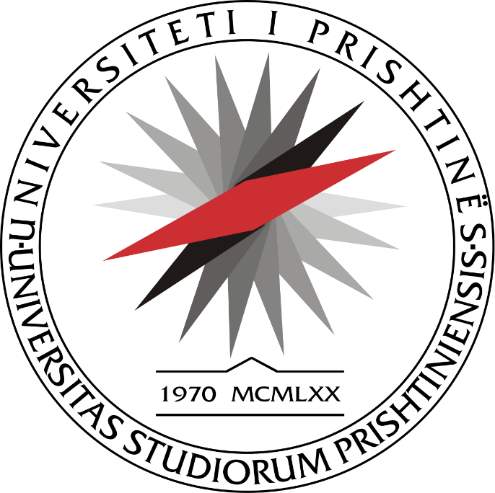
UNIVERSITETI I PRISHTINËS “HASAN PRISHTINA”

FAKULTETI I INXHINIERISË ELEKTRIKE DHE KOMPJUTERIKE



**PROJEKTI:** DIZAJNIMI KLIENT-SERVER

**LËNDA:**RRJETAT KOMPJUTERIKE

**Mentor:** Prof. Dr.Blerim Rexha **Punoi:** Arlinda Kastrati

MSc. Haxhi Lajqi **ID**: 190714100006

**Prishtinë, 2021**

# **Abstrakti**

Ky raport paraqet një përmbledhje të detajeve në dizajnimin, implementimi dhe testimi i programit klient dhe server në versionet TCP dhe UDP .Ky projekt është implementuar duke përdorur soketa dhe gjuhën programuese Python.

Objektivi kryesor i këtij raporti është të paraqesë parimet që qëndrojnë prapa programimit të programit komunikues klient/server përmes soketeve në TCP/UDP.

**Permbajtja**

[**Abstrakti** 2](#_Toc70865418)

[**Hyrja** 4](#_Toc70865419)

[**1.Metodat** 5](#_Toc70865420)

[**1.1** **Metodat kryesore** 5](#_Toc70865421)

[Metoda **IP** 5](#_Toc70865422)

[Metoda **NRPORTIT** 6](#_Toc70865423)

[Metoda **NUMERO** 6](#_Toc70865424)

[Metoda **ANASJELLTAS** 6](#_Toc70865425)

[Metoda **PALINDROM** 7](#_Toc70865426)

[Metoda **KOHA** 7](#_Toc70865427)

[Metoda **LOJA** 7](#_Toc70865428)

[Metoda **GCF** 8](#_Toc70865429)

[Metoda **KONVERTO** 8](#_Toc70865430)

[**1.2** **Metodat shtesë** 8](#_Toc70865431)

[Metoda **ANAGRAM** 8](#_Toc70865432)

[Metoda **PRIM** 9](#_Toc70865433)

[**2.** **FIEK TCP –PROGRAMIMI PERMES SOKETEVE** 9](#_Toc70865434)

[**2.1** **TCP** 9](#_Toc70865435)

[**2.2** **TCP-Modeli i komunikimit përmes soketave Klient/Server** 10](#_Toc70865436)

[**2.3** **TCP Klienti** 11](#_Toc70865437)

[**2.4** **TCP-Serveri** 12](#_Toc70865438)

[**3.** **FIEK UDP –PROGRAMIMI PERMES SOKETEVE** 13](#_Toc70865439)

[**3.1** **UDP-Protokolli i te Dhenave te Perdoruesit** 14](#_Toc70865440)

[**3.2** **UDP-Modeli i komunikimit përmes soketave Klient/Server**  14](#_Toc70865441)

[**3.3** **UDP-Klienti** 15](#_Toc70865442)

[**3.4** **UDP-Serveri** 16](#_Toc70865443)

[**4.** **Testimi i programit klient/server ne TCP dhe UDP** 17](#_Toc70865444)

[**5.** **Perfundimi** 19](#_Toc70865445)

[**6.** **Referencat** 20](#_Toc70865446)

# **Hyrja**

**Interneti** është pa dyshim sistemi më i madh i inxhinierisë i cili eshte krijuar ndonjëherë nga njerëzimi, me qindra miliona kompjutera të lidhur me vete permes lidhjeve komunikuese gjegjesisht rrjetave.Në të kaluarën, rrjetet kompjuterike ishin një grup kompjuterash të ngjashëm,të lidhur me një numer te madh te kabllove dhe te palevizshem. Në ditët e sotme, rrjetet kompjuterike janë shumë fleksibile. Teknologjia e rrjetave është ajo e cila lejon komunikimin e miliarda përdoruesve nga e gjithë bota të cilët lidhen përmes laptopëve, tabletëve dhe telefonave të menqur.

Një rrjete është e përbërë nga kompjuterë i cili është nje klient ose një server. Një server është një program që ofron disa shërbime, ndërsa klienti është një program që është duke kërkuar disa shërbime.

**Serverët** janë kompjuterë të fuqishëm ose procese të dedikuara për menaxhimin e disqeve të diskut (skedari serverat), printerët (serverat e printimit) ose trafiku i rrjetit

(shërbimet e rrjetit) ndërsa **klientët** janë PC ose stacione pune në të cilën përdoruesit ekzekutojnë aplikacione.

Kur një përdorues kërkon një shërbim,pajisja e tij duhet të jetë e lidhur në një rrjetë,qoftë ajo rrjetë lokale ose internet. Kështu që kur nje pajisjeje i nevojitet lidhja ne rrjeten lokale apo në internet, ajo pajisje përdorë komponenten softwerike të quajtur **soket**.

**Soketi** është një pikë fundore e një lidhje komunikimi dykahëshe midis dy programeve në një rrjetë. Soketetet lejojnë komunikimin midis dy proceseve të ndryshme në të njëjtën ose në pajisje të ndryshme.

Lejimi i këtij komunikimi është i mundur përmes protokolleve siq jane **TCP/IP** dhe **UDP**.

Me poshtë kemi dizjnimin,implementimin dhe testimin e një programi klient/server të implementuar përmes protokolleve TCP/IP dhe UDP.

**Veglat softwerike per realizimin e projektit:**

* Sistemi Operativ: Microsoft Windows 10
* Gjuha programuese dhe version i saj: Python 3.9.2
* Editori: Visual Studio Code,Visual Studio 2020

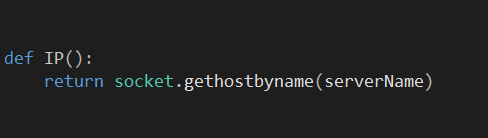
# **1.Metodat**

Të gjitha metodat janë implementuar përmes funksioneve pra me fjalën kyqe **def.**

* 1. **Metodat kryesore**

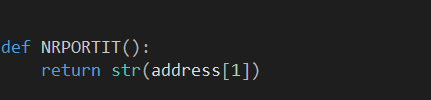
## Metoda **IP**

Përcakton dhe kthen IP adresën e klientit përmes **socket.gethostbyname(serverName)** por është e mundur edhe të realizohet përmes **str(address[0]).**



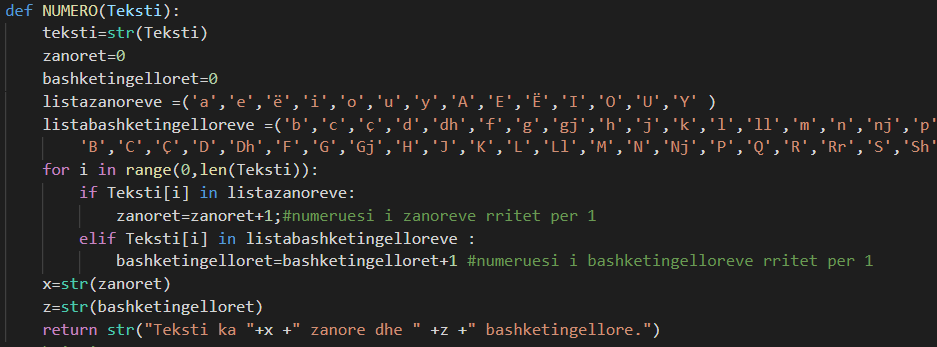
## Metoda **NRPORTIT**

Përcakton dhe kthen portin e klientit përmes **str(address[1]).** Pas deklarimit të IP adresës me emrin **serverName** dhe të portit me emrin s**everPort** ,gjatë krijimi të soketave këta dy parametra konsiderohen si **address=(serverName,serverPort)** by default.Andaj **str(address[1] )** shfaq numrin e portit në string.



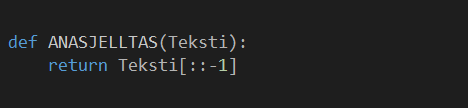
## Metoda **NUMERO**

Përcakton numri e zanoreve dhe bashkëtingelloreve të një teksti të dhënë.Deklarimi i variablave numëruese **zanoret=0** dhe **bashketingellore=0**,për cdo karakter në një rang prej **0** deri te **gjatësia e teksiti** të dhënë duke krahasuar nëse ai karakter gjendet në **listen (a,e, ë,I,o,u,y,A,E,** [**Ë**](https://sq.wikipedia.org/wiki/%C3%8B)**,I,O,U,Y)** nëse gjendet numëruesin e zanoreve **rrite +1** ,përndryshe **rrite te bashkëtingelloreve** dhe kthe nr e tyre.



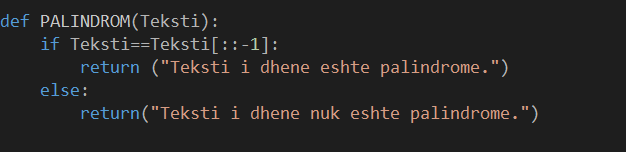
## Metoda **ANASJELLTAS**

Kthen tekstin që e ka marr si parameter në fillim në anën e kundërt. **Teksti[::-1]** nënkupton se do të fillosh nga fundi i stringut dhe të përfundosh në pozicioni **0** pra pozicionin e parë ,me hapin **-1** që do të thotë një hap prapa.



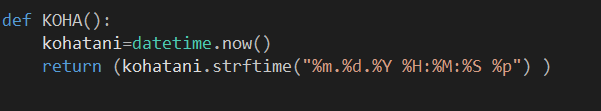
## Metoda **PALINDROM**

Palindromi nuk është gjë tjetër veçse ndonjë numër ose fjali e cila mbetet e pandryshuar edhe kur përmbyset. Përmes kushti **if Teksti==Teksti[::-1]** krahason nese Teksti i marr si parameter është i njejte me Tekstin e permbysur (Metoda ANASJELLTAS). Dhe përmes **return(“Teksti…”)** na tregon nese fjalia është palindrome apo jo.



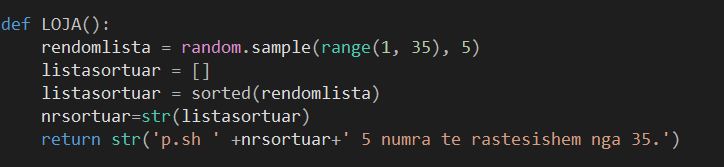
## Metoda **KOHA**

Përcakton kohën aktuale në server dhe e dërgon atë tek klienti varësisht formatit që ai e ka zgjedhur.Për realizimin e saj nevojitet importimi i libraris datetime.



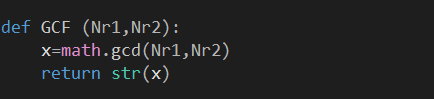
## Metoda **LOJA**

Ka për qëllim që të kthej 5 numra nga rangu i caktuar [1,35]. **Rendomlista = random .sample (range (1,35),5)** gjenron nje varg prej 5 numrash në rangun 1-35, deklarojme një varg të zbrazët ku vendosen numrat pas sortimit dhe kthejme atë.



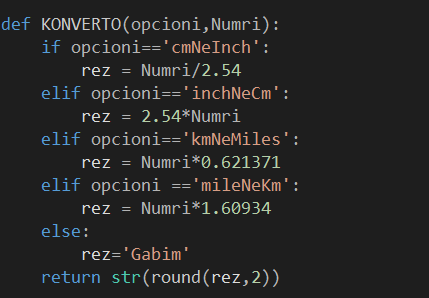
## Metoda **GCF**

Përcakton faktorin më të madh të përbashkët në mes dy numrave dhe kthen atë që është. Shfrytezojmë funksionin që gjenden tek libraria **math** sic është **math.gcd**.



## Metoda **KONVERTO**

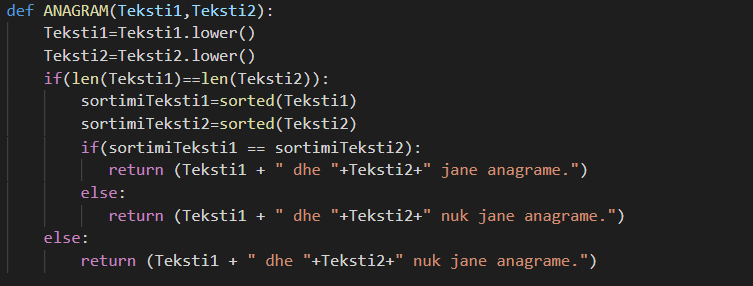
Kthen si rezultat konvertimin e një vlerë në një vlerë tjeter varësisht opcionit të zgjedhur. Opcionet e mundshme janë: **cmNeInch inchNeCm kmNeMiles mileNeKm**.



* 1. **Metodat shtesë**

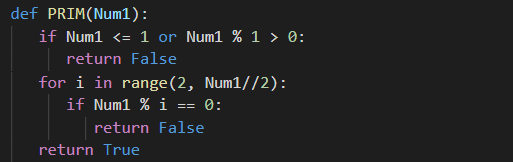
## Metoda **ANAGRAM**

Përcakton nesë dy tekste janë anagrame. Anagram-nënkupton nëse dy tekste të ndryshme me gjatësi të njëjtë përmbajnë shkronjat e njëjta.



## Metoda **PRIM**

Përcakton nëse një numer është i thjesht dhe kthen përgjigjen në fomen booleane.



1. **FIEK TCP –PROGRAMIMI PERMES SOKETEVE**
   1. **TCP**

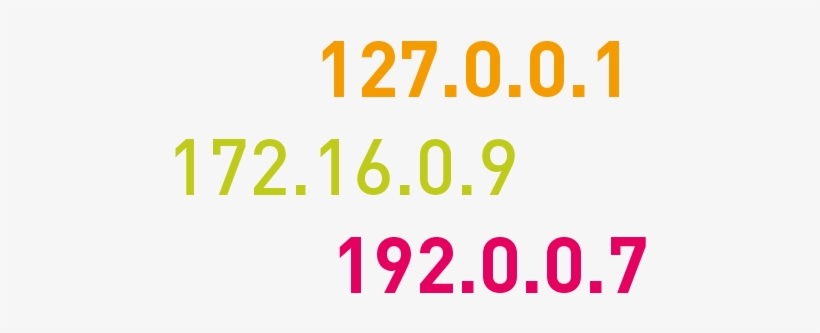
**TCP**(Transmission Control Protocol- Protokolli i Kontrollit të Transmetimit) është një standard i komunikimit që mundëson programet dhe pajisje kompjuterike për të shkëmbyer mesazhe mes vete në një rrjet.

Pasi që,TCP është e orientuar drejt lidhjes(connection-oriented) ,fillimisht duhet vendosur një lidhje midis klientit dhe serverit përpara se të dërgohen të dhënat. Serveri duhet të jetë duke dëgjuar për kërkesat e lidhjes nga klientët para se të vendoset lidhja ose te behet connection me klientin.

TCP është i përshtatshëm për aplikacione apo programet që kërkojnë besueshmëri të lartë dhe koha e transmetimit është relativisht jo shume kritike. Përdoret nga protokollet si HTTP, FTP, SMTP etj.

TCP rregullon paketat e të dhënave në rendin e specifikuar. Njëkohesisht,ekziston një garanci absolute se të dhënat e transferuara mbeten të paprekura dhe arrijnë me të njëjtin rend me të cilin janë dërguar. TCP merret me besueshmërinë,kontrollin e rrjedhes se transmetimit se te dhenave, kontrollin e mbingarkesës, kontrollimin e gabimeve dhe rikuperimin e gabimeve. Paketat e gabuara ritransmetohen nga burimi në destinacion

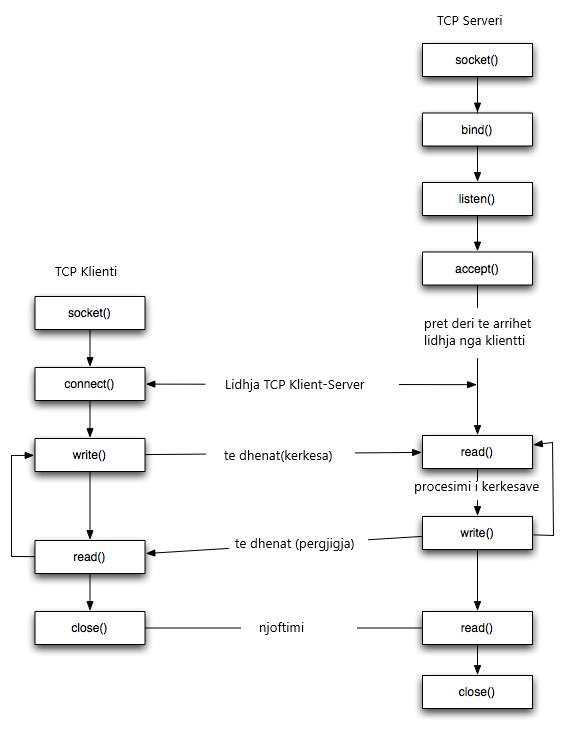
IP(Internet Protocol-Protokolli I Internetit) është metoda për dërgimin e të dhënave nga një pajisje në tjetrën përmes internetit. Çdo pajisje ka një adresë IP që e identifikon atë në mënyrë unike dhe i mundëson asaj të komunikojë dhe të shkëmbejë të dhëna me pajisje të tjera të lidhura në internet.



TCP dhe IP janë protokolle të veçanta që punojnë së bashku për të siguruar që të dhënat dorëzohen në destinacionin qe eshte synuar ne fillim brenda nje rrjeti.

IP merr dhe përcakton adresën e aplikacionit ose pajisjes në të cilën duhet të dërgohen të dhënat. TCP më pas është përgjegjës për transportimin e të dhënave dhe sigurimin që ato dorëzohen në aplikacionin e destinacionit ose pajisjen që IP ka përcaktuar.

* 1. **TCP-Modeli i komunikimit përmes soketave Klient/Server**



1.Realizimi i lidhjes TCP Klienti-Serveri 2.Klienti lexon kërkesat nga tastiera dhe i dergon tek serveri 3.Serveri lexon kërkesen e marr nga Klienti dhe e proceson varësishte se qfarë kërkon kërkesa 4.Serveri dërgon përgjigjen e kërkesës 5.Klienti e lexon përgjigjen e pranuar 6.Klienti mbyllet dhe dergon njoftimin te severi 7.Serveri lexon njoftimin dhe mbyllet

* 1. **TCP Klienti**

Kodi për implementimin e soketit tek TCP-Klienti. Kodi i plotë gjendet ne fjallin TCP-Klienti.py.

import socket #Importi i te gjitha moduleve nga libraria socket

serverName = '127.0.0.1' #IP Adresa e localhostit

serverPort = 14000 #NUMRI I PORTIT te serverit

#Funksioni/Kushtet per nderrim te portit

#.....

soketi = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)#Ky rresht krijon soketin e klientit te quajtur soketi dhe permban dy parametra.

#Parametri i pare AF\_INET-tregon se rrjeti perdore IPv4

#Parametri i dyte SOCK\_STREAM-tregon se soketi eshte i tipit SOCK STREAM qe nenkupton qe oshte TCP soket

soketi.connect((serverName,serverPort))#Komunikimi mes Klientit dhe Serverit

#Parametri i metodes connect eshte adresa e anes se komunikimit te Serverit

print('---------------------------SOKETI TCP\_Klienti-------------------------')

kerkesa = input("I/e nderuara klient/e ju sapo jeni lidhur me serverin,ju lutem shkruani kerkesen tuaj: ")

#...................

while True:#unaza e pafundme

      if (kerkesa == ""): #Nese kerkesa eshte ENTER

          soketi.close() #Soketi do te mbyllet

          print('Lidhja me serverin eshte mbyllur') #Lajmerimi qe lidhja me  serverin rin eshte mbyllur

          break

      else:

  #Dergimi i kerkeses tek serveri(gjegjesishte ne localhost me portin

#qe e kemi percaktuar tek serverPort)

           soketi.sendall(kerkesa.encode())#Default encoding=utf-8

           #Variabla pergjigja nenkupton pergjigjen nga serverin ne baze te   kerkesave te parashtruara

           pergjigjja = soketi.recv(128).decode()#Default decoding=utf-8

           print(pergjigjja) #Printimi i pergjigjes

           kerkesa = input('Ju lutem shkruani kerkesen tuaj: ')

    soketi.close()

* 1. **TCP-Serveri**

Kodi për implementimin e soketit tek TCP-Serveri. Kodi i plotë gjendet në fjallin TCP-Serveri.py.

#Importimi i librarive te nevojshme

import socket

#.....

import threading

serverName = '127.0.0.1' #IP Adresa e localhostit

serverPort = 14000 #NUMRI I PORTIT te serverit

serverS = socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)#Ky rresht krijon soketin e serverit te quajtur serverS dhe permban dy parametra.

#Parametri i pare AF\_INET-tregon se rrjeti perdore IPv4

#Parametri i dyte SOCK\_STREAM-tregon se soketi eshte i tipit SOCK STREAM qe nenkupton qe oshte TCP soket

serverS.bind((serverName, serverPort)) #Percaktimi i serverName dhe serverPort per lidhje

print('Serveri eshte startuar ne localhost ne portin ' + str(serverPort))

serverS.listen(5) #Serveri pret kerkesat e klientit

print('----------------------------TCP\_Serveri--------------------------')

print('Server eshte duke punuar dhe eshte duke pritur per ndonje kerkese!')

#Funksionet

#.......

#Funksionet

#Threads

#...

while True:

    print('-----------------------------------------------------------------')

    connection, address = serverS.accept() #Pergjigjja,pranimi i nje lidhjeje te   re ne soketin e serverS

    print("Serveri është lidhur me klientin me IP Adresë %s, në portin %s" % addr

ess)

    threading.\_start\_new\_thread(ThreadFunction,(connection,)) #Krijimi i threads

serverS.close()

1. **FIEK UDP –PROGRAMIMI PERMES SOKETEVE**
   1. **UDP-Protokolli i te Dhenave te Perdoruesit**

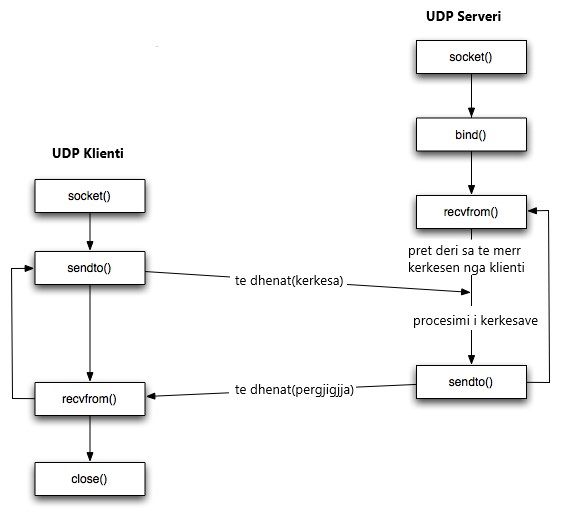
**UDP**(User Datagram Protocol-Protokolli i Datagram e Perdoruesit) është një nga protokollet kryesore në grupin e protokolleve te internetit. Me UDP, programet dhe pajisjet kompjuterike mund të dërgojnë mesazhe, të referuara si të datagrame(dhena), te hostet e tjera që janë të identifikuara me IP.

Përdoret,posaçërisht për aplikacione të ndjeshme si lojërat, luajtja e videove ose kërkimet e Sistemit të Emrit të Domenit (DNS). UDP rezulton në komunikim më të shpejtë sepse nuk kalon kohë duke formuar një lidhje të fortë me destinacionin para transferimit të të dhënave sikurs TCP.

UDP vjen me kontrolle, të cilat kanë për qëllim të sigurojnë integritetin e të dhënave dhe numrat e portave, të cilët ndihmojnë në dallimin e rolit që luajnë të dhënat në burim dhe destinacion.

Ne krahasim me TCP ka disa të meta sepse pas identifikimit të një pajisjeje edhe pas dërgimit të të dhënave, nuk ekziston diqka e tillë për të treguar rendin me të cilin duhet të mbërrijne të dhënat apo edhe një njoftim nëse të dhënat arritën në destinacionin e caktuar ne fillim.

Për këtë arsye edhe është e ekspozuar ndaj rrjeteve të ndryshme dhe sulmohet.

* 1. **UDP-Modeli i komunikimit përmes soketave Klient/Server** 

1.Serveri pret deri sa të merr kërkesën nga klienti 2.Klienti lexon kërkesat nga tastiera dhe e dergon te serveri 3.Serveri lexon kërkesën e marr nga Klienti dhe e proceson varësishte se qfarë kërkon kërkesa 4.Serveri dergon përgjigjen e kërkesës 5.Klienti e lexon përgjigjën e pranuar 6.Klienti mbyllet

* 1. **UDP-Klienti**

Kodi për implementimin e soketit tek UDP-Klienti. Kodi i plotë gjendet në fjallin UDP-Klienti.py.

#Programimi i soketit FIEK Klienti permes protokollit TCP

import socket #Importi i te gjitha moduleve nga libraria socket

server\_name = '127.0.0.1'#IP Adresa e localhostit

server\_port = 14000 #NUMRI I PORTIT te serverit

#Funksioni/Kushtet per nderrim te portit

#.....

soketiK = socket.socket(socket.AF\_INET,socket.SOCK\_DGRAM)#Ky rresht krijon soketin e klientit te quajtur soketiK dhe permban dy parametra.

#Parametri i pare AF\_INET-tregon se rrjeti perdore IPv4

#Parametri i dyte SOCK\_DGRAM-tregon se soketi eshte i tipit SOCK DGRAM qe nenkupton qe oshte UDP soket

print('---------------------------SOKETI UDP\_Klienti-------------------------')

#........

kerkesa = input("I/e nderuara klient/e ju sapo jeni lidhur me serverin,ju lutem shkruani kerkesen tuaj: ")

while True:

    #Dergimi i kerkeses tek serveri(gjegjesishte ne localhost me portin qe e kemi percaktuare tek serverPort)

    soketiK.sendto(kerkesa.encode(), (server\_name, server\_port))

    pergjigjja = soketiK.recv(128).decode()#Default decoding=utf-8

    print(pergjigjja) #Printimi i pergjigjes

soketiK.close()

* 1. **UDP-Serveri**

Kodi për implementimin e soketit tek UDP-Serveri. Kodi i plotë gjendet në fjallin UDP-Serveri.py.

#Importimi i librarive te nevojshme

import socket

#.....

import threading

server\_name = '127.0.0.1'#IP Adresa e localhostit

server\_port = 14000 #NUMRI I PORTIT te serverit

soketiS = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)#Ky rresht krijon soketin e klientit te quajtur soketiK dhe permban dy parametra.

#Parametri i pare AF\_INET-tregon se rrjeti perdore IPv4

#Parametri i dyte SOCK\_DGRAM-tregon se soketi eshte i tipit SOCK DGRAM qe nenkupton qe oshte UDP soket

soketiS.bind((server\_name, server\_port))#Percaktimi i serverName dhe serverPort per lidhje

print('Serveri eshte startuar ne localhost ne portin ' + str(server\_port))

print('----------------------------UDP\_Serveri--------------------------')

#..........

print('Server eshte duke punuar dhe eshte duke pritur per ndonje kerkese!')

#Funksionet

#.........

#Thread

while True:

    print('-----------------------------------------------------------------')

    pergjigjjja, address = soketiS.recvfrom(128)#Kerkesa,kontollimi i bajtave te mesazhit varesisht kerkeses dhe procesimi i tyre

    print('Klienti eshte lidhur me server %s ne portin %s ' %address)

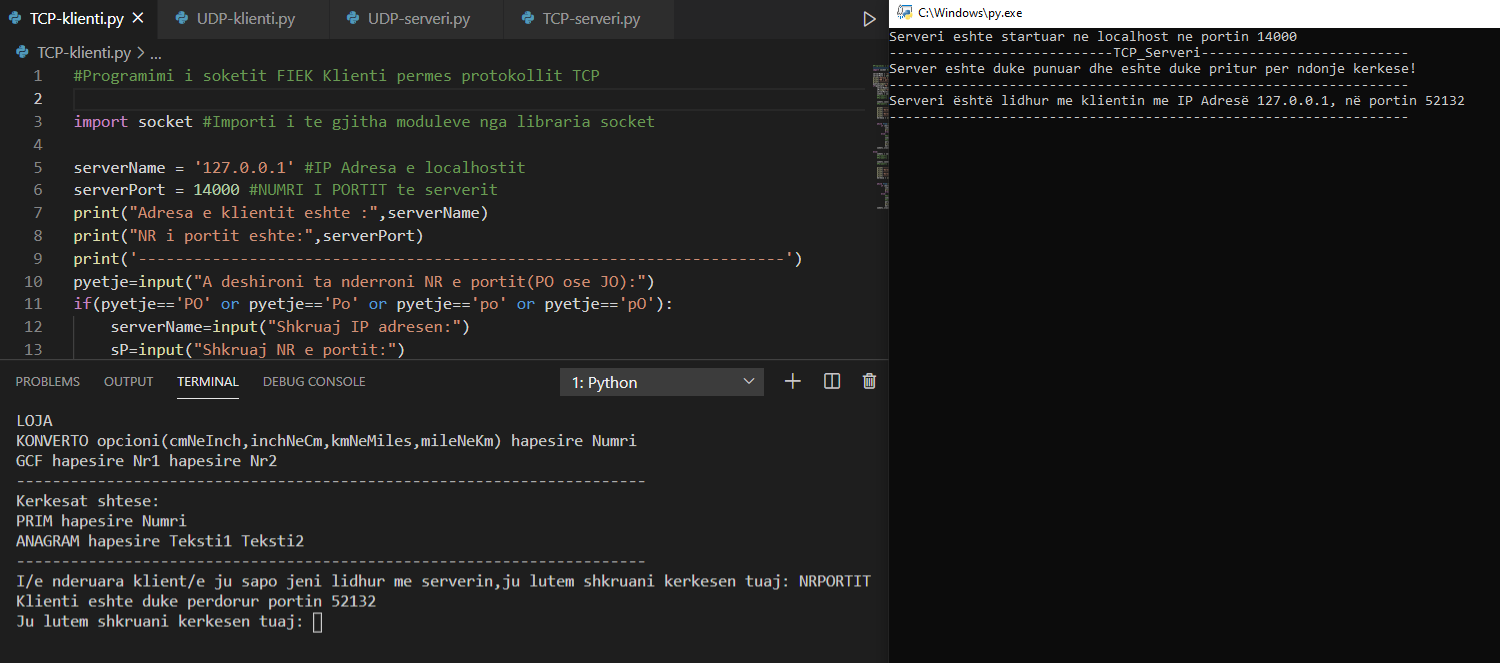
    print('Kerkesa nga klienti: ' + str(pergjigjjja.decode('utf-8')))#Kerkesa e klientit

    threading.\_start\_new\_thread(ThreadUDP,(pergjigjjja,address,))#Krijimi i thredave

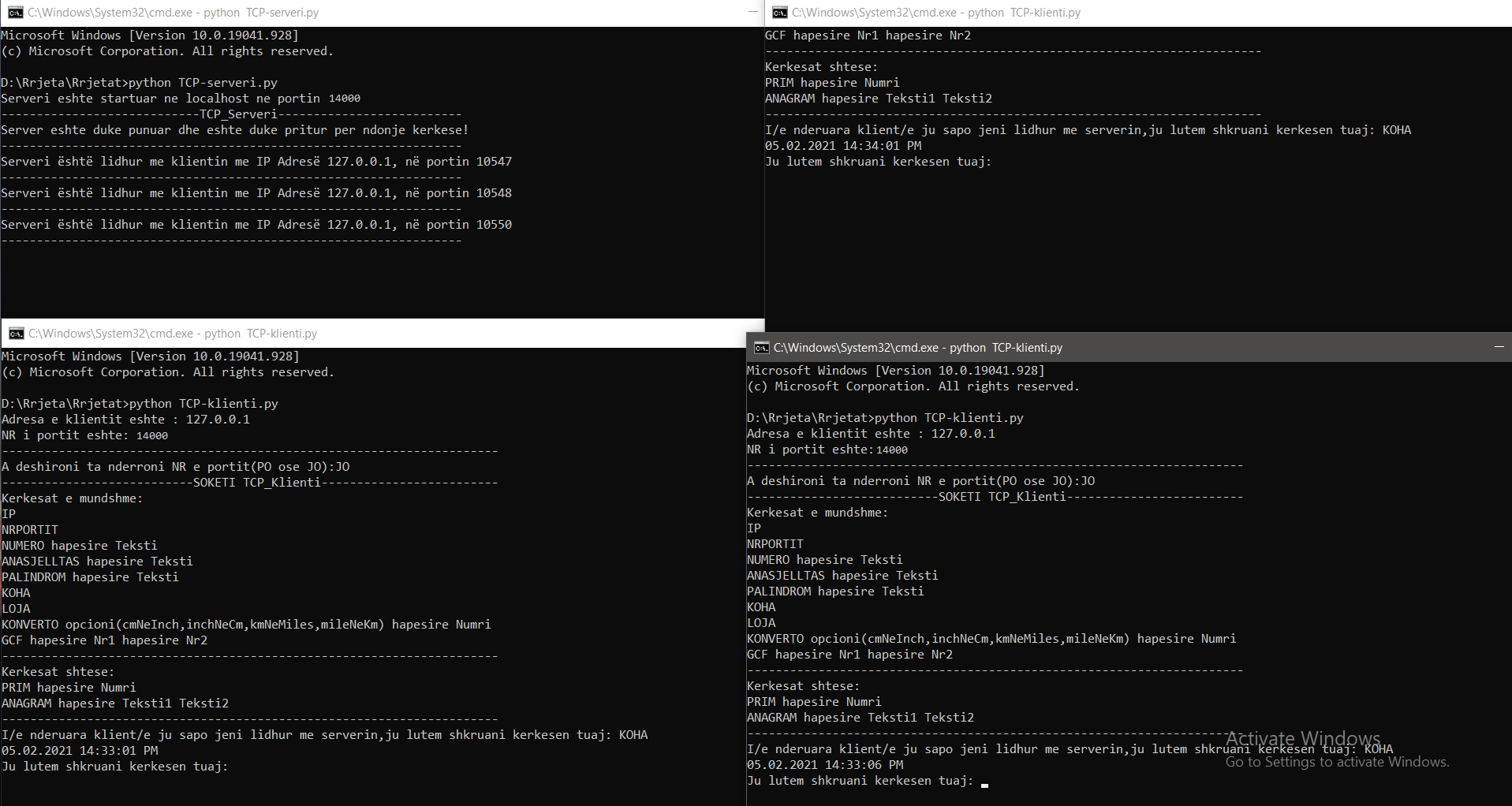
soketiS.close()

1. **Testimi i programit klient/server ne TCP dhe UDP**

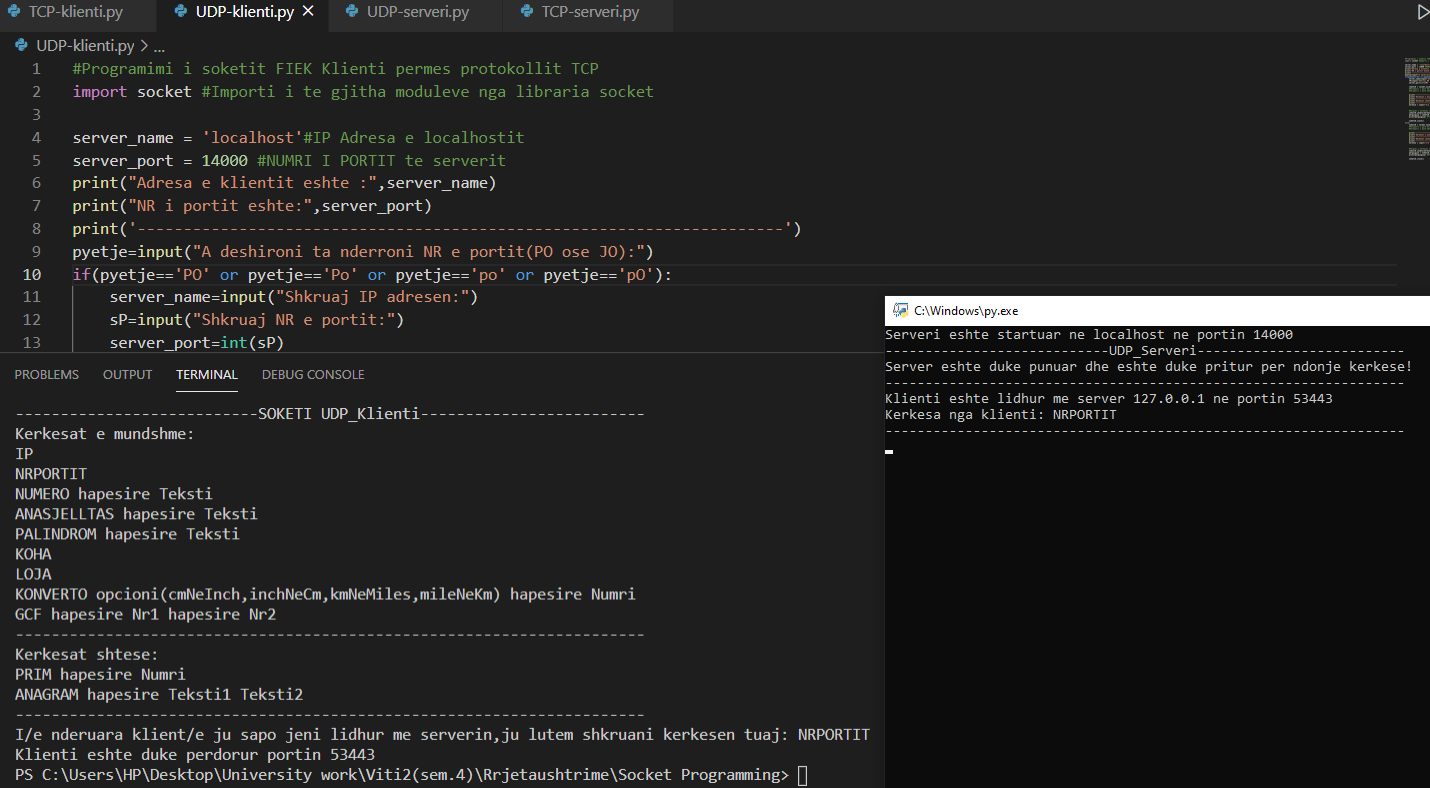
TCP Klient/Server



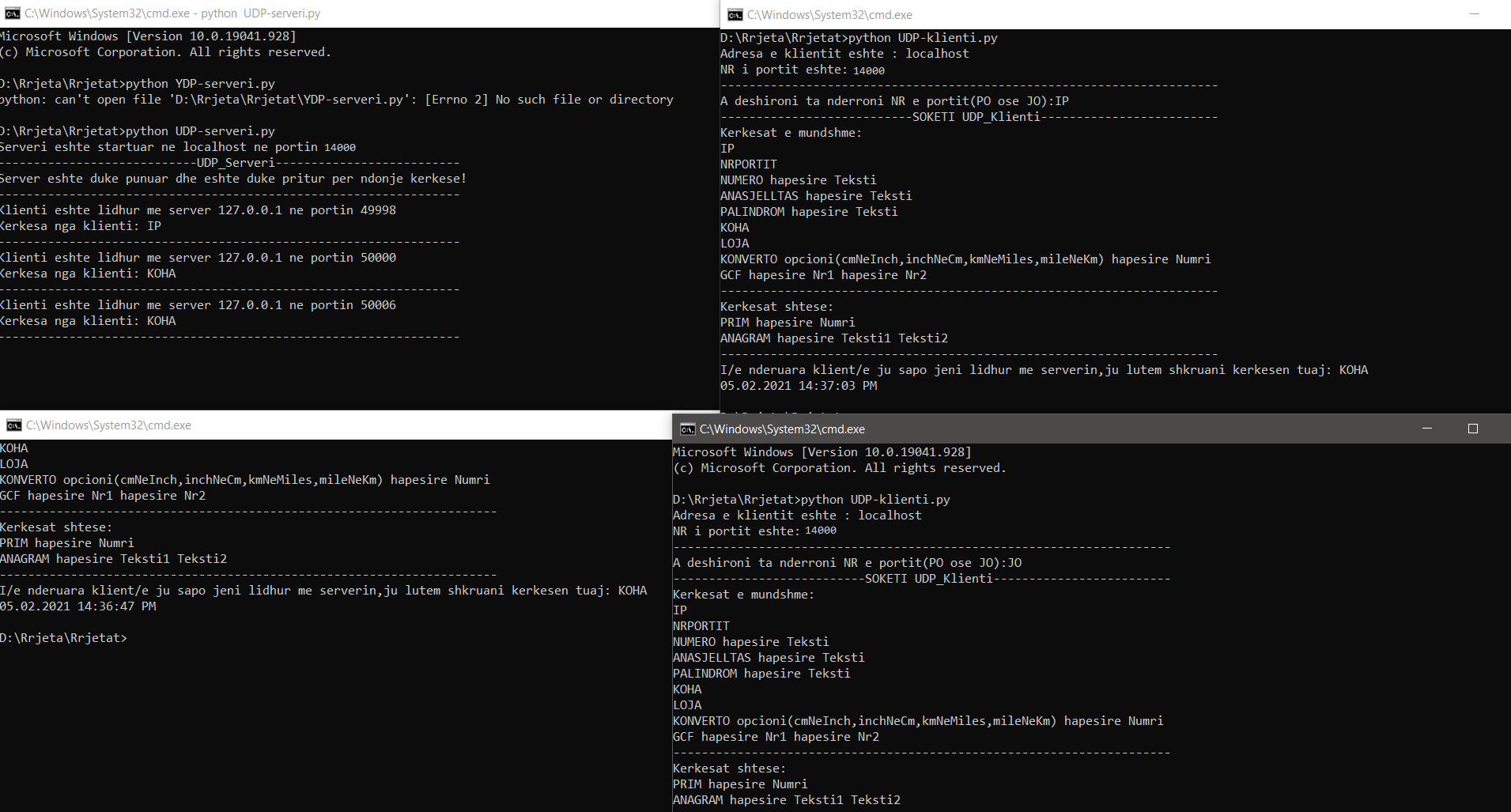
TCP Klient/Server permes multithread



UDP Klient/Server



UDP Klient/Server permes multithreads



1. **Perfundimi**

Realizimi i gjithe projekti ishte mjafte sfidues gjate gjithe kohes por njekohesisht edhe argetues.Ngase per implementimin e nje programi te tille komunikues vetem ushtrimet ne lenden Rrjetave Kompjuterike nuk kane mjaftuar . Andaj eshte dashur te hulumtojme dhe analizojm funksione te ndryshme ne baze te kerkesave te projektit. Dhe e gjithe kjo ka rezultuar ne perforcimin me te madh te koncepteve ndaj arkitektures klient/server,qasjes permes soketeve,dergimin-pranimin e kerkesave si dhe procesimin e tyre.

Programi i implementuar me larte realizon komunikimin mes Klientit dhe Serverit .Te gjitha metodat e larte permendura funksionojn ne menyren e duhur. Poashtu edhe implementim tek serveret TCP/UDP per te komunikuar me disa kliente njekohesishte permes threads eshte funksional.

1. **Referencat**

* Rajkumar Buyya,“Socket Programming”.
* Joseph M. Dibella ,“Socket Programming”.
* James F. Kurose, Keith W. Ross, “Computer Networking”.
* David Ascher, “Learning Python”.
* <http://en.wikipedia.org/wiki/Berkeley_sockets>.
* <https://en.wikipedia.org/wiki/User_Datagram_Protocol>
* <https://en.wikipedia.org/wiki/Transmission_Control_Protocol>
* <https://www.tutorialspoint.com/python/index.html>
* <https://www.tutorialspoint.com/socket-programming-in-python>
* <https://www.geeksforgeeks.org/python-programming-language/>
* <https://www.w3schools.com/python/default.asp>
* <https://www.javatpoint.com/python-functions>
* <https://www.learnpython.org/en/Functions>
* <https://www.pitt.edu/~naraehan/python3/user_defined_functions.html>
* <https://www.programiz.com/python-programming>
* <https://stackoverflow.com/questions/tagged/python>
* <https://www.codementor.io/community/topic/python>
* <https://diveintopython3.net/strings.html>
* <https://www.techbeamers.com/python-tutorial-write-multithreaded-python-server/>
* <https://www.tutorialspoint.com/python3/python_multithreading.htm>