sends a UDP packet to the server. Both windows show the output of the program in the 'Run' console. The server console shows the received IP address '192.168.0.122' and the sum of its digits '32'. The client console shows the message 'Clientul a trimis adresa IP catre server.' and 'Process finished with exit code 0'.

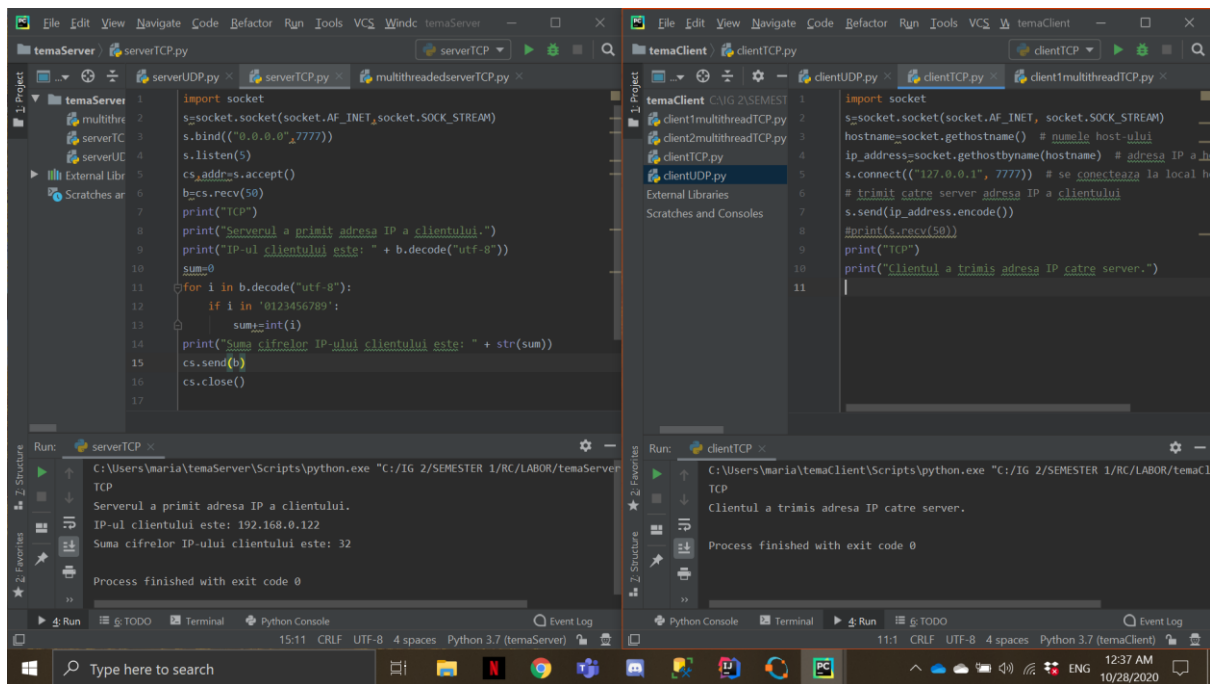
```
temaServer\serverUDP.py
1 import socket
2 s=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
3 s.bind(("0.0.0.0",5555))
4 buff_addr=s.recvfrom(40)
5 print("UDP")
6 print("Serverul a primit adresa IP a clientului.")
7 print("IP-ul clientului este: " + buff.decode("utf-8"))
8 sum=0
9 for i in buff.decode("utf-8"):
10     if i in '0123456789':
11         sum+=int(i)
12 print("Suma cifrelor IP-ului clientului este: " + str(sum))
13 s.sendto(buff_addr)
14
15

Run: serverUDP
C:\Users\maria\temaServer\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaServer
UDP
Serverul a primit adresa IP a clientului.
IP-ul clientului este: 192.168.0.122
Suma cifrelor IP-ului clientului este: 32
Process finished with exit code 0

temaClient\clientUDP.py
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a h
5 # trimite catre server adresa IP a clientului
6 s.sendto(ip_address.encode(), ("127.0.0.1", 5555)) #127.0.0.1
7 #print(s.recvfrom(50))
8 print("UDP")
9 print("Clientul a trimis adresa IP catre server.")
10
11

Run: clientUDP
C:\Users\maria\temaClient\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaC
UDP
Clientul a trimis adresa IP catre server.
Process finished with exit code 0
```

-TCP-



The screenshot shows two IDE windows side-by-side. The left window, titled 'temaServer', displays the code for 'serverTCP.py'. The code imports 'socket', binds to '0.0.0.0' on port 7777, and listens for incoming TCP connections. It prints the received IP address and data, and calculates the sum of the digits of the IP address. The right window, titled 'temaClient', displays the code for 'clientTCP.py'. The code imports 'socket', gets the host name and IP address, and sends a TCP packet to the server. Both windows show the output of the program in the 'Run' console. The server console shows the received IP address '192.168.0.122' and the sum of its digits '32'. The client console shows the message 'Clientul a trimis adresa IP catre server.' and 'Process finished with exit code 0'.

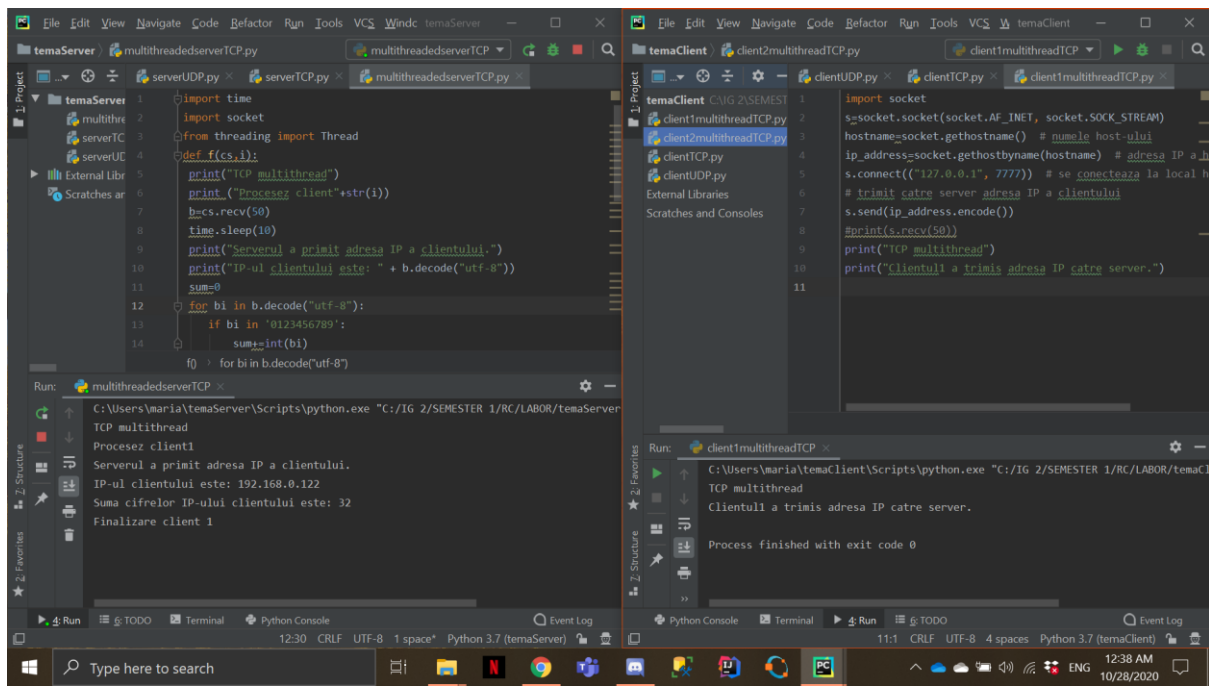
```
temaServer\serverTCP.py
1 import socket
2 s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
3 s.bind(("0.0.0.0",7777))
4 s.listen(5)
5 cs,addr=s.accept()
6 b=cs.recv(50)
7 print("TCP")
8 print("Serverul a primit adresa IP a clientului.")
9 print("IP-ul clientului este: " + b.decode("utf-8"))
10 sum=0
11 for i in b.decode("utf-8"):
12     if i in '0123456789':
13         sum+=int(i)
14 print("Suma cifrelor IP-ului clientului este: " + str(sum))
15 cs.send(b)
16 cs.close()
17

Run: serverTCP
C:\Users\maria\temaServer\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaServer
TCP
Serverul a primit adresa IP a clientului.
IP-ul clientului este: 192.168.0.122
Suma cifrelor IP-ului clientului este: 32
Process finished with exit code 0

temaClient\clientTCP.py
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a h
5 s.connect(("127.0.0.1", 7777)) # se conecteaza la local h
6 # trimite catre server adresa IP a clientului
7 s.send(ip_address.encode())
8 #print(s.recv(50))
9 print("TCP")
10 print("Clientul a trimis adresa IP catre server.")
11
12

Run: clientTCP
C:\Users\maria\temaClient\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaC
TCP
Clientul a trimis adresa IP catre server.
Process finished with exit code 0
```

-TCP multithread-



```
temaServer\multithreadedserverTCP.py
1 import time
2 import socket
3 from threading import Thread
4 def f(cs,i):
5     print("TCP multithread")
6     print("Procesez client"+str(i))
7     bcs.recv(50)
8     time.sleep(10)
9     print("Serverul a primit adresa IP a clientului.")
10    print("IP-ul clientului este: " + b.decode("utf-8"))
11    sum=0
12    for bi in b.decode("utf-8"):
13        if bi in '0123456789':
14            sum+=int(bi)
15    for bi in b.decode("utf-8")
```

```
temaClient\client2multithreadTCP.py
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a host-ului
5 s.connect(("127.0.0.1", 7777)) # se conecteaza la local host
6 # trimite catre server adresa IP a clientului
7 s.send(ip_address.encode())
8 #primeste de la server
9 s.recv(50)
10 print("TCP multithread")
11 print("Clientul 1 a trimis adresa IP catre server.")
```

Run: multithreadedserverTCP

C:\Users\maria\temaServer\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaServer/multithreadedserverTCP.py"

TCP multithread

Procesez client1

Serverul a primit adresa IP a clientului.

IP-ul clientului este: 192.168.0.122

Suma cifrelor IP-ului clientului este: 32

Finalizare client 1

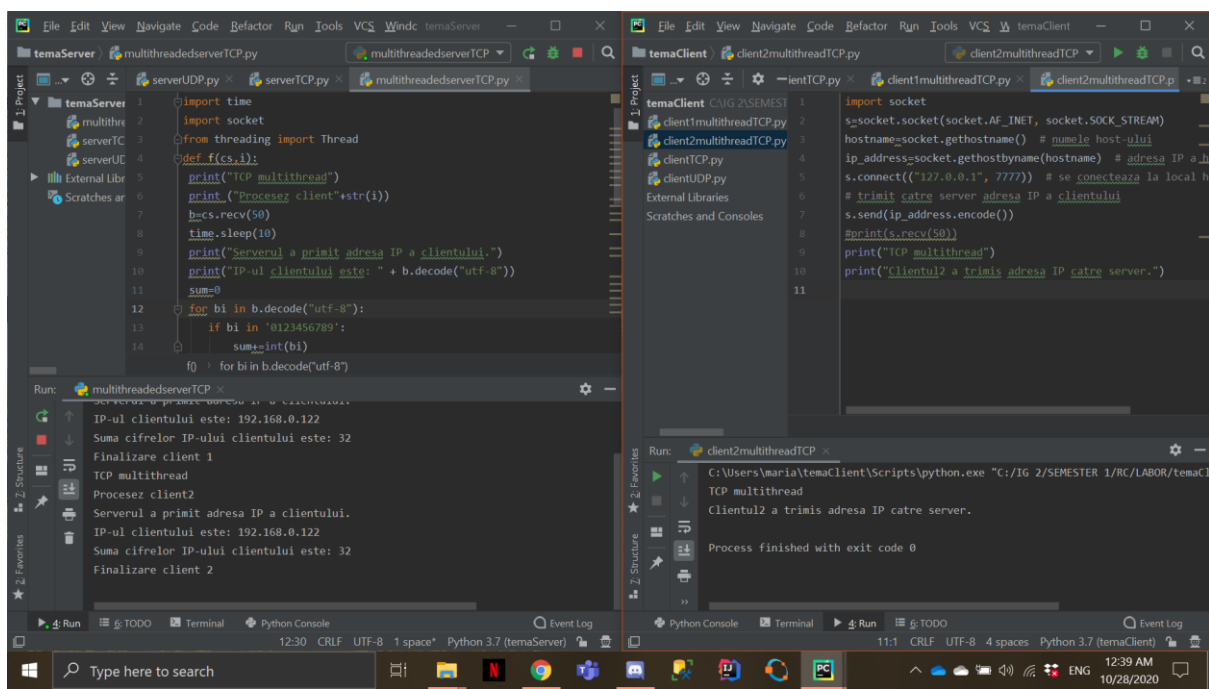
Run: client2multithreadTCP

C:\Users\maria\temaClient\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaClient/client2multithreadTCP.py"

TCP multithread

Clientul 1 a trimis adresa IP catre server.

Process finished with exit code 0



```
temaServer\multithreadedserverTCP.py
1 import time
2 import socket
3 from threading import Thread
4 def f(cs,i):
5     print("TCP multithread")
6     print("Procesez client"+str(i))
7     bcs.recv(50)
8     time.sleep(10)
9     print("Serverul a primit adresa IP a clientului.")
10    print("IP-ul clientului este: " + b.decode("utf-8"))
11    sum=0
12    for bi in b.decode("utf-8"):
13        if bi in '0123456789':
14            sum+=int(bi)
15    for bi in b.decode("utf-8")
```

```
temaClient\client2multithreadTCP.py
1 import socket
2 s=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 hostname=socket.gethostname() # numele host-ului
4 ip_address=socket.gethostbyname(hostname) # adresa IP a host-ului
5 s.connect(("127.0.0.1", 7777)) # se conecteaza la local host
6 # trimite catre server adresa IP a clientului
7 s.send(ip_address.encode())
8 #primeste de la server
9 s.recv(50)
10 print("TCP multithread")
11 print("Clientul 2 a trimis adresa IP catre server.")
```

Run: multithreadedserverTCP

C:\Users\maria\temaServer\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaServer/multithreadedserverTCP.py"

TCP multithread

Procesez client2

Serverul a primit adresa IP a clientului.

IP-ul clientului este: 192.168.0.122

Suma cifrelor IP-ului clientului este: 32

Finalizare client 2

Run: client2multithreadTCP

C:\Users\maria\temaClient\Scripts\python.exe "C:/IG 2/SEMESTER 1/RC/LABOR/temaClient/client2multithreadTCP.py"

TCP multithread

Clientul 2 a trimis adresa IP catre server.

Process finished with exit code 0

Intai se da RUN la server (care asteapta clientul), iar apoi se da RUN la client, iar cand serverul primeste IP-ul de la client, calculeaza suma cifrelor din IP si o intoarce. In cazul TCP-ului cu Multithread, se da RUN la Server si apoi se da RUN la fiecare client.