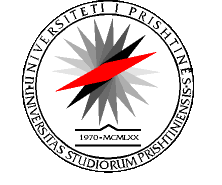
UNIVERSITETI I PRISHTINËS

FAKULTETI I INXHINIERISË ELEKTRIKE DHE KOMPJUTERIKE

DEPARTAMENTI I INXHINIERISË KOMPJUTERIKE



**Lënda: Rrjeta Kompjuterike**

**Raporti për projektin e pare**

**Titulli i projektit: Dizajnimi Klient-Server**

Profesori dhe asistenti i lëndës: Emri i studenti:

Prof. Dr. Blerim Rexha Bleona Zenelaj

MSc. Haxhi Lajqi ID: 160714100084

29.05.2019

Prishtinë

Hyrje

Synimi i këtij projekti ka qenë që të jepet kuptimi i komunikimit në mes klientit dhe serverit. Kërkesa ka qenë që krijohen programet përkatëse për implementimin e serverit dhe klientit të cilët do të komunikojnë në mes vete kur shkëmbejnë të njejtin port. Komunikimi i tyre është mundësuar përmes soketave dhe implementimi është bërë përmes gjuhës së lartë programuese Python, gjuhë kjo që thjeshtëson problemin për nga volumi i kodit që duhet të shkruhet meqë përmban shumë librari dhe funksione të gatshme (BuiltIn). Programi është zhvilluar duke përdorur veglat e platformës Visual Studio 2017 e cila e mbështetë gjuhën Python dhe është mjaft e përshtatshme për përdorim. Katër programet e zhvilluara për TCP Server, TCP Klient, UDP Server, UDP Klient janë funksionale bashkë me metodat e implementuara.

Përshkrimi i projektit dhe funksionalitetit të tij

Ideja e projektit në fjalë ka qenë që të krijohet një server me emrin FIEK i cili përmban funksione që plotësojnë disa kërkesa të klientëve që konektohen me të. Si kërkesë ka qenë zhvillimi i kodin në dy versione, me protokollin TCP dhe me atë UDP. Dihet se gjatë komunikimit në mes klientit dhe serverit, protokolli TCP ofron garancë që paketat e dërguara do të mbërrijnë në cak apo nëse nuk mbërrijnë dërguesi dhe marrësi, në këtë rast klienti dhe serveri, do të njoftohen për arsyet. Përkundër kësaj, kur klienti dhe serveri komunikojnë përmes protokollit UDP nuk dihet me siguri nëse paketat e dërguara nëpër port komunikues mbërrijnë në cak apo humbin gjatë rrugës apo edhe nëse humb radha e dërgimit. Procedura e kodimit është e ngjashme vetëm se gjatë komunikimit me TCP krijohet kanal komunikues në mes klientit dhe serverit, ndërsa gjatë komunikimit me UDP serveri gjatë cilësdo përgjigje kërkon adresën e klientit përkatës.

Kur klienti lidhet me serverin atij i paraqitet një menu me funksionet që serveri i ofron. Klientit i shfaqen disa qifte funksion-përshkrim ku specifikohet mënyra e paraqitjes së kërkesës dhe përgjigja përkatëse e serverit për kërkesën e zgjedhur. Jam munduar që të parashikoj gabimet e mundshme gjatë shtrimit të kërkesave dhe të ju përshkruaj klientëve në mënyrë sa më të kuptueshme se si duhet të veprojnë për të marrë përgjigjet që ata kërkojnë. Funksionet e disponueshme të serverit që kam zhvilluar janë:

1. IPADRESA – Funksion pa parametra hyrës që kthen si përgjigje IP adresën e klientit në formë dhjetore, p.sh. (10.10.7.251).
2. NUMRIIPORTIT – Funksion pa parametra hyrës që kthen numrin e portit të klientit (Është numër që specifikon portin që soketi ka shfrytësuar për të mundësuar lidhjen përkatëse. Gjenerohet sa herë që krijohet lidhje e re.)
3. BASHKETINGELLORE – Funksion që si parametër hyrës shfrytëzon teksin e dhënë nga klienti përmes inputit, numëron bashketingelloret në atë tekst dhe kthen si përgjigje numrin e tyre.
4. PRINTIMI – Funksion që si parametër hyrës shfrytëzon tekstin e dhënë nga klienti përmes inputit. Ky tekst formatohet duke i larguar hapësirat e panevojshme në fillim dhe fund të tekstit duke kursyer kështu hapësirën memorike, dhe i kthehet shfrytëzuesit si përgjigje nga serveri.
5. EMRIIKOMPJUTERIT– Funksion pa parametër hyrës që kthen si përgjigje emrin e host-it të klientit.
6. KOHA– Funksion pa parametra hyrës që kthen si përgjigje kohën reale të serverit si format te lexueshem.
7. LOJA – Funksion pa parametra hyrës që kthen si përgjigje 7 numra random të gjeneruar nga serveri për klientin në rangun e numrave nga 1 deri në 49.
8. FIBONACCI – Funksion me parametër hyrës që kthen si përgjigje numrin e serisë Fibonacci ku parametri i dhënë nga klienti përmes inputit paraqet indeksin përkatës të numrit të kësaj serie.
9. KONVERTO – Funksion që mundëson konvertimin e temperaturës nëpër njësi

* Nga Kilowatt në Horsepower përmes opsionit KilowattToHorsepower
* Nga Horsepower në Kilowatt përmes opsionit HorsepowerToKilowatt
* Nga Degrees në Radians përmes opsionit DegreesToRadians
* Nga Radians në Degrees përmes opsionit RadiansToDegrees
* Nga Gallons në Liters përmes opsionit GallonsToLiters
* Nga Liters në Gallons përmes opsionit LitersToGallons.

Poashtu janë të disponueshme edhe dy funksione shtesë:

1. TEMPKONVERTO – Funksion që mundëson konvertimin e numrit nga Kelvin në Celsius dhe e anasjellta. (Është fjala për njesite matese te temperatures.)
2. RRETHI – Funksion që kthen si përgjigje perimetrin dhe sipërfaqen e rrethit me rreze të specifikuar nga klienti përmes inputit.

Klientit i mundësohet që të mbyllë lidhjen me serverin kurdo që ai dëshiron. Klienti do të njoftohet nga serveri kurdo që nuk e specifikon mirë kërkesën si dhe do të njoftohet saktësisht se për cilën pjesë të kërkesës bëhet fjalë. Me raste serveri do të japë sqarime për klientin se çfarë tipi duhet të kenë parametrat që ai jep përmes inputit. Sfidë për serverin ka qenë që të jetë në gjendje të pranojë kërkesa të njëpasnjëshme nga klientët. Kjo është mundësuar përmes të ashtuquajturve thread-a. Lidhja mund të ndërpritet përmes komandës që është specifikuar për të kryer shërbimin e mbylljes “C” që qëndron për “Close”. Serveri është në gjendje të lidhet me më shumë se një klient dhe tu përgjigjet atyre në mënyrë të pavarur.

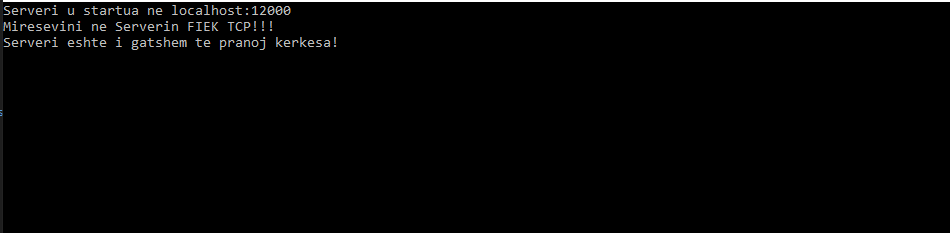
Kodi dhe rezultatet e arritura

Pamjet në vazhdim paraqesin implementimin e programit për komunikimin Klient-Server sipas protokollit TCP (Transmission Control Protocol).

Kodi për serverin:

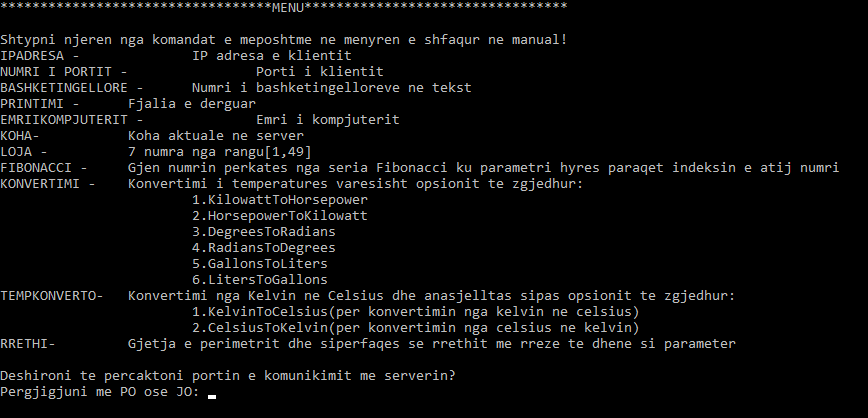


Pamja që paraqitet pas ekzekutimit të serverit:



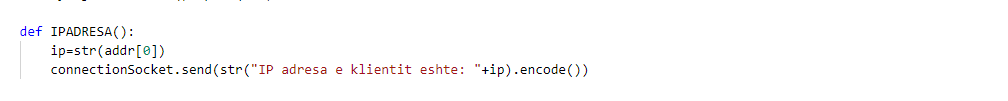
Kodi për klientin:

Pamja që paraqitet pas ekzekutimit të klientit:

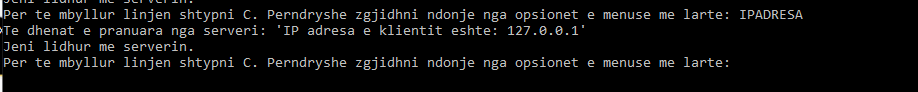


Kodi për funksionet e implementuara në kuadër të FIEK TCP Serverit është paraqitur më poshtë.

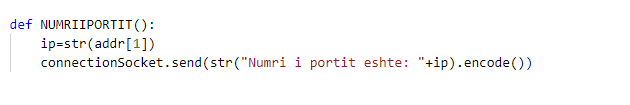
**Funksioni IPADRESA**:



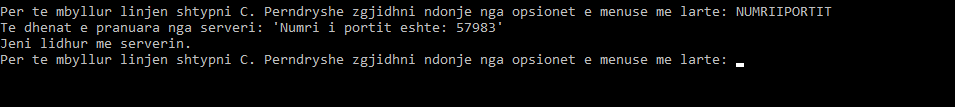
Rezultati që fitohet kur pas ekzekutimit:



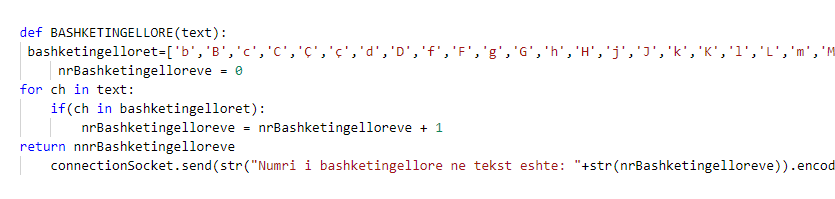
**Funksioni NUMRIIPORTIT**:



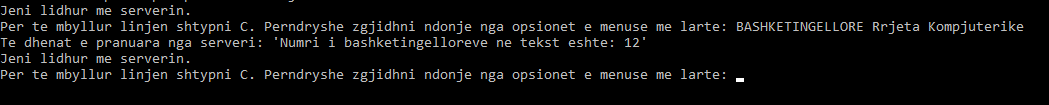
Rezultati që fitohet kur pas ekzekutimit:



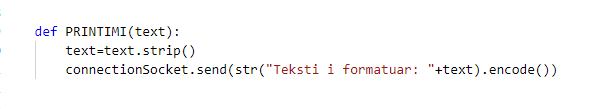
**Funksioni BASHKETINGELLORE**:



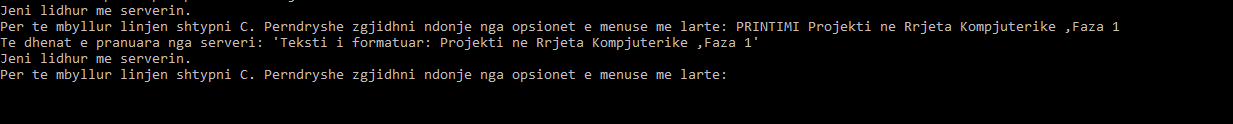
Rezultati që fitohet kur pas ekzekutimit:



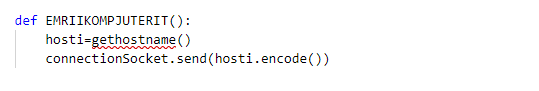
**Funksioni PRINTIMI**:



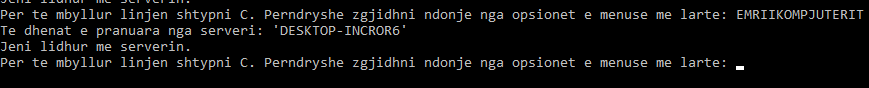
Rezultati që fitohet kur pas ekzekutimit:



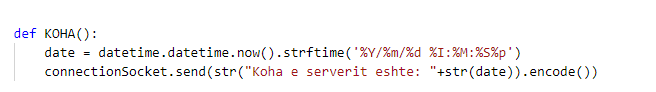
**Funksioni EMRIIKOMPJUTERIT**:



Rezultati që fitohet kur pas ekzekutimit:

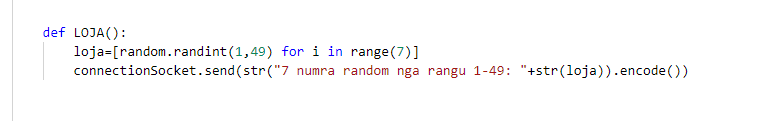


**Funksioni KOHA**:

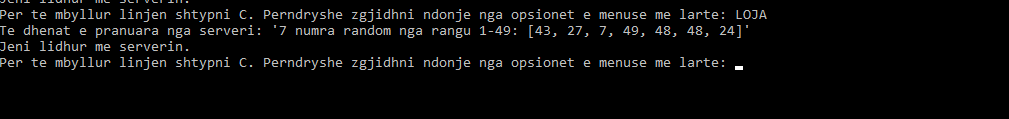


Rezultati që fitohet kur pas ekzekutimit:

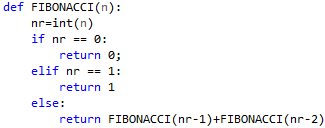
**Funksioni LOJA**:



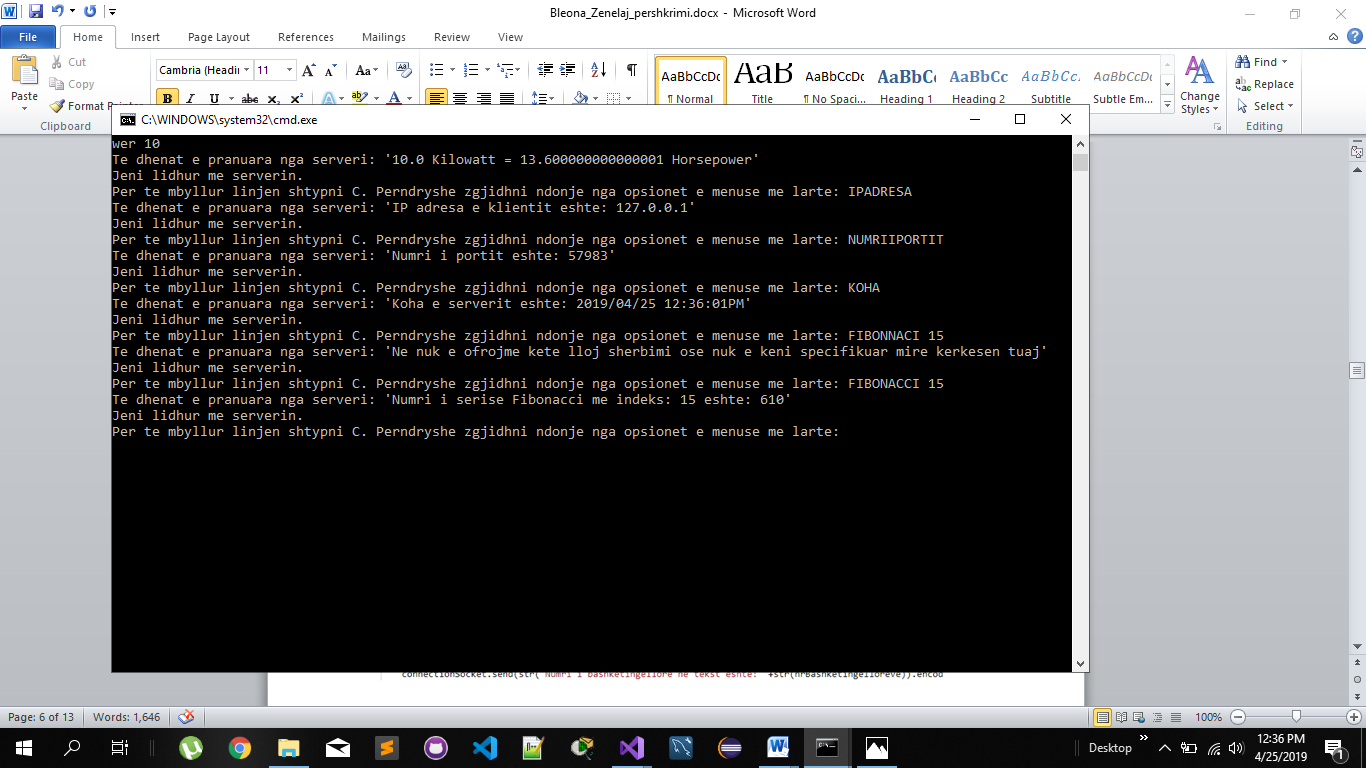
Rezultati që fitohet kur pas ekzekutimit:



**Funksioni FIBONACCI**:

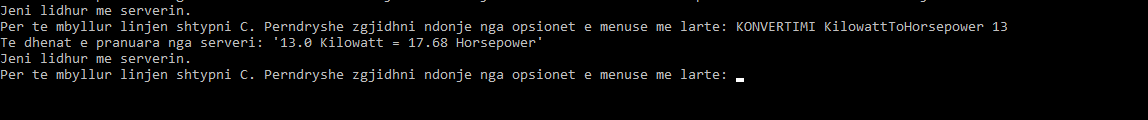


Rezultati që fitohet kur pas ekzekutimit:

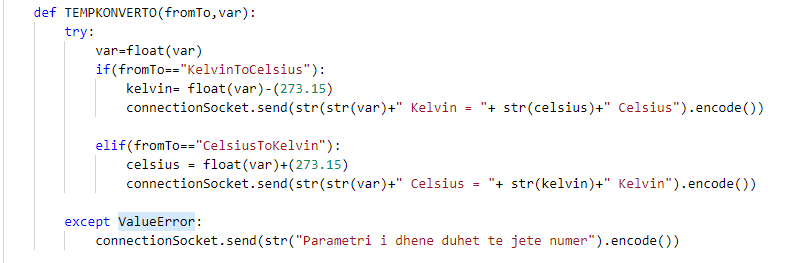


**Funksioni KONVERTIMI**:

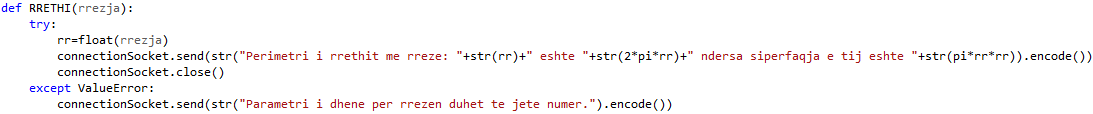
Rezultati që fitohet kur pas ekzekutimit:



Funksionet shtesë janë paraqitur më poshtë bashkë me rezultatet që kthejnë kur ekzekutohen.

**Funksioni TEMPKONVERTO**:

**Funksioni RRETHI**:



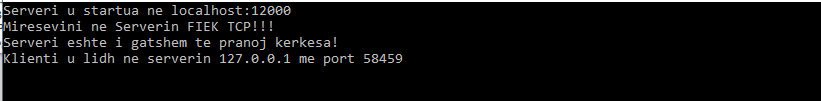
Rezultati që fitohet kur pas ekzekutimit:

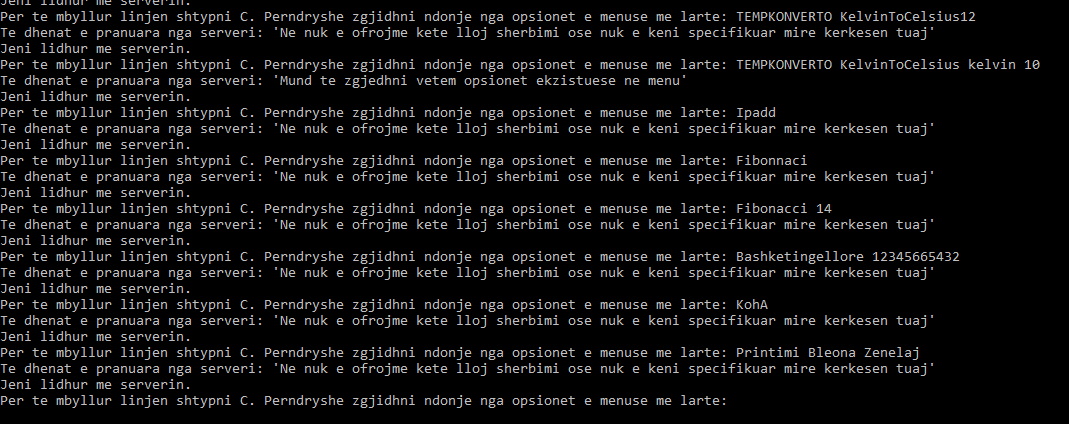


Shfrytëzimi i “C” për të mbyllur lidhjen:

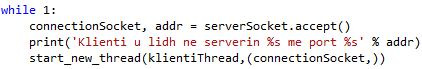


Shkronja “C” në këtë rast qëndron për “Close”.

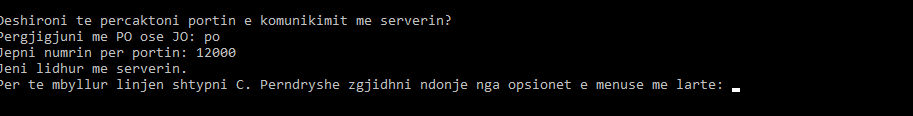
Sa herë që lidhet një klient i ri do të krijohet një port i ri përkatësisht një soket i ri për komunikim. Nga ana e serverit është e mundur të shihet numri i atij porti përderisa serveri është në funksion. Pamja më poshtë tregon se qka shihet në konzolën e serverit.

Raste të ndryshme të gabimeve gjatë shtrimit të kërkesave:

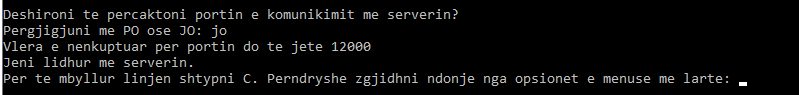
Shumë klientë mund të lidhen në të njejtën kohë me serverin i cili u kthen atyre përgjigje përmes multithreading.



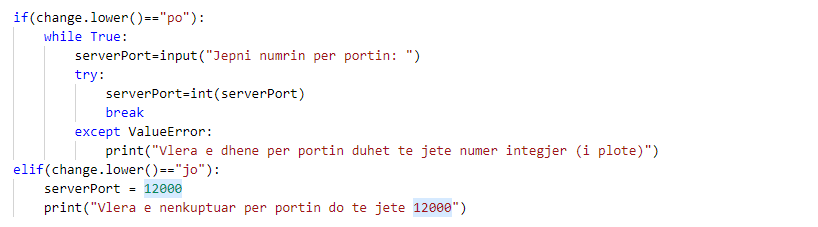
Në fillim të ekzekutimit të programit nga ana e klientit atij do ti parashtrohet pyetja nëse dëshiron ta përcaktojë vetë portin e serverit me të cilin dëshiron të lidhet. Nëse përgjigjet me PO ai do të japë si input numrin e portit. Nëse ai nuk shkruan numër të plotë do të ketë mundësi të përsërisë shkruarjen.



Nëse klienti pëgjigjet me JO, numri i nënkuptuar i portit do të jetë 12000 që është numri i portit të serverit FIEK. Kjo është paraqitur në pamjen në vazhdim:



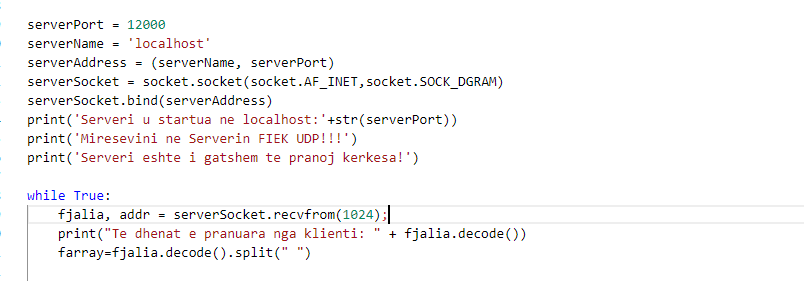
Kodi i shkruar për këtë pjesë:



Sa i përket kodimit në protokollin UDP procedura e funksioneve është tërësisht e njejtë ashtu si edhe mënyra e thirrjes së tyre nga ana e klientit si dhe përgjigjet që kthen serveri. Disa dallime të vogla ekzistojnë në formën e komunikimit pra në mënyrën sesi pranohen dhe dërgohen kërkesat.

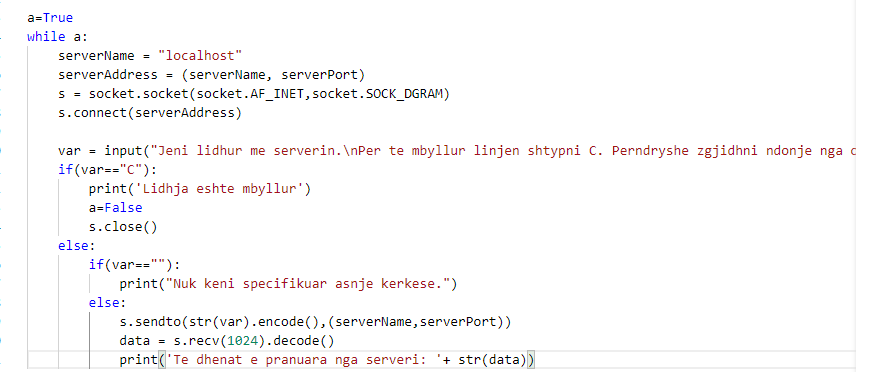
Pamjet në vazhdim paraqesin implementimin e programit për komunikimin Klient-Server sipas protokollit UDP (User Datagram Protocol).

Një pjesë e programit i serverit (aty ku vërehen dallimet përkrah TCP serverit):



Shihet se dallimet ekzistojnë te funksioni socket.SOCK\_DGRAM duke ditur se serveri në TCP përdorë socket.SOCK\_STREAM. Të gjitha metodat e implementuara në UDP serverin si edhe në TCP serverin funksionojnë dhe japin rezultate të sakta. Poashtu është e mundshmë lidhja e disa klientëve me serverin në të njejtën kohë, mbyllja e programit përmes “C” dhe shumë funksionalitete tjera të përshkruara më lartë për TCP protokollin.

Një pjesë e programit të serverit është shfaqur më poshtë në mënyrë që të shihet se ku qndrojnë dallimet e UDP Klientit me TCP Klientin.



Konkluzionet

Komunikimi në mes klientëve dhe serverëve për mua ka qenë një abstraksion para zhvillimit të këtij projekti. Kjo ngase gjatë tërë kohës kam qenë vetë kliente dhe përmes makinave të ndryshme kërkuese (search engines) kam bërë kërkesa nga më të ndryshmet mirëpo asnjëherë më parë se kam ditur se çka ndodhë me saktësi kur unë shkruaj një pyetje për Google apo kudo tjetër. Tashmë kam kuptuar konceptet themelore të mënyrës së komunikimit përmes mjeteve të ndryshme komunikuese dhe procesi që ndodhë në fakt është mjaft kompleks.

Qëllimet e mia mbi projektin dhe kërkesat e parashtruara janë plotësuar duke përdorur të dhëna të ndryshme nga interneti dhe duke i kombinuar ato me idetë e mia, kjo sepse nuk kam pasur njohuri të mëparshme për gjuhën Python apo për konceptin e soketave. Kam shfrytëzuar njohuritë e mëparshme në gjuhë tjera programuese (më shumë nga Java,C++ e më pak nga C#), të marra këto nga ligjëratat që kam ndjekur.  
 Në rastin konkret falënderime të veçanta shkojnë për profesorin e lëndës Rrjeta Kompjuterike, Prof.Dr. Blerim Rexha si dhe asistentin e lëndës Msc. Haxhi Lajqi meqë këta të fundit më kanë dhënë konceptet bazë që më ndihmuan në realizimin e këtij projekti.

Referencat

Disa nga uebfaqet që më kanë ndihmuar të plotësoj kërkesat e projektit:

https://www.stackoverflow.com/

https://wiki.python.org/moin/UdpCommunication

http://net-informations.com/python/net/thread.htm

https:// www.jquery-az.com/hoë-to-remove-ëhitespaces-in-strings-of-python/

<https://tutorialedge.net/python/>

<http://www.techbeamers.com/python-multithreading-concepts/>

https://docs.python.org/2/library/datetime.html