### Abstrakt

Rrjetat kompjuterike janë të dizajnuara që të mundësojnë komunikimin mes pajisjeve të ndryshme të cilat janë të kyçura në një rrjet të njejtë.

Zakonisht ashtu si në jetën reale gjatë komunikimit nëpër rrjeta ekziston dërguesi(klienti) dhe marrësi(serveri), pra, klienti është ai që vazhdimisht dërgon dhe kërkon të dhëna nga serveri kurse serveri iu përgjigjet atij. Nevoja për ekzistimin e një lidhjeje klient/server në rrjeta kompjuterike ka lindur për arsye se zakonisht klienti ka kapacitet të kufizuar dhe për të bërë punën më të shpejtë dhe efikase kanë hyrë në përdorim serverët të cilët kryejnë çfarëdo veprimi që klienti ka kërkuar dhe kthejnë vetëm rezultatin final tek klienti, pra i gjithë procesi kryhet tek serveri dhe klienti nuk ngarkohet.

Për të arritur lidhjen mes klientit dhe serverit ka metoda të ndryshme , në këtë projekt kemi përdorur soketat (sockets) të cilët shërbejnë si pika fundore në secilën anë dhe na mundësojnë komunikimin mes dy programeve.

**Përmbajtja**

[**Abstrakt 2**](#_Toc6473779)

[**Hyrje 4**](#_Toc6473780)

[**Përshkrimi i socket klient dhe socket server 5**](#_Toc6473781)

[**METODAT 6**](#_Toc6473782)

[**METODA IPADRESA 6**](#_Toc6473783)

[**METODA NUMRIPORTIT 6**](#_Toc6473784)

[**METODA BASHTINGËLLORE 6**](#_Toc6473785)

[**METODA PRINTIMI 6**](#_Toc6473786)

[**METODA EMRIKOMPJUTERIT 6**](#_Toc6473787)

[**METODA KOHA 7**](#_Toc6473788)

[**METODA LOJA 7**](#_Toc6473789)

[**METODA FIBONACCI 7**](#_Toc6473790)

[**METODA KONVERTIMI 7**](#_Toc6473791)

[**METODA REVERSE 7**](#_Toc6473792)

[**-TESTIMI- 8**](#_Toc6473793)

[**8**](#_Toc6473794)

[**Referencat 12**](#_Toc6473795)

### Hyrje

Komunikimi sa më i lehtë mes pajisjeve të ndryshme është qellimi kryesor i rrjetave kompjuterike gjë e cila jemi munduar ta realizojmë me prototipet e klientit dhe serverit të programuara në këtë projekt.

Për realizimin e klientit dhe serverit është përdorur gjuha programuese PYTHON dhe editori Visual Studio 2017 ndërsa për krijimin e lidhjes mes klientit dhe serverit janë përdorur soketat(sockets) si pika fundore në secilën anë të cilat mundësojnë qarkullimin e të dhënave mes klientit dhe serverit.

Ekzistojnë dy mënyra të komunikimt të klientit me serverin , lidhja TCP dhe lidhja UDP. Lidhja TCP është lidhje më e sigurtë, serveri krijon lidhje me klienit në fillim, informatat nuk humbin gjatë rrugës dhe klienti informohet për çdo parregullsi , ndërkaq tek lidhja UDP serveri nuk krijon lidhje të vazhdueshme me klientit , shkëmbimi i informatave është i shpejtë por rritet mundësia e humbjes se informatave për shkak mungesës së lidhjes. Në realizimin e këtij projekti janë programuar të dy lidhjet , lidhja TCP dhe UDP. Dallimi mes programimit të këtyre dy programeve nuk është i madhë pasi që soketat na mundësojnë të krijojmë soketat e nevojshëm për secilën lidhje. Përdoret protokolli i lidhjes sock.STREAM për lidhjen TCP dhe sock.DGRAM për lidhjen UDP.

Një server duhet të jetë në gjendje te komunikojë me më shumë se një klient njëkohësisht dhe gjithashtu duhet të merret parasyshë se ai duhet të pranojë dhe të kthejë informata tek klienti vazhdimisht dhe për implementimin e këtyre vetive tek serveri kemi perdorur threading modul i cili i mundëson qasjen e shumë klientave njëkohësisht tek i njejti server pa paraqitur problem dhe ngatrresë mes kërkesave.Gjithashtu, përveq kësaj që të mos kemi ndërprerje të punës së serverit edhe pasi që klienti të jep një komandë të panjohur kemi programuar serverin në atë mënyrë që serveri të injoroj kërkesat e panjohura por të mos shkëputet lidhja me klientin.

Vetitë e cekura janë përkrahur në programimin e të dyja lidhjeve të lartcekura.

### Përshkrimi i socket klient dhe socket server

Në kuadër të gjuhës python, për realizimin e socket-ave janë përdorur libraritë e gatshme: socket me metodat e saj, datetime, thread, sys, random.  
Përveç metodave që kemi pasur detyrë t’i krijojmë kemi përdorur edhe metodat e definuara në klasët e librarive të lartpërmendura: bind (bën lidhjen e socket-it me adresën përkatëse që përbëhet nga hosti dhe porti), listen, encode (kodon të dhëna në nivel të bajtave), decode (bën deokodimin e të dhënave në UTF-8), recv (me parametër buffer size, pranon të dhëna nga kilenti), recvfrom (data,address nga socket-i përkatës), rsplit (si delimiter shfrytëzon hapësirën dhe krijon një listë nga stringu i dhënë), clientthread (krijon thread për secilin klient që kyçet në server, për TCP ka parametrin që përshkruan socket-in me të cilin është lidhur, ndërsa për UDP përmban vetëm të dhënat nga klienti dhe adresën e tij), toupper(shkronjat e shtypura nga klienti i kthen në shkronja të mëdha), now (kthen kohën aktuale), gethostname (emri i hostit), gethostbyname(kthen IP e paisjes) etj.

# 

# METODAT

### METODA IPADRESA

Me anën e metodës IPADRESA klienti është në gjendje të gjej IP e pajisjes së tij. Për realizimin e kësaj metode është përdorur funksioni i gatshëm i gjuhës programuese Python gethostbyname() e cila si parametër hyrës ka marrë një funksion tjetër të gatshëm(gethostname()) e cila kthen emrin e hostit të klientit për arsye që metoda të marr IP adresën e secilës paisje në të cilën ekzekutohet programi.

### METODA NUMRIPORTIT

Metoda NUMRIPORTIT kthen portin të cilin klienti është duke e përdorur . Për realizimin e kësaj metode janë përdorur komandat për të dërguar dhe marrë informata prej serverit i cili si përfundim ka kthyer portin e klientit.

### METODA BASHTINGËLLORE

Metoda BASHTINGELLORE është metodë me anë të së cilës klienti ka mundësi të gjej numrin e bashtingelloreve në një fjali të dhënë. Metoda bashtingëllore është realizuar duke marrë fjalinë e klientit pastaj duke e krahasuar atë fjali shkronjë për shkronjë me një array në të cilën janë të vendosura të gjitha bashtingëlloret. Përveq kësaj është deklaruar dhë një variabël tjetër me emrin “numratori” dhe është inicializuar në vlerën 0. Vlera ë kësaj variable shtohet për një sa herë që programi hasë në një bashtingëllore gjatë krahasimit.

Në lidhjen TCP metoda është dhënë në atë mënyrë që fillimisht ceket metoda pastaj programi kërkon fjalinë nga klienti ndërsa në lidhjen UDP për arsye se serveri nuk krijon lidhje me serverin kjo nuk ka mundur të arrihet dhe metoda është dhënë në formatin BASHTINGËLLORE {hapsirë} fjalia e klientit.

### METODA PRINTIMI

Më anë të metodës PRINTIMI klientit i ofrohet mundësia të shtypë një fjali dhe pas kësaj i shfaqet në ekran fjalia të cilën ai e ka shtypur. Për realizimin e kësaj metode jan përdorur komandat për dërgimin dhe marrjen e informatave në server dhe pastaj më anë të komandës print() printohet fjalia e klientit.

### METODA EMRIKOMPJUTERIT

Metoda EMRIKOMPJUTERIT është bërë me qëllimin që klienti të këtë mundësi të dijë emrin e pajisjes së tij në çdo kohë. Për realizimin e kësaj metode është përdorur funksioni i gatshëm gethostname() e cila e kthen emrin e pajisjes në të cilën ekzekutohet programi

### METODA KOHA

Metoda KOHA kthen kohën aktuale të serverit. Për realizimin e kësaj metode në fillim kemi importuar librarinë e gatshme datetime të gjuhës programuese Python e cila na mundëson të manipulojmë me data dhe kohë e pastaj kemi përdorur funksionin e gatshëm të kësaj librarie datetime.now() e cila kthen kohën dhe datën aktuale por është formatuar në mënyrë që të kthej vetëm kohën e serverit.

### METODA LOJA

Qëllimi i metodës LOJA është që të gjenerojë numra rëndom në mes numrit 1 dhe 49 , kurse për caktimin e sa numrave të gjenerojë e cakton klienti se sa i nevoiten. Për realizimin e kësaj metode së pari është importuar libraria e gatshme random pastaj është përdorur funksioni i gatshëm random.sample(range(,),)ku tek range caktohen pika e fillimit dhe e fundit e vargut prej të cilit dëshirojmë të gjenerojmë numrat dhe pas kësaj caktohet sa numra prej atij vargu dëshironë klienti të gjenerojë.

### METODA FIBONACCI

Metoda FIBONACCI ka për funksion të gjejë fibonaccin e numrit të shtypur nga klienti. Në lidhjen TCP kjo metodë është dhënë në atë mënyrë që së pari i tregohet serverit se dëshirojë të përdorë metodën Fibonacci pastaj i kërkohet klientit të shtyp numrin më qëllim që programi të jetë sa më “i lehtë për përdorim” kursë në lidhjën UDP për arsye që lidhja më klientin dhe serverin nuk krijohet në fillim nuk është e mundur të dërgohet më shumë se një e dhënë kjo metodë është dhënë në formatin FIBONACCI {hapsire} numri.

### METODA KONVERTIMI

Kjo metodë i ofron klientit mundësinë të konvertojë madhësi të caktuara në njësi të ndryshme. Kjo metodë lejon konvertimin e njësive Kilowatt në Horsepower, Horsepower në Kilowatt, Degrees në Radians, Radians në Degrees, Gallons në Liters dhe Liters në Gallons.

Në lidhjen TCP kjo metodë është dhënë në atë mënyrë që klienti të tregojë që dëshiron të përdorë metodën konvertimi pastaj të jipet opsioni dhe më pas sasia e madhësisë kurse tek lidhja UDP është dhënë në formatin KONVERTIMI {hapsirë} opsioni {hapsirë} sasia e madhësisë.

### METODA REVERSE

Me ane te metodes REVERSE, ne rastin kur klienti shtyp nje fjale si pergjigjese kthehet fjala e kthyer mbrapsht.

### -TESTIMI-

**TCP**

### 

Figure Ekzekutimi i serverit

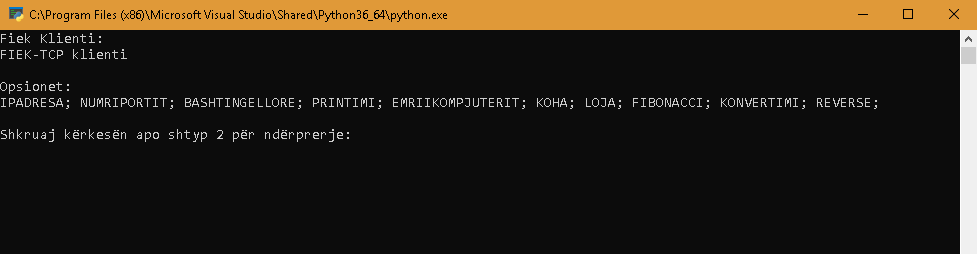


Figure 2 Lidhja e klientit me serverin

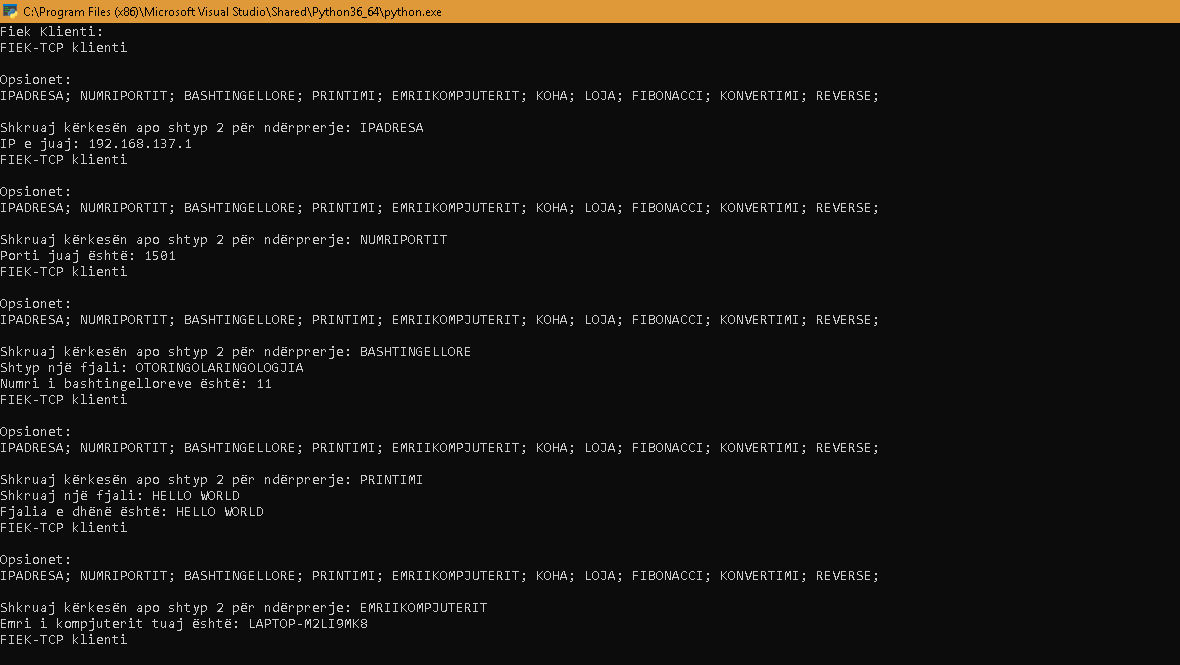


Figure 3 Testimi i metodave ne TCP

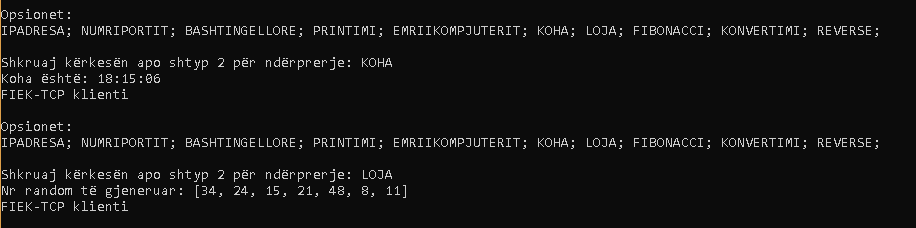


Figure 4 Testimi i metodave ne TCP

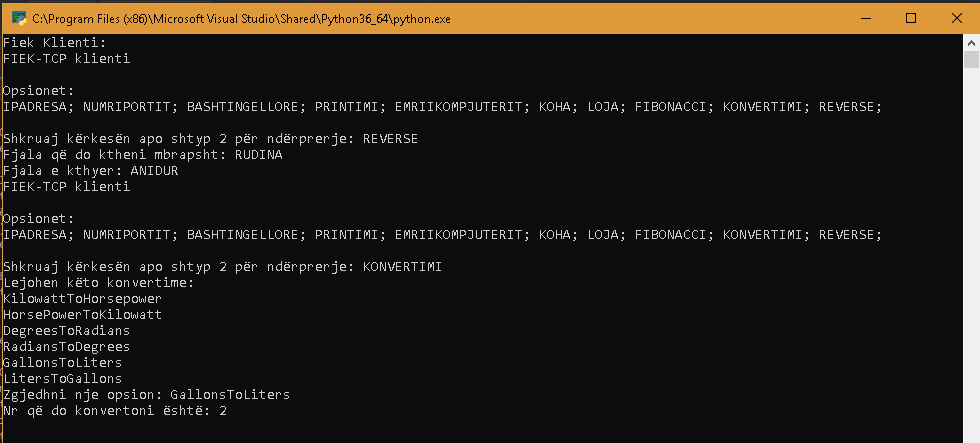


Figure 5 Testimi i metodave ne TCP

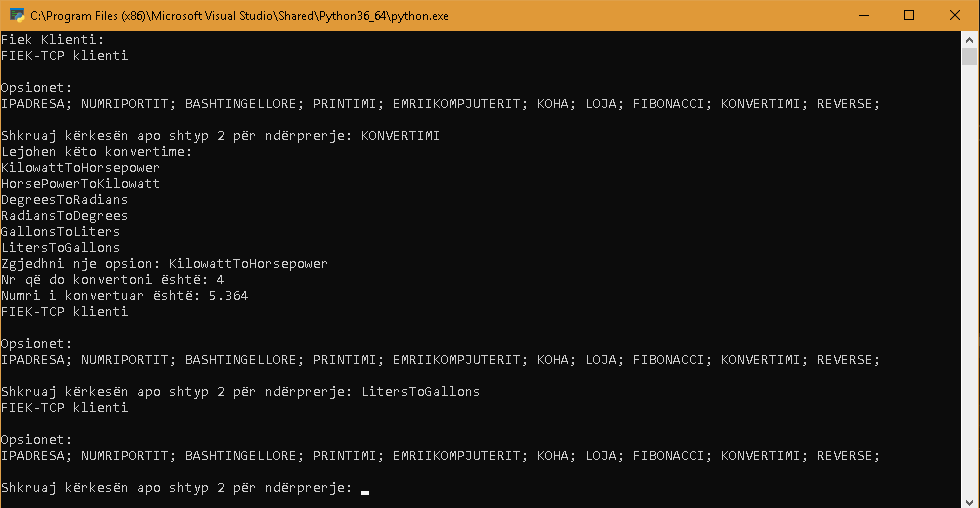


Figure 6 Testimi i metodave ne TCP

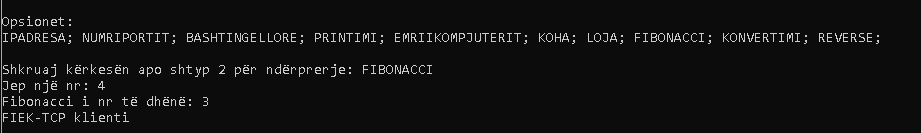


Figure 7 Testimi i metodave ne TCP

**UDP**

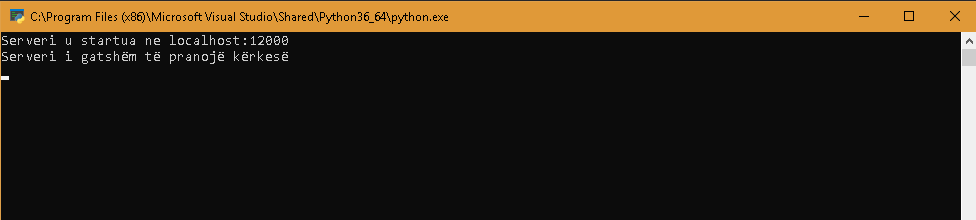


Figure 9 Lidhja e klientit me serverin

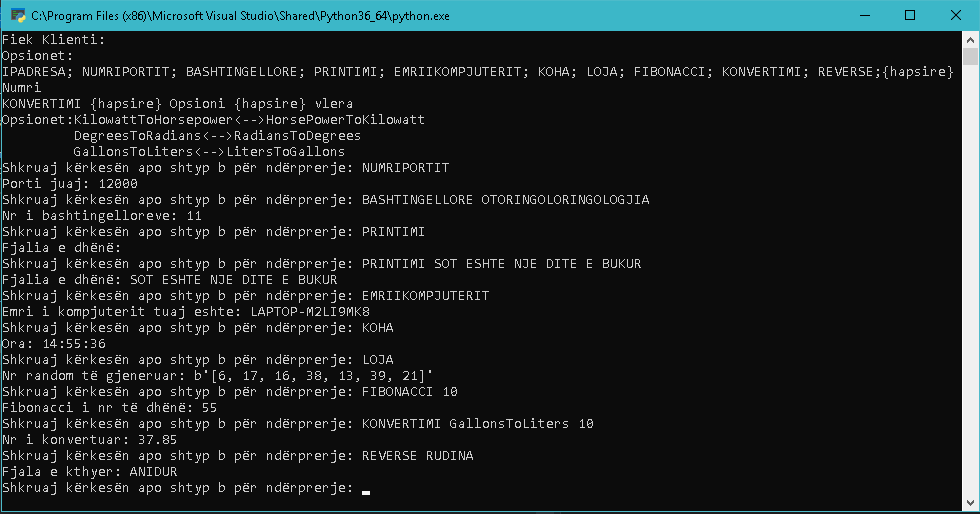


Figure 10 Testimi i metodave ne UDP

**Permbledhje rreth rezultateve:** Sic mund të vërehet edhe nga print Screen-at e vendosur me lartë, secila metodë e dhënë funskionon, dhe secila prej tyre kthen rezultatin i cili realisht duhet të kthehet dhe është i sakti.

# Referencat

### [Computer Networking A Top-Doën Approach 6th Edition](http://www.bau.edu.jo/UserPortal/UserProfile/PostsAttach/10617_1870_1.pdf), James Kurose, Keith Ross

<https://docs.python.org/>

<https://www.pythonforbeginners.com/>

<https://stackoverflow.com/>  
<https://www.tutorialspoint.com/python/>