## KOLEGIJ

Mrežno programiranje

Laboratorijska vježba br.2



***Teme:***

* Socket

U ovom laboratorijskoj vježbi ćemo detaljnije proći pojam priključnica, tj. *socketa*.

# Osnove socketa

Do sada, kroz kolegij Računalne mreže i mrežne usluge smo prošli osnovne koncepte računalnih mreža, a s prvom laboratorijskom vježbom smo prošli osnovne karakteristike Python programskog jezika.

Da bi razumjeli što je socket, počet ćemo s Internet konekcijom. Internet konekcija spaja dvije točke, tj. dva čvora preko Interneta u svrhu razmjene podataka.

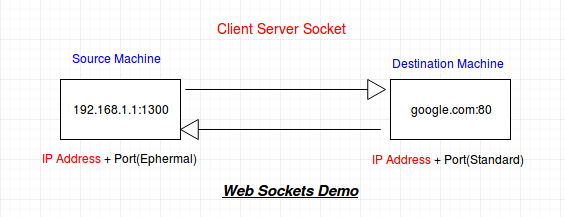
Jedan proces s računala PC1 može komunicirati s procesom na računalu

PC2 preko takve konekcije. Svojstva takve komunikacije su:

* Pouzdanost
* Point-to-point
* Full-duplex

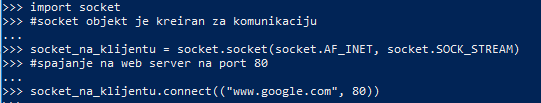
**Priključnice** ili **socketi** su krajnje točke bidirekcionalnog, point-to- point komunikacijskog kanala. Ako za primjer uzmemo konekciju između klijenta (web preglednik) i servera (npr. www.google.com), imat ćemo dvije priključnice. Klijentski socket i Serverski socket.

Socket se sastoji od dva dijela: **IP adresa + Port**



Generalno gledajući, naprimjer, svaki put kad odete na [www.google.com](http://www.google.com/),

browser napravi sljedeće:



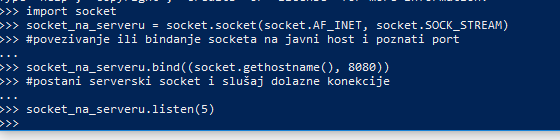
Ovo je napravljeno na klijentskoj strani. Kada se klijent pokuša spojiti na server, toj konekciji operacijski sustav dodijeli random port. Taj random port se naziva **Ephermal Port**. Na gornjoj ilustraciji, 1300 je ephermalni port na klijentu. Socket na klijentu je kratkog

„životnog“ vijeka, naprimjer, čim su podaci razmijenjeni on se

zatvori.

Na serveru se događa sličan, ali ipak malo različit proces u odnosu

na klijenta.



Navedeno možemo vidjeti i u izlistu TCP konekcija:



Za sada ćemo se fokusirati samo na *connect* i *bind* metode.

*Connect* koristi klijentski socket za uspostavljanje konekcije sa serverom. Taj zahtjev ispunjava *bind* na serverskom socketu.

Razlike između klijentskih i serverskih socketa:

* Za razliku od klijentskih socketa serverski nisu kratkotrajni. Naprimjer; možda vi trebate jednokratno ili s vremena na vrijeme pristup Aspirinoj web stranici, ali ta stranica mora biti dostupna cijelo vrijeme za bilo kojih zahtjev koji stigne od bilo kojeg korisnika
* Za razliku od klijentskih socketa koji koriste ephemeral port za ostvarivanje konekcije, serverski socket zahtijeva standardni ili dobro definirani port za konekciju kao što je naprimjer port

80 za normalnu HTTP konekciju.

# Socketi u Pythonu

Razlikujemo dva tipa socketa: SOCK\_STREAM i SOCK\_DGRAM.

|  |  |
| --- | --- |
| SOCK\_STREAM | SOCK\_DGRAM |
| ZA TCP PROTOKOL | ZA UDP PROTOKOL |
| POUZDANA ISPORUKA | NEPOUZDANA ISPORUKA |
| GARANTIRA ISPRAVAN REDOSLIJED  PAKETA | NE GARANTIRA REDOSLIJED |
| KONEKCIJSKI ORIJENTIRAN | NIJE KONEKCIJSKI ORIJENTIRAN |
| BIDIREKCIONALAN | NIJE BIDIREKCIONALAN |

Za kreiranje konekcije u Pytonu moramo koristiti socket.socket() funkciju koja je dostupna u Python socket modulu koji ima generalnu sintaksu sljedećeg oblika:

S = socket.socket(socket\_family, socket\_type, protocol=0)

* **socket\_family:** može biti ili AF\_UNIX ili AF\_INET. Govoriti ćemo samo o INET socketima pošto oni zauzimaju više od 99% upotrebe.
* **socket\_type:** može biti ili SOCK\_STREAM ili SOCK\_DGRAM.
* **protocol:** obično se izostavi što rezultira defaultiranjem na 0

# Socket metode

### connect()

Klijentska metoda koji se koristi za spajanje na udaljeni socket na nekoj adresi. Adresni format (host, port) se koristi za AF\_INET adrese.

### bind()

Serverska metoda koja binda socket na adresu. Format adrese

također ovisi o AF\_INET adresama.

### listen(backlog)

Serverska metoda koja sluša konekcije koje se spajaju na socket. Backlog je maksimalan broj konekcija u redu čekanja koje moraju biti poslušane prije odbijanja konekcije.

### accept()

Serverska metoda koja se koristi za prihvaćanje konekcije. Socket mora biti bindan na adresu i mora slušati konekcije. Return vrijednost je pair(conn, address) gdje je *conn* novi socket objekt koji može biti iskorišten za slanje i primanje podataka na toj konekciji, a adresa je adresa bindana na socket na drugom kraju konekcije.

**;**za detaljan opis metoda pogledajte Python dokumentaciju:

[**https://docs.python.org/2/library/socket.html**](https://docs.python.org/2/library/socket.html)

# RAD SA SOCKETIMA

Napraviti ćemo jednostavan klijent-server program.

Pripremite sljedeći program i pokrenite ga u Powershellu ili cmd-u. Po pokretanju program neće raditi u tom trenutku ništa osim što će osluškivati dolazne konekcije. Slično tome, svaka web stranica koju posjetite ima server na kojem je hostana, a koji uvijek čeka klijente da se spoje, tj. osluškuje konekcije.

#tcp\_server.py

**import** socket

server\_socket **=** socket**.**socket**()** host **=** socket**.**gethostname**()** port **=** 9999

server\_socket**.**bind**((**host**,**port**))**

**print** "Waiting for connection..." server\_socket**.**listen**(**5**)**

**while True:**

conn**,**addr **=** server\_socket**.**accept**() print** 'Got Connection from'**,** addr conn**.**send**(**'Server Saying Hi'**)** conn**.**close**()**

Niže se nalazi program za klijenta. Klijent se pokušava spojiti na

serverski port (9999).

**Pitanje**: što radi linija koda client\_socket**.**connect**((**host**,**port**))?**

### Odgovor: Povezuje klijenta na poslužitelj, na port 9999. (No connection could be made because the target machine actively refused it).

#tcp\_client.py

**import** socket

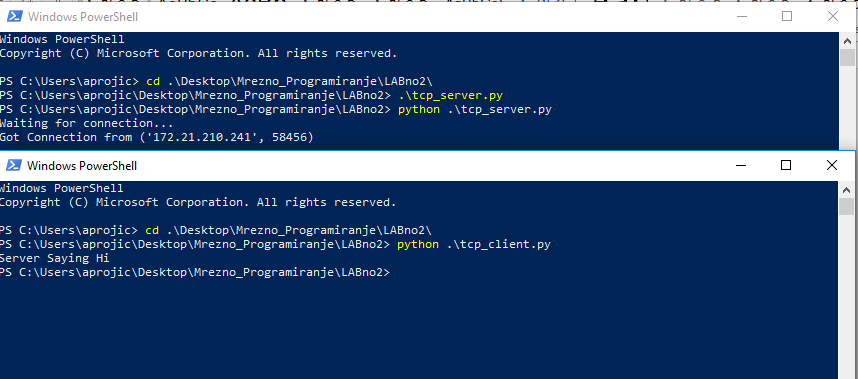
client\_socket **=** socket**.**socket**()** host **=** socket**.**gethostname**()** port **=** 9999

client\_socket**.**connect**((**host**,**port**)) print** client\_socket**.**recv**(**1024**)**

client\_socket**.**close**()**

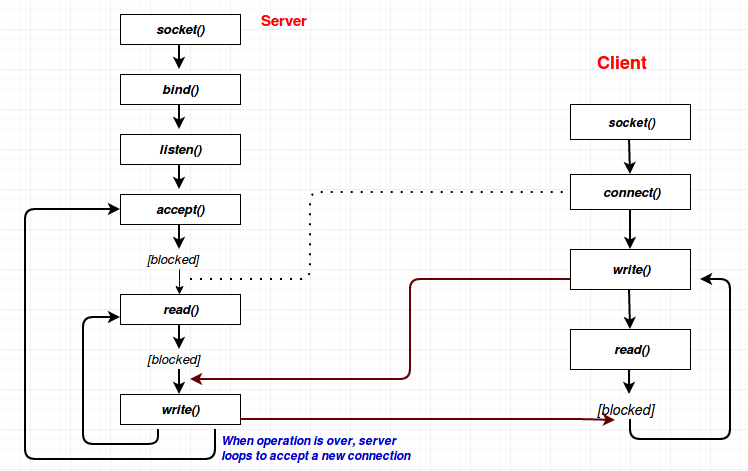
U dva odvojena Powershella pokrenite oba programa. Ako je sve u redu

trebali bi dobiti sljedeće:



**Pitanje**: U serverskom dijelu vašeg programa, što znači drugi broj u zagradama kod ispisa „Got connection…“**?**

### Odgovor: Port

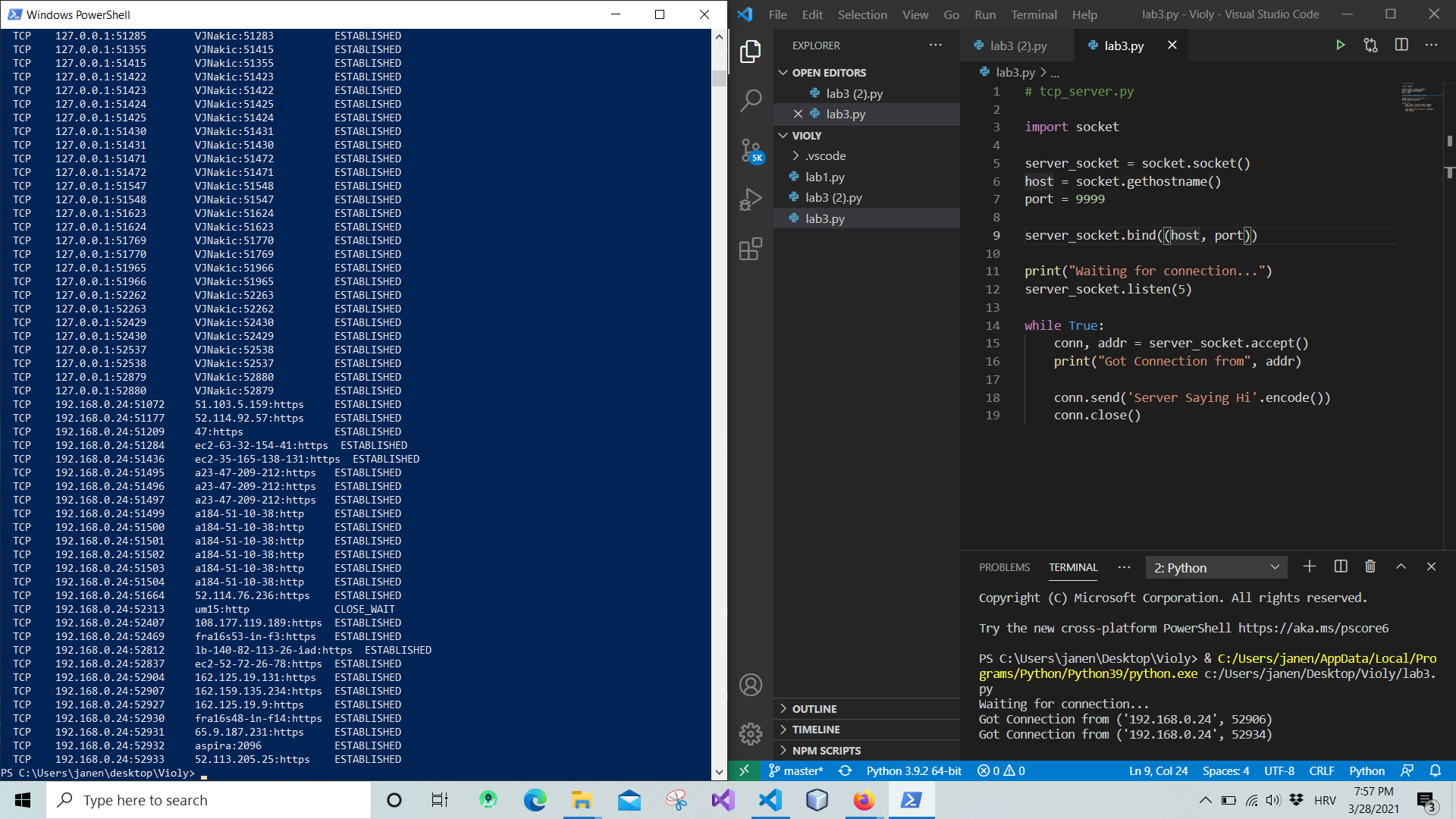


**ZAKLJUČNO**

Kreirajte .py skripte s navedenim kodom te ih pokrenite. Dostavite kod

i odgovore za pitanja unutar laboratorijske vježbe (2 su).

Priložite izlist aktivnih konekcija na koji god način hoćete (netstat, tcpconn, itd..) gdje se vidi na kojem portu vaš serverski program sluša.



Modificirajte tcp\_client program na način da otvorite socket konekciju

prema [www.google.com](http://www.google.com/) i priložite kod. Ispis može biti sličan ovome:

