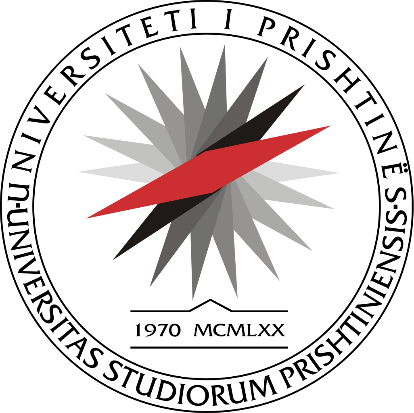
UNIVERSITETI I PRISHTINËS “HASAN PRISHTINA”

FAKULTETI I INXHINIERISË ELEKTRIKE DHE KOMPJUTERIKE

DEPARTAMENTI:INXHINIERI KOMPJUTERIKE



**PROGRAMIMI ME SOKETA**

***Lënda:”Rrjeta Kompjuterike”***

Profesori: Studenti:

Prof. Dr. Asoc. Blerim Rexha Muhamed Zeqiri

Ass. Msc. Haxhi Lajqi

Prishtinë ,Prill 2019

Përmbajtja

[**HYRJE** 3](#_Toc6509101)

[**VEGLAT E PËRDORURA GJATË ZHVILLIMIT TË PROJEKTIT** 3](#_Toc6509102)

[**PROTOKOLLET TCP DHE UDP.NGJASHMERITË DHE DALLIMET** 4](#_Toc6509103)

[**ZHVILLIMI I SERVERIT-KLIENTIT SIPAS PROTOKOLLIT TCP. KARAKTERISTIKAT KRYESORE** 5](#_Toc6509104)

[**OPERACIONET E DOMOSDOSHME TË SERVERIT** 7](#_Toc6509105)

[**IPADRESA DHE NUMRIIPORTIT** 7](#_Toc6509106)

[**PRINTIMI** 8](#_Toc6509107)

[**BASHKETINGELLORE** 8](#_Toc6509108)

[**KOHA** 9](#_Toc6509109)

[**LOJA** 10](#_Toc6509110)

[**FIBONACCI** 11](#_Toc6509111)

[**EMRIIKOMPJUTERIT** 12](#_Toc6509112)

[**KONVERTO** 12](#_Toc6509113)

[**OPERACIONET SHTESË TË SERVERIT** 14](#_Toc6509114)

[**SHUMA** 14](#_Toc6509115)

[**SHUMABIN** 14](#_Toc6509116)

[**CAP** 15](#_Toc6509117)

[**SHMVP** 15](#_Toc6509118)

[**IMPLEMENTIMI I MULTITHREAD.SERVERI PRANON NË TË NJËJTËN KOHË SHUMË KËRKESA NGA SHUMË KLIENTË.** 16](#_Toc6509119)

[**ZHVILLIMI I SERVERIT-KLIENTIT SIPAS PROTOKOLLIT UDP** 20](#_Toc6509120)

[**PËRFUNDIME** 22](#_Toc6509121)

# **HYRJE**

Komunikimi ne distance gjithmone ka qene sfide.Por me zhvillimin teknologjik dhe konceptin e “rrjetave” kjo eshte bere shume mire e realizueshme permes bartjes se te dhenave nga serveri deri ne pikat fundore (klientet) . Cdo e dhene qe dergohet nga serveri (si pasoje e nje kerkese nga klienti) kalon nje rruge te caktuar deri te arrije te caku.Edhe ne web edhe ne aplikacione mobile apo desktop apo cdo sistem qe te dhenat i merr nga nje databaze kerkon qe te kete komunikim me serverin ku ruhen ato te dhena. Pra klienti ka detyre te dergoje kerkesa dhe te marre pergjigje.

Keshtu edhe ne kete projekt si detyre kemi pasur zhvillimin e serverit dhe nje klientit testues per serverin, por ne dy protokolle te ndryshme. TCP[[1]](#footnote-1) dhe UDP[[2]](#footnote-2).

Për realizimin e projektit janë përdorur te ashtuquajturat ‘Sockets’ të cilat paraqesin pika fundore të komunikimit në mes dy programeve në rrjetë. Soketa lidhet për IP adresën dhe portin ashtu që TCP të identifikojë se kujt do i dërgohen te dhenat.

## **VEGLAT E PËRDORURA GJATË ZHVILLIMIT TË PROJEKTIT**

Zhvillimi dhe testimi i funksionalitetit të programeve ne fjalë (Server/Client në TCP/UDP) janë bërë në:

-Sistemi Operativ: Microsoft Windows 10 Professional

-Gjuha Programuese: Python

-Konzollë/GUI: Konzollë

-IDE [[3]](#footnote-3):JetBrains PyCharm Community Edition 2018.3.5

-Interpreter:Python 3.6

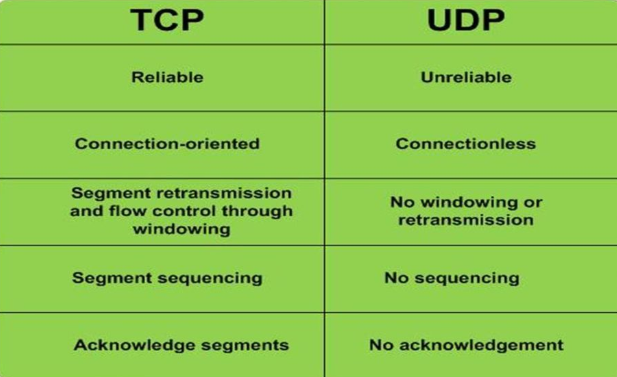
Në vijim do të keni në detaje të demonstruar implementimin në kod të projektit sipas protokollit ‘FIEK’ si dhe vetitë apo funksionalitetet shtesë.

# **PROTOKOLLET TCP DHE UDP.NGJASHMERITË DHE DALLIMET**

TCP është protokoll i cili kërkon që të krijohet një lidhje para se dy paisjet që komunikojnë të transmetojnë të dhëna. Pra është protokoll i orientuar ne koneksion(lidhje). Gjithashtu duhet te mbyllet lidhja pas transmetimit të të dhënave.Kështu mund të themi se TCP është shumë i besueshëm sepse gjithmonë duke pyetur derguesi A marrësin B se a i ke pranuar paketat [[4]](#footnote-4)e te dhenave garantohet që cdo e dhënë të arrijë te B dhe paketat të jenë te renditura si duhet(sekuencimi).

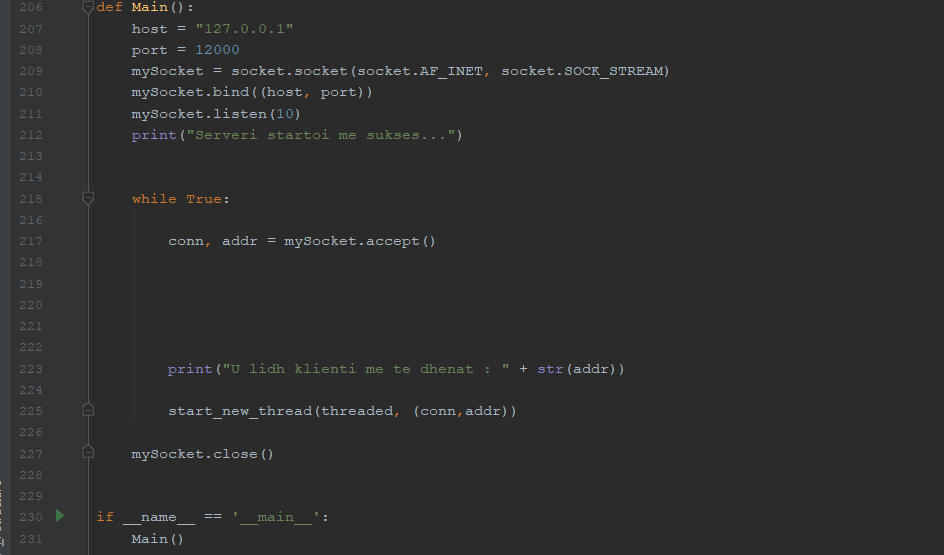
UDP ndryshe nga TCP nuk është i orientuar në koneksion por në datagram. Nuk është i besueshëm ngase derguesi A vazhdimisht dërgon të dhëna pa marrur konfirumimn se a i ka pranuar apo jo marrësi B. Andaj si protokoll i tillë është i përshtatshme për broadcasting [[5]](#footnote-5), pamje në kohë reale si psh sporte etj.

Më poshtë kemi edhe disa karakteristika të tjera.



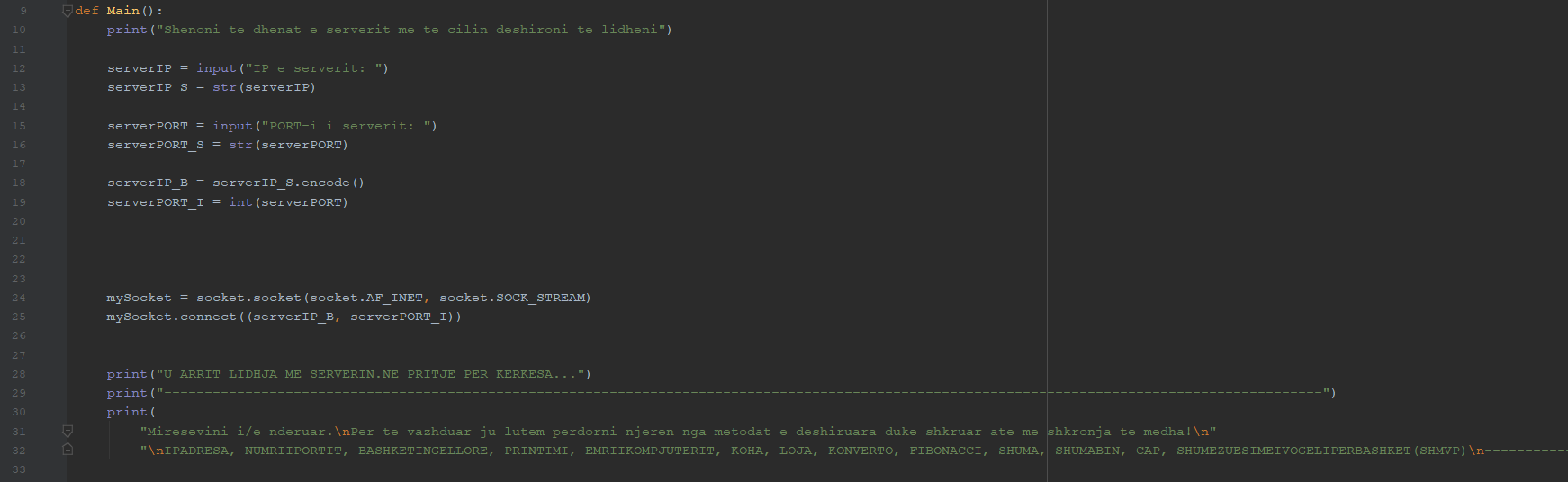
# **ZHVILLIMI I SERVERIT-KLIENTIT SIPAS PROTOKOLLIT TCP. KARAKTERISTIKAT KRYESORE**

Për të krijuar serverin na duhet një soketë dhe të deklarohet hosti(IP) dhe porti ne te cilin do të operojë.Dhe pastaj të lidhet apo behet bind soketa per tyre.Për këtë arsye së pari duhet importuar moduli per soketa përmes komandës ‘import socket’,e mandej prej saj krijojmë nje objekt soketë.



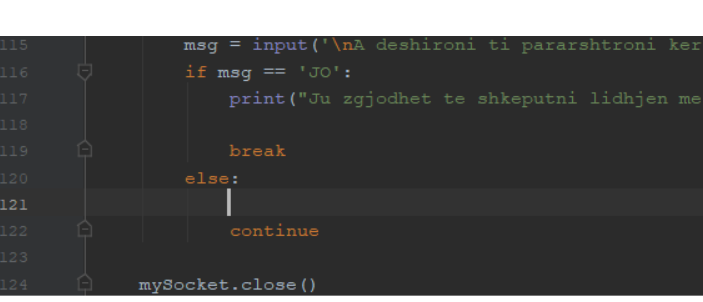
Për tja lehtesuar punën vetes dhe per organizim sa më kompakt te kodit preferohet që të deklarohet nje funksion që do të permbajë kodin për krijimin e soketës dhe konfigurimin e lidhjes qoftë sipas TCP apo UDP dhe më pas ky funksion të thirret sapo te startojë programi ,shih kodin nga rreshti 230-231

-Soketës në rastin konkret ‘mySocket’ duhet dhënë dy parametra. Parametri i parë paraqet familjen e IP adresave që perkah.Si e nenkuptuar është IPv4 (AF\_INET) dhe parametri i dyte paraqet tipin e soketës që meqenëse kemi te bejme me protokollin tcp atëherë ky është SOCK\_STREAM.Me metoden listen() caktojmë se deri në sa klientë mund të qëndrojnë në pritje për të dhëna nga serveri. Në rastin tim i 11ti klient nuk mund të marrë shërbime nga serveri sepse maksimalisht 10 koneksione mund t’i pranojë.

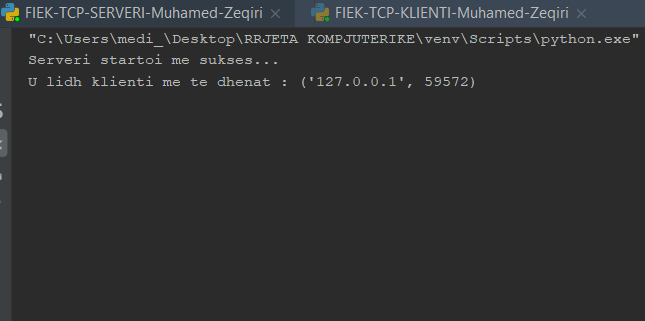
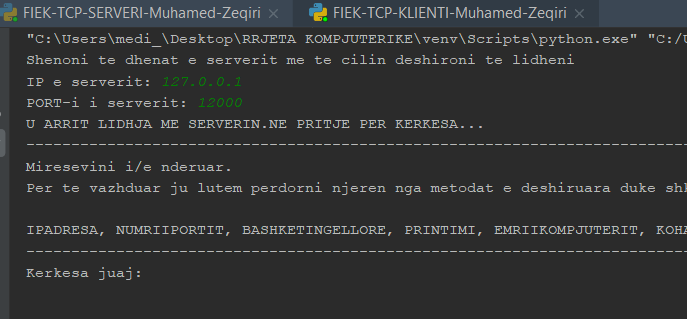
-Në anën tjetër edhe tek klienti definohet soketa per komunikim.Sikur ne pamjen me poshtë

Mirëpo unë tek klienti siç shihet më lartë e kam zhvilluar në atë formë që qysh ne startim të programit të kerkohet nga klienti të shtyp vetë IP dhe PORT të serverit me të cilin dëshiron të lidhet.variablat serverIP dhe serverPORT ruajnë cka shtyp klienti dhe te metoda connect() e soketës duhet kemi kujdes sepse serverIP duhet konvertuar në string permes str() dhe serverPORT ne int permes int(). Gjatë analizimit të projektit(kodit) mund të vëreni se çdo input në shumicen e rasteve e kam shndrruar së pari në string e më pas ne tipe tjera nëse më është dashur. Andaj emërtimet e variablave me fundoren ‘\_I’ apo ‘\_S’ qendrojne per integjer dhe string.

-Perdorimi i unazes while true mund te shihet ne disa raste, dhe kjo për qëllim që të mbajë ate pjesë të kodit gjithnjë aktiv. Psh tek klienti përpos unazes në fund pyesim nëse dëshiron të vazhdojë të dergojë kërkesë tjetër serverit.Nëse jo atëherë mbyllim soketin dhe klienti shkëputet nga serveri.



Ja nje demostrim i strukturës themelore të serverit dhe klientit deri më tani duke bere lidhjen me ‘127.0.0.1’(localhost). Kujdes! Për IP shtypni vetem IP valide sepse nuk ka validim.



## **OPERACIONET E DOMOSDOSHME TË SERVERIT**

Sipas ‘FIEK’ protokollit, serveri duhet ti ketë domosdoshmërisht këto operacione apo metoda:

-IPADRESA :Kthen si rezultat te klienti IP e tij.

-NUMRIIPORTIT:Kthen si rezultat PORT-in në të cilin është lidhur klienti për komunikim me serverin

-PRINTIMI:Thjesht kthen atë çka jepet si input.

-BASHKETINGELLORE:Për nje tekst të caktuar kthen numrin e bashkëtingelloreve.

-KOHA: kthen kohën e serverit në format të lexueshem ( psh dd-mm-yyyy hh:mm:ss).

-LOJA: Kthen si rezultat 7 numra te rastit te intervalit 7 deri 49.

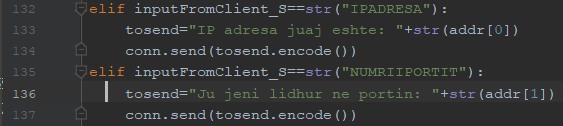
-FIBONACCI: Llogarit sekuencën fibonacci varësisht cili anëtar jepet si input nga klienti.

-EMRIIKOMPJUTERIT: Kthen tek klienti emrin e hostit/serverit.

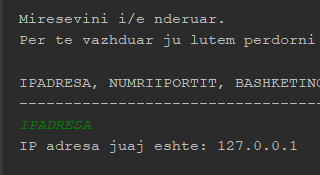
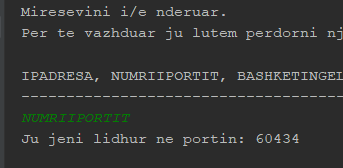
-KONVERTO:Varësisht se cili opsion zgjedhet klientit i kthehet një vlerë numerike ekuivalente me njësinë e madhësive gjegjëse.

### **IPADRESA DHE NUMRIIPORTIT**

Këto dy operacione janë realizuar duke shfrytezuar variablen ‘addr’ e cila pranon koneksionin. Realisht kjo variabel (permbajtja e saj) është listë dy anëtarëshe ku anëtari i parë ‘addr[0]’ paraqet IP adresën kurse anëtari i dytë ‘addr[1]’ portin në të cilin lidhet klienti.

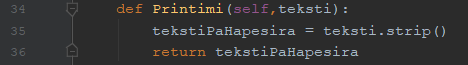
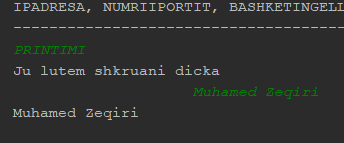


Me ngjyrë të gjelbër paraqitet inputi i klientit kurse përgjigja menjëherë pas inputit.



### **PRINTIMI**

Ky operacion realizohet përmes përdorimit të metodës ‘strip()’ e cila stringut qe e merr nga klienti si input ja largon hapesirat në fillim dhe në fund.Për çështje estetike dhe më praktike kam definuar funksionin ‘Printimi()’ ku ne vete kthen fjaline qe perpunohet nga strip().Kjo metodë nuk largon hapesirat mes fjalëve ne string. Por merret tërë fjalia dhe trajtohet si e tillë.



### **BASHKETINGELLORE**

Problemin për numërimin e bashkëtingëlloreve në një fjali e kam zgjidhur kështu:

-Duke ditur se ka shumë më shumë bashkëtingellore në alfabet ( shqip apo anglisht) sesa zanore,atëherë vendosim që ti numërojmë zanoret.Atë e bejmë duke numëruar karakter dhe si numrator perdorim variablen ‘nrzanoreve’.

-Pavarsisht se a na vjen teksti me shktronja të mëdha,të vogla apo me hapësira ne ja lehtësojmë punën vetës duke përdorur metodën e gatshme ‘lower()’ për ta kthyer fjalinë krejt me shkronja të vogla.Gjithashtu përmes metodes ‘replace(‘ ‘,’’)’ heqim hapesirat kudo ne string.

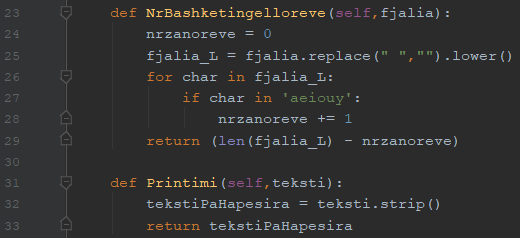
Kjo na siguron që të kemi vetëm shkronja për të numëruar. Algoritmi i tillë nuk jep përgjigje të saktë nëse klienti shtyp karaktere jashta intervalit te shkronjave a-Z dhe A-Z.

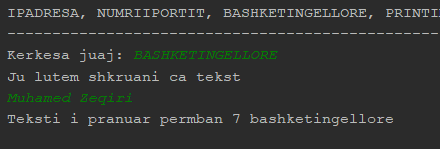
-Në çdo cikël pyesim a përputhet karakteri i pozitës në fjali me njërën nga zanoret ‘aeiouy’.Nëse po rrit vlerën e numratorit për një.

-Në fund kthe si rezultat ndryshimin e gjatësisë së fjalisë pa hapësira dhe vlerës në numrator.

-Kjo paraqet numrin e bashkëtingëlloreve në një fjali.

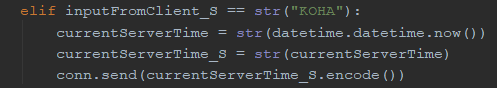
Më poshtë kemi një demostrim.

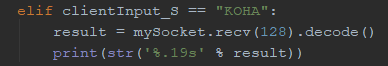


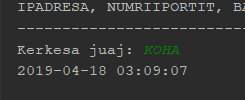


### **KOHA**

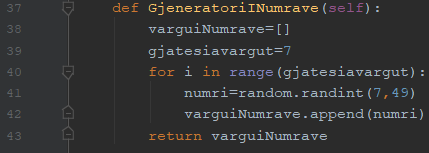
Së pari importojmë modulin datetime ‘import datetime’.Ky modul ka metodë (‘datetime.datetime.now()’) të gatshme që na kthen kohën në format të lexueshëm.Mirëpo tek klienti kam formatuar përgjigjen duke e kufizuar në vetëm 19 karaktere.Kësisoji e vërejta që shifrat pas dy shifrave tek sekonda është mirë të mos shfaqen ngase atherë bëhet e panevojshme për klientin të dijë kohën edhe në të mijtën e sekondës.

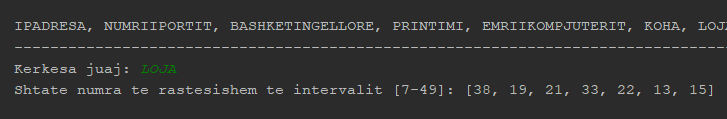






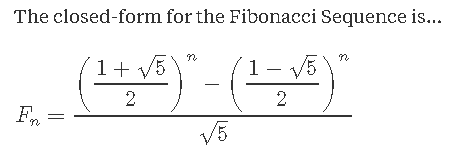
### **LOJA**

Për gjenerimin e numrave të rastit kam përdorur modulin random,duke zhvilluar funksionon GjeneratoriNumrave(),i cili ka metoden ‘randint()’ që gjeneron numra të plotë të rastit.Por që t’i dërgojmë klientit një varg prej 7 numrave të tillë kam përdorur një listë(në python listat përdoren në vend të array-s,ngase nuk ka suport built-in për to).Mandej me nje unazë kemi kufizuar që të gjenerohen vetëm 7 numra dhe pas cdo cikli secili numër i shtohet listes apo vargut logjik sepse ne praktikë është listë.Metoda e modulit random randInt(x,y) per parameter x repektivisht y merr vlerën minimale dhe maksimale që do të gjenerohet.

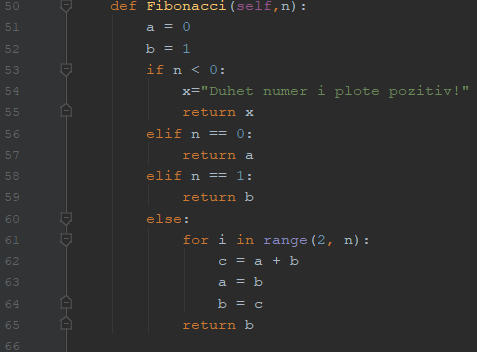


Mund të vëreni se të gjitha metodat i kam vënë si anëtarë të një klase ‘MetodatServerit’ thjesht për te çështje estetike.Mandej kam inicializuar një objekt/instancë të klasës ‘obj’ dhe me operatorin ‘.’ ju kam çasur cdo metode që kam implementuar.

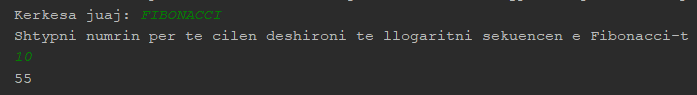
### **FIBONACCI**

Për sekuencën e famshme të Fibonacci-t kam implementuar algoritmin llogaritur sipas formulës në vijim:

Kurse metoda implementuar duket kësisoji:



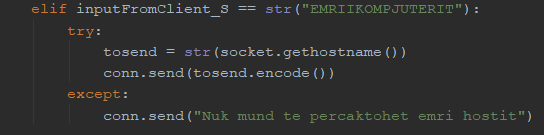
Së pari nuk lejojmë që të pranohet numër negativ për llogaritje të sekuencës.Mandej për anëtarin e parë (0) dhe të dytin (1) kthejmë të njëjtën vlerë.Kurse për anëtarët tjerë llogaritim sipas algoritmit matematik të sekuencës Fibonacci.P.sh për anëtarin e dhjetë vlera është 55.

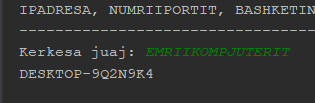


.Te parametrat e metodave në PyCharm si parametër default duhet të jepet ‘self’ e mandej parametrat tjerë.Nuk e di a është e njëjta rregull kur zhvillojmë me IDE tjetër p.sh Visual Studio.

### **EMRIIKOMPJUTERIT**

Përdorim modulin socket për t’ju çasur direkt metodës ‘gethostname()’ e cila kthen emrin e hostit. Në rast se për x arsye nuk mund të përcaktohet emri atëherë në ‘except’ shfaqim një mesazh informues për mosgjetjen e emrit të tij.

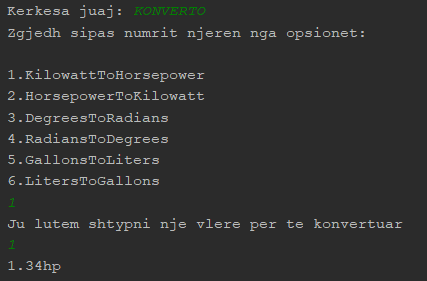


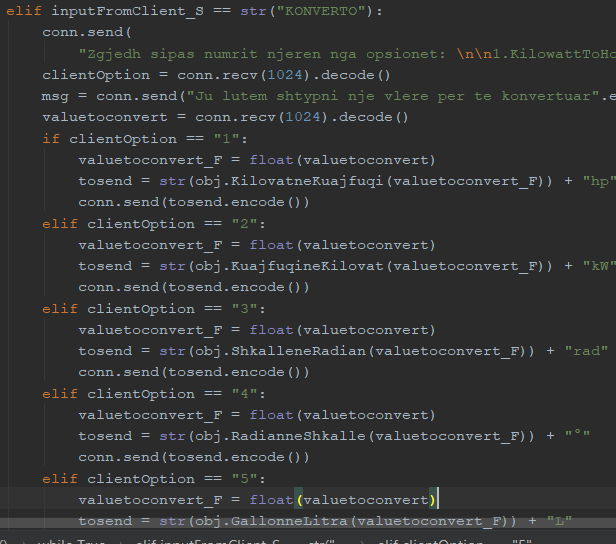


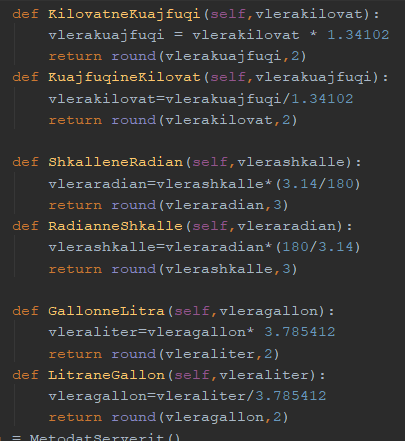
Variablat ‘tosend’ që do ti vëreni në kod i kam marrë si variabla që mbajnë përgjigjen e përpunuar dhe të gatshme për dërgim klientit,ku vetëm enkodohen si të nënkuptuara në ‘UTF-8’.

### **KONVERTO**

Për këtë operacion kam zhvilluar metoda aq sa ka operacione për konvertime.Në vijim kemi pamje nga ana e serverit(validimi ekziston edhe te klienti) ku kam validuar se pas pranimit të kërkeses KONVERTO të ipet mundësia përmes numrave [1-6] të zgjidhet opsioni i dëshiruar e më pas të ipet vlera për konvertim.Në fund rezultati me njësinë përkatëse.Sa për demostrim po shfaqi një konvertim te 1 kilovati në kuaj fuqi.





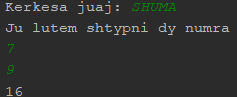


## **OPERACIONET SHTESË TË SERVERIT**

Për operacione shtesë të serverit kam zgjedhur të implementojë 4 funksione ku secili kryen një veprim të caktuar.

### **SHUMA**

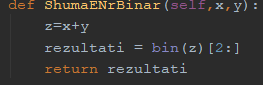
Me funksionin ‘Shuma(x,y)’ nga klienti vijnë dy inpute të njepasnjeshme që duhet të jenë numra në formatin decimal.Funksioni thjesht mbledh numrat dhe ja kthen klientit.

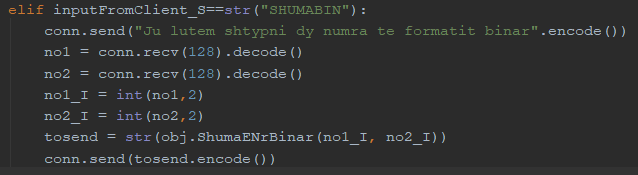


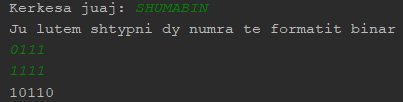
### **SHUMABIN**

Për të kryer veprimin e mbledhjes klienti duhet të shtyp dy numra në zero dhe njësha.Funksioni ‘ShumaNrBinar() ‘ merr dhe mbledh numrat e më pastaj shumën e tyre e kthen në trajtë binar ‘bin()’ , dhe largon dy karakteret e para që jane ‘0b’ që perfaqësojnë formatin binar për numrin.

Mandej si rezultat klienti merr shumën e numrave binar që ka shtypur.Kështu problemi i ‘carry bit-it’ zgjidhet shumë lehtë pa nevojë të llogaritjeve shtesë.





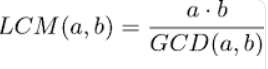


### **CAP**

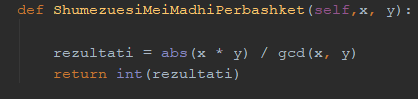
Ky funksion kthen si rezultat fjalinë e shtypur por me shkronja të mëdha. Realizohet përmes funksionit të gatshëm ‘upper()’.

### **SHMVP**

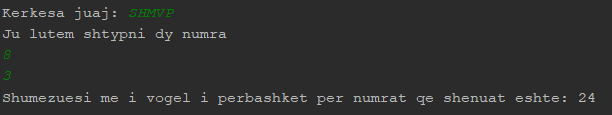
Me SHMVP klienti mund të llogaris ‘Shumëzuesin më të vogël të përbashkët’. Klienti duhet të shtyp dy numra njëri pas tjetrit kurse llogaritja përmes funksionit ‘ShumezuesiMeiVogeliPerbashket()’ bëhet duke shfrytezuar formulën matematikore:



GCD paraqet ‘Pjestuesin më të madh të perbashkët’ dhe këtë e llogarisim nga funksioni i modulit fractions i cili duhet importuar paraprakisht.Më pastaj thjesht kryejmë llogaritjet e zakonshme.



-typo error në emërtim të funksionit. Në vend të ‘MeiMadh’ është ‘MeiVogel’.

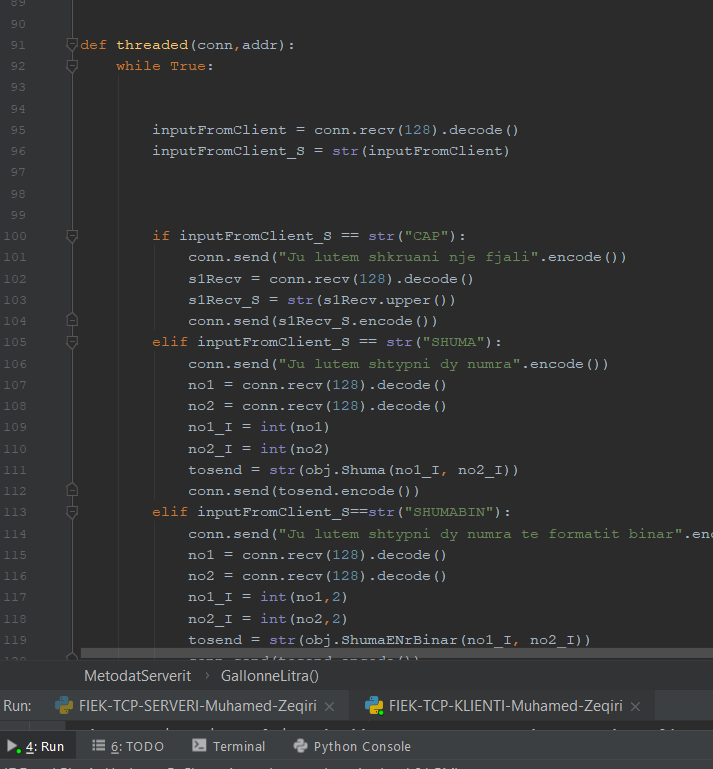


## **IMPLEMENTIMI I MULTITHREAD.SERVERI PRANON NË TË NJËJTËN KOHË SHUMË KËRKESA NGA SHUMË KLIENTË.**

Serverët kur dizajnohen duhet të jenë të tillë që të mund ti shërbejnë shumë klientëve dhe në të njëjtën kohë të jenë në gjendje ta bëjnë nje gjë të tillë.

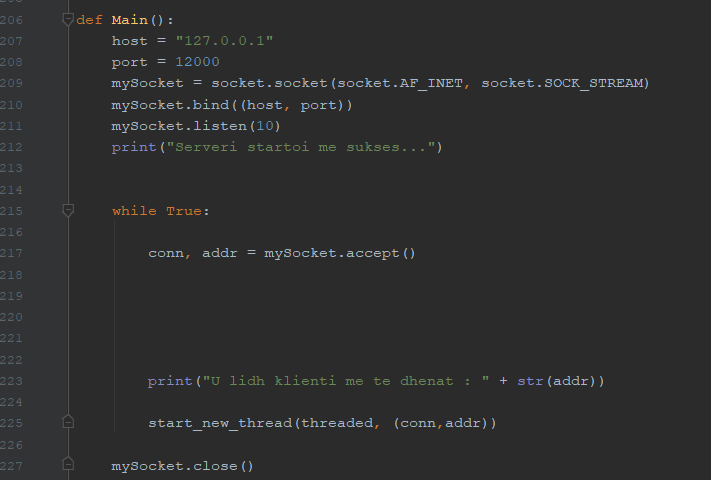
Në kuadër të TCP Serverit kam arritur të implementojë mundësinë që brenda një procesi të kryhen shumë punë dhe thjesht të ndahet puna e në këtë formë mundet që serveri të pranojë të komunikojë me shumë klient pa ndonjë problem.Këtë e zgjidhim duke implementuar ‘multithreading’.

Së pari duhet importuar moduli ‘threading’ dhe i tërë kodi për validim dhe operacione te serveri të vendoset brenda nje funksioni që do ta quajmë ‘threaded(koneksioni,adresa[[6]](#footnote-6))’



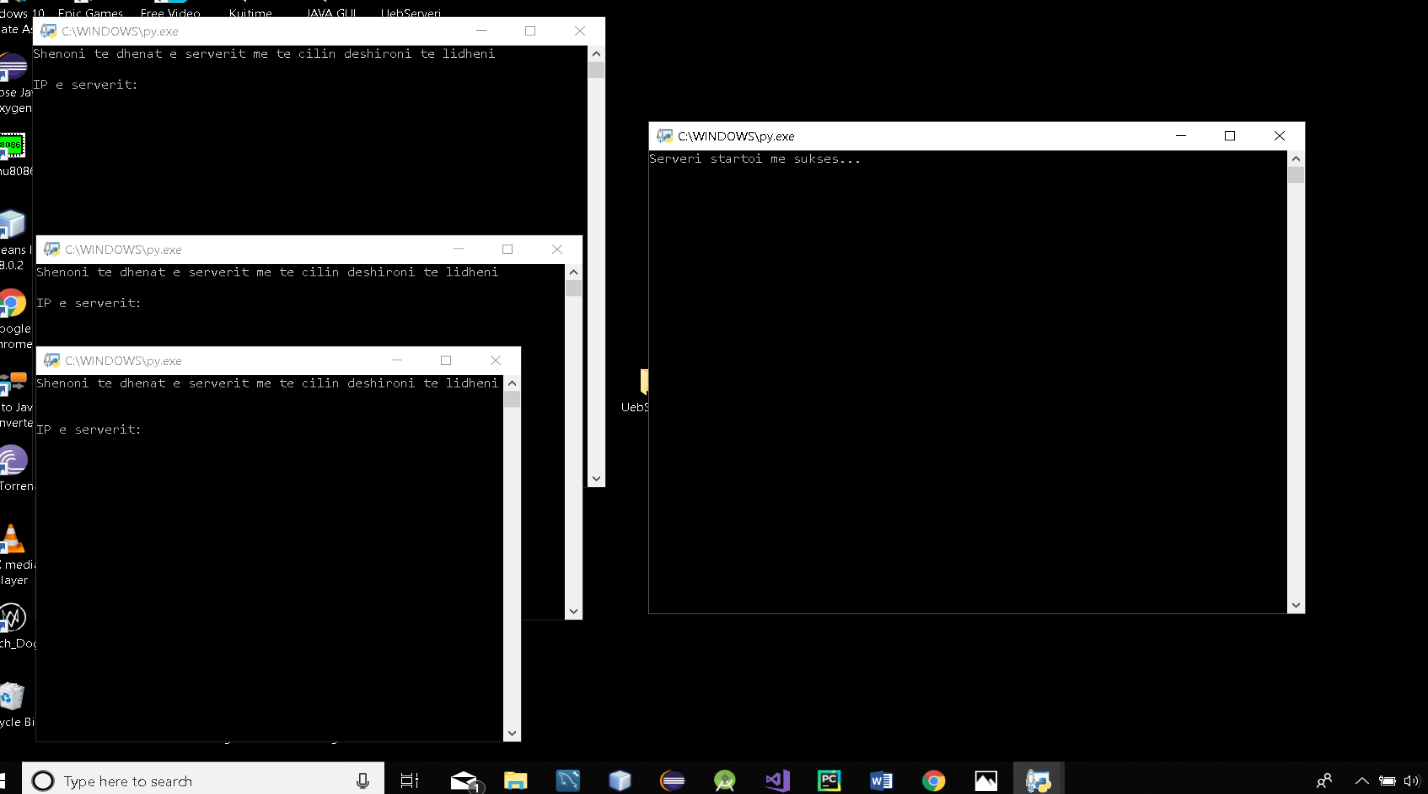
Më pas këtë funksionin ‘Main()’ startojmë një thread përmes funksionit ‘start\_new\_thread(x,y).

Ku x paraqet funksionin threaded() që e kemi krijuar më herët dhe y paraqet atë se çka pranon soketa,pra koneksionin dhe adresen(conn,addr).

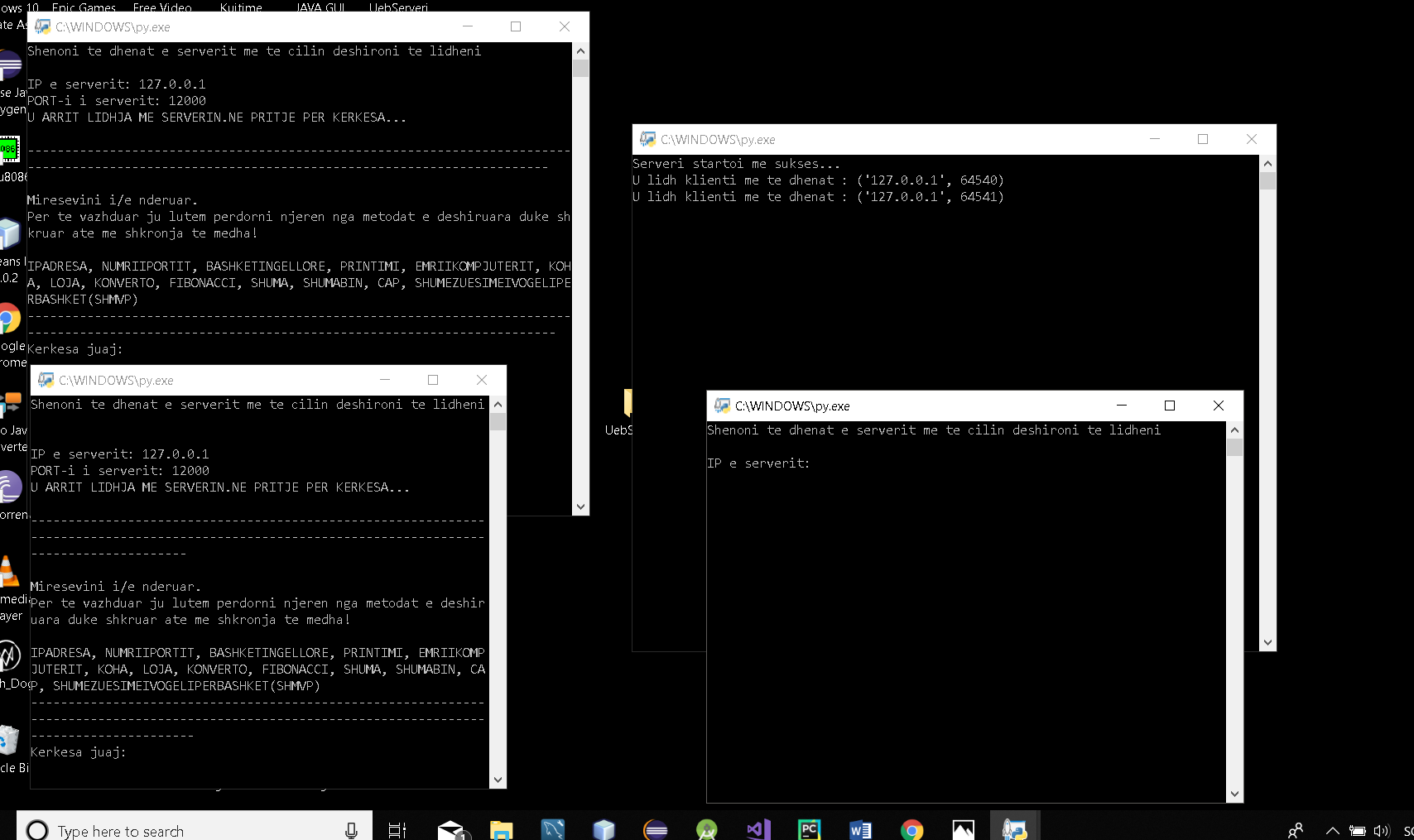


Me këto veprime serveri tashmë është i gatshëm të ju shërbejë shumë klientëve në të njëjtën kohë.

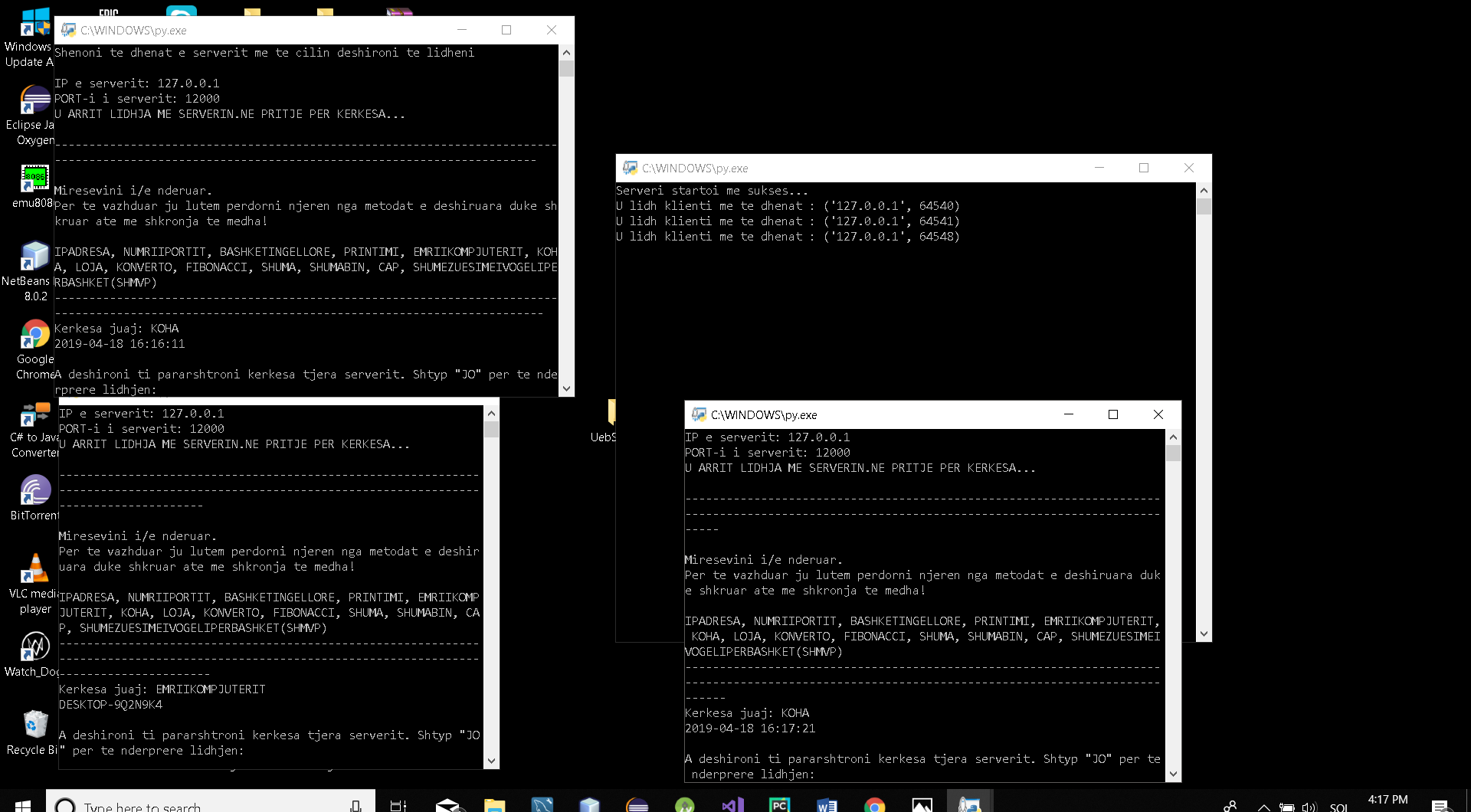
Tash për ta dëshmuar këtë kam ekzekutuar serverin dhe tre klientë përmes CMD sepse nuk me ofronte PyCharm të shihja 4 dritaret e programeve të ekzekutuara.



Më poshtë kemi demostrimin duke konektuar 3 klientë:

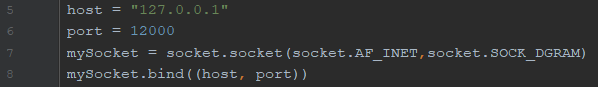


Kurse fotoja në vijim është dëshmi që pas konektimit kemi dërguar kërkesa të njëjta (KOHA nga dy klientë) dhe të ndryshme (EMRIIKOMPJUTERIT nga një klient).

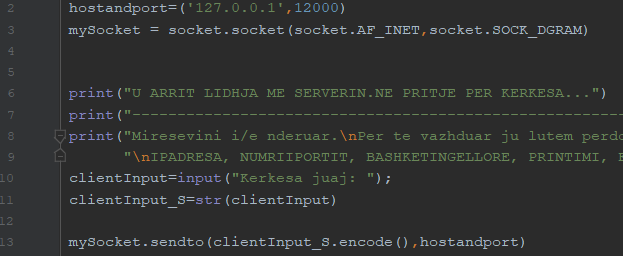


# **ZHVILLIMI I SERVERIT-KLIENTIT SIPAS PROTOKOLLIT UDP**

Sa i përket pjesës me UDP , në vijim kam demostruar dallimet në kod me TCP. Së pari tek serveri nuk kemi të bëjmë me ‘SOCK\_STREAM’ por me ‘SOCK\_DGRAM’.Mandej nuk ka koneksion.



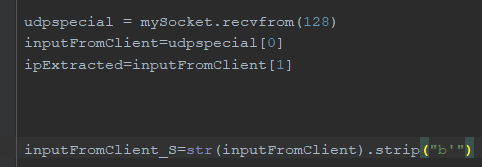
Kurse te klienti kemi pamjen si në vijim:



Vërejmë se si dergohen të dhënat nga klienti përmes metodes ‘sendto()’.

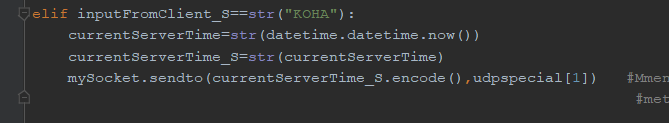
Operacionet e njëjta sikurse te TCP mund të implementohen edhe këtu por për testim se si funksionon UDP kam marrë operacionin ‘KOHA’

Kur dërgojmë të dhëna nga klienti te serveri , dhe i pranojmë ato atëherë e verejmë se atë se cka pranojmë është listë dy anëtareshe.Anëtari i parë paraqet mesazhin cka ka derguar klienti ndërsa anëtari i dytë është një listë tjeter që përmban IP dhe PORT si anëtar të parë respektivisht të dytë.



Në ‘udspecial’ pranohet çka dërgon klienti.Më pas nxjerrim mesazhin që gjendet në anëtarin e parë të kësaj variable ‘udpspecial[0]’.Ip dhe Port-in e ruajmë në ‘ipExtracted’ që është anëtari i dytë.Mesazhit nga klienti ja heqim reprezentimin ‘b’ permes metodës strip që të validojmë saktë në anën e serverit.

Tash të vërejmë se si dërgojmë përgjigje kur klienti shtyp operacionin ‘KOHA’ (njësoj edhe për operacionet tjera’.



Për UDP nuk ka nevojë në implementimin e multithreading, ngase nuk komunikohet përmes ‘IO Streams’ por përmes datagrameve nga të gjithë klientët në të njëjtin kanal.Për dallimet tjera është spjeguar më lartë në dallimet mes dy protokoleve TCP dhe UDP.

# **PËRFUNDIME**

Realizimi i këtij projekti më ndihmoi shumë që ti përforcojë konceptet se si dy protokollet funksionojnë në praktikë, dhe poashtu përdorimin e soketave.

E veçanta është përdorimi për herë të parë i gjuhës Python. Duke qenë se preferoj gjuhën Java, u sfidova që të bëj kalimin në Python por çuditërisht për të mirë këtë arrita ta bëjë me lehtësi.Sintaksa e thjeshtë e Python dhe strukturimi vetem me zhvendosje të kodit është benefit.

Si referencë kam shfrytezuar konceptet teorike të mesuara gjatë ligjëratave dhe ushtrimeve si dhe burime si uebfaqen zyrtare të gjuhës Python, dokumentimin e gjuhës në lidhje me metoda ,video demostrime mbi Python, dhe koncepte të tjera.

1. Transmission Control Protocol [↑](#footnote-ref-1)
2. User Datagram Protocol [↑](#footnote-ref-2)
3. Integrated Development Enviroment, softuer që mundëson programerëve zhvillimin e aplikacioneve [↑](#footnote-ref-3)
4. Njësi e të dhenave që shkëmbehet mes paisjeve në rrjetë. [↑](#footnote-ref-4)
5. Shpërndarje e imazhit/videos sipas kardinalitetit një në shumë përmes spektrit elektromagnetik. [↑](#footnote-ref-5)
6. Variabla addr që mban IP dhe PORT-in pra si listë dy anëtarëshe [↑](#footnote-ref-6)