

Universiteti i Prishtinës

Fakulteti i Inxhinierisë Elektrike dhe Kompjuterike

Lënda: Rrjetat Kompjuterike – 2017

Projekti 1:

Dizajnimi Klient – Server duke përdorur Socketët (ang. Sockets)

UDP dhe TCP

Studentja: Vegim Karaçica

Asistenti : Msc. Edon Mustafa

Data : 18/04/2017

Gjate zhvillmit te projektit jane perdorur vegla Visual Studio(2015) me gjuhen programuese dhe version: Python 2.7.13.

Sistemi operative është përdorur Windows 10 (Home) 64-bit.

Lista e metodave:

**IP, PORT, ZANORE, PRINTO, HOST, TIME, KENO, KONVERTO, FAKTORIEL , FLOATNUMBER DHE ASCII**

Projekti është i koncentruar ne krijimin e UDP(server/client) si dhe të TCP(server/client).

Le t’i shtrojmë pytjet se cilat jan dallimet dhe ngjashmerite?!

Ngjamerit ne mes UDP dhe TCP nuk ka diqka esencial sepse ne fakt ka shum dallime ne mes ketyre në aspektin e mënyrës së veprimit të tyre.

Tek TCP-ja kemi siguri ne dergim informacioni, marrja parasysh e voneses(mundesisht sa ma pak vones delay🡪0), ndërsa tek UDP-ja nuk i kushtohet vëmendje disa bitave te humbur kohe pas kohe dhe integriteti nuk është i garantuar.

Objektivat e Projektiti janë:

* Të kuptuarit e Socket-ave
* Të kuptuarit e një programi që funksionon në rrjetë
* Njohja dhe familjarizimi me modulet në Python të cilat ndihmojnë në realizim të komunikimit në rrjetë me Socket-a.
* Kodimi i një klient aplikacioni që komunikon në rrjetë.
* Çka është serveri
* Të kuptuarit e arkitekturës klient/server
* Kodimi i një server aplikacioni që komunikon në rrjetë përmes protokollit UDP

Kur një aplikacioni i nevoitet të lidhet në rrjetë lokale (intranet) ose në internet, ai aplikacion e përdorë komponentën softuerike të quajtur *Socket*. *Socket* i hap lidhjet ne rrjetë për t’i lejuar aplikacionet që të mund të shkruajnë dhe të lexojnë nëpermjet rrjetës.

*Socket* janë një pikë fundore (ang. Endpoint) e komunikimit në mes të dy programeve në rrjetë. Një *Socket* perbehet prej numrit te portit (ang. port number) dhe prej një IP adresë dhe këto dy komponente e identifikojnë në mënyrë unike *Socket*-in.

Gjatë shkuarjes së kodeve për programet që komunikojnë në rrjetë, programerët duhet të vendosin se me cilin protokol të komunikimit do të komunikojnë aplikacionet. Kemi dy protokolle të cilat mund ti përdorim: TCP dhe UDP. TCP është protokoll që siguron një kanal të besueshëm të komunikimit në mes klientit dhe serverit ndërsa protokolli UDP nuk siguron nje kanal të tillë dhe rrjedha e informatave bëhet pa garancionin se ato do të arrijnë në destinacion. Ne gjatë këtij ushtrimi do të përdorim protokollin UDP.

Kur një klient-aplikacion ekzekutohet ai krijon një proces për komunikim. Ngjashëm ndodhë edhe në një server-aplikacion. Kështu komunikimi në mes të ketyre dy proceseve bëhet duke dërguar mesazhe në *Socket* dhe përmes tyre bëhet komunikimi me procesin tjeter i cili ekzekutohet në *host*-in tjetër.

UDP

Klient

Socket

UDP

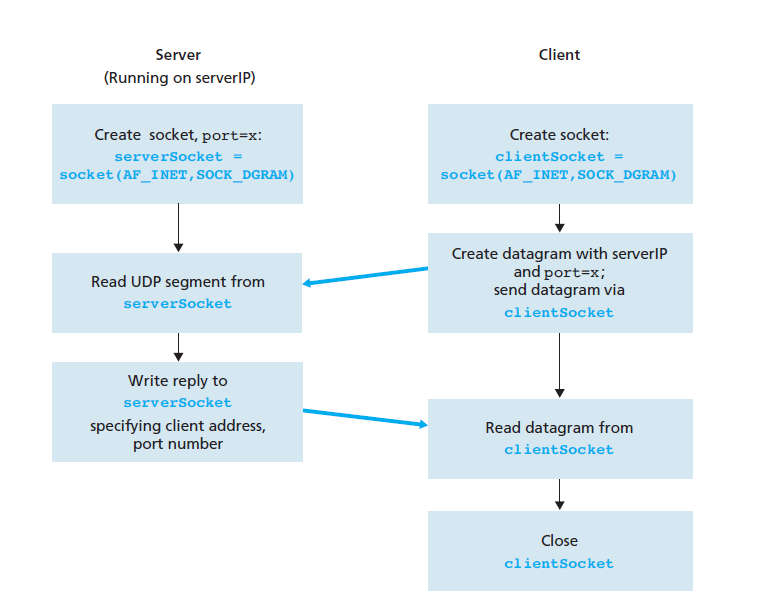
Server

Socket

Interneti

Kodimi i një klient/server aplikacioni ne Python:

1. Klienti lexon një fjali nga hyrja standarde (tastiera) dhe e dërgon këtë fjali në socket per ta dërguar më tutje të serveri.
2. Serveri e lexon fjalin nga lidhja që krijon me socket-in.
3. Serveri e konverton fjalin në shkronja të mëdha.
4. Serveri e dërgon fjalin e modifikuar në socket për ta dërguar drejt klientit.
5. Klienti e lexon fjalin e modifikuar nga soket-i i saj dhe e shfaq në monitor.



Klient – Server aplikacioni duke përdorur UDP

**Serveri** njëjtë sikurse edhe kompjuterët e tjerë është një makinë procesorike e fuqishme. Detyrat e kësaj makinë janë që të shërbejnë klientët. Si punon arkitektura klient/server? Klienti e bënë kërkesën dhe kështu kërkesa dërgohet për tek serveri, pastaj serveri e merr atë kërkesë dhe e proceson. Pas procesimit të kërkesës serveri kthen përgjigjen tek klienti.

Pjesa e zhvillimit të serverin:

1. Serveri e lexon fjalinë nga lidhja që krijon me klientin nëpermjet socket-it.
2. Serveri e konverton fjalinë në shkronja të mëdha.
3. Serveri e dërgon fjalinë e modifikuar në socket për ta dërguar drejt klientit.

Për dallim nga UDP, TCP është protokol i bazuar në lidhje. Kjo nënkupton që klienti dhe serveri paraprakisht duhet të krijonë në lidhje, para se të shkëmbejnë të dhëna. Pasi të vendoset kjo lidhje të dhënat shkëmbehen përmes saj.

Krijimin e lidhjes e inicion (fillon) klienti. Paraprakisht serveri duhet të jetë i ekzekutuar dhe i gatshëm (ngjashëm sikurse përmes UDP-së). Për dallim me UDP-në, tek TCP në pjesën e serverit përdoren dy (2) socke-ta:

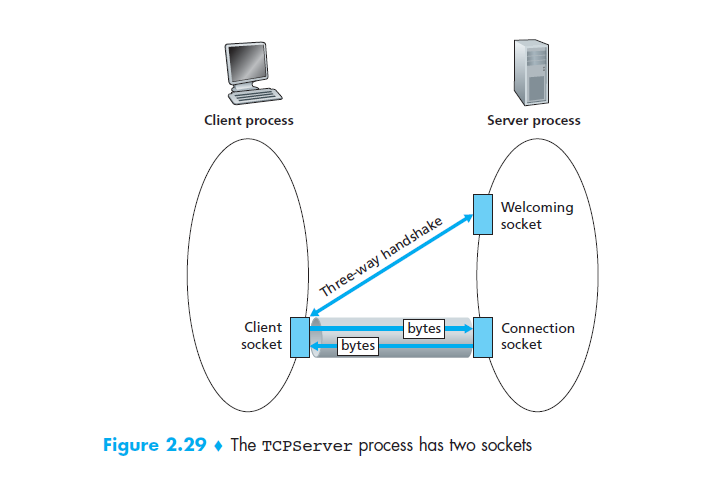
1. një socket special, i dedikuar për vendosje të lidhjes
2. një socket për komunikim

Procedura e vendosjes së lidhjes me serverin njihet si three-way handshake dhe siq u tha iniciohet nga klienti. Kjo procedurë zhvillohet në shtresën e Transportit në mënyrë automatike dhe gjatë programimit të aplikacionit klient dhe server nuk merremi me këtë.

Procedura është ilustruar në figurën 1 ku shihet vendosja e lidhjes dhe pastaj shkëmbimi i të dhënave mes klientit dhe serverit[[1]](#footnote-1).

Pra Client socket (në kodin e më poshtëm clientSocket) komunikon me Welcoming socket (serverSocket në kod), i cili pastaj krijon një Connection socket (connectionSocket në kod) dhe komunikimi vazhdon në mes të Client Socket dhe Connection Socket.

Kur një program ekzekutohet, sistemi operativ në të cilin ekzekutohet krijon një proces. Prandaj në figurë dhe më herët i referohemi si komunikim në mes të proceseve. Socket janë njëra nga metodat për të realizuar komunikimin ndërmjet procesve, veçanërisht në rrjeta kompjuterike, ku kompjuterët janë të lidhur përmes rrjetës globale të kompjuterëve (Internetit).



Skenari mbetet i ngjashëm sikurse në mes të UDP klientit dhe serverit:

1. Klienti lexon një fjali nga hyrja standarde (tastiera)
2. Serveri e lexon fjalin nga lidhja që krijon me socket-in.
3. Serveri e konverton fjalin në shkronja të mëdha.
4. Serveri e dërgon fjalin e modifikuar në socket
5. Klienti e lexon fjalin e modifikuar nga soket-i i saj

Përmbajtja:

**Libraritë e Python-it1.0.0**

Socket1.1.0

AF\_INET1.1.1

SOCK\_Stream/DGRAM1.1.2

Socket.bind()1.1.3

Socket.listen()1.1.4

Socket.connect()1.1.5

Socket.accept()1.1.6

Socket.recvfrom()/recv()1.1.7

Socket.sendto()/send()1.1.8

Socket.close()1.1.9

Datetime1.2.0

Datetime.datetime()/now()1.2.1

Random1.3.0

Sys1.4.0

Decimal1.5.0

**Metodat themelore python**2.0.0

Input()/raw\_input()2.1.0

Print()2.2.0

String2.3.0

List/Dictionary2.3.1

Length()2.3.2

Split()2.3.3

Find()/Replace()2.3.4

Upper()/Lower()2.3.5

Convert2.3.0

StringToInt2.3.1

StringToDecimal2.3.2

StringToFloat2.3.3

StringToBin2.3.4

StringToHex2.3.5

**Metodat e krijuara3.0.0**

IP3.1.0

PORT3.2.0

ZANORE3.3.0

PRINTO3.4.0

HOST3.5.0

TIME3.6.0

KENO3.7.0

KONVERTO3.8.0

CelsiusToKelvin 3.8.1 CelsiusToFahrenheit 3.8.2

KelvinToFahrenheit 3.8.3

KelvinToCelsius 3.8.4

FahrenheitToCelsius 3.8.5

FahrenheitToKelvin 3.8.6

PoundToKilogram 3.8.7

KilogramToPound 3.8.8

FAKTORIEL3.9.0

FLOATNUMBER3.10.0

ASCII3.11.0

Libraritë e Python-it:

Import Socket:

Kur një program ekzekutohet, sistemi operativ në të cilin ekzekutohet krijon një proces. Prandaj në figurë dhe më herët i referohemi si komunikim në mes të proceseve. Socket janë njëra nga metodat për të realizuar komunikimin ndërmjet procesve, veçanërisht në rrjeta kompjuterike, ku kompjuterët janë të lidhur përmes rrjetës globale të kompjuterëve (Internetit).

AF\_INET:

Përcakton familjen e adresave, në rasin tone IPv4

SOCK\_STREAM/DGRAM:

Në këtë rast e përcaktojm se a është me tip transporti TCP apo UDP

Socket.bind():

Përmes këtij rreshti i caktojmë në mënyrë eksplicite portin me numër 9000 socket-it të serverit. Thonjëzat e njëfishta nënkuptojnë që ky socket është i çasshëm nga çfardo adrese që e ka serveri. Në këtë mënyrë kur dërgohet ndonjë paketë në portin 9000 në IP adresën e serverit, paketa do të drejtohet për tek ky socket. Pastaj serveri hyn në një loop të pafundme (while 1 ose while True) për të pranuar dhe procesuar paketat nga klientët.

Socket.listen():

Socket-i përdoret për vendosje të lidhjes (pranon kërkesat dhe krijon socket-in për komunikim).Pas krijimit serveri duhet të jetë në gjendje për pranim të kërkesave. Kjo bëhet përmes metodës listen(), kur server pret që ndonjë TCP klient të bëjë kërkesë për formim të lidhjes.

Socket.connect():

Përmes rreshtit të mësipërm klienti incion krijimi ne lidhjes me serverin, duke përdorur metodën *connect()*. Këtu specifikojmë adresën e serverit dhe portin e socket-it në server. Pasi ekzekutimit të kësaj pjese të kodit realizohet procedura *three-way handshake* dhe bëhet vendosja e lidhjes TCP (ang. TCP connection) në mes të klientit dhe serverit.

Socket.accpet():

Në momentin që vie ndonjë kërkesë nga klienti, programi thërret metodën accept() e cila krijon një socket të ri në server, pra connectionSocket i dedikuar vetpër atë klient. Pas vendosjes së lidhjes klienti dhe server mund të shkëmbejnë bajtat përmes.

Socket.recvfrom()/recv():

Metoda .recvfrom() përdoret për pranimin e të dhënave tek UDP-transporti ndersa vetem recv() tek TCP-ja.

Socket.sendto()/send():

Socket-I përdoret për dërgim të të dhënave. Sendto() tek UDP-ja perdoret ndersa send() tek TCP-ja.

Socket.close():

Pas dërgimit të mesazhit të modifikuar, e mbyllim Socket-i. Por serverSocket mbetet i hapur dhe ndonjë klienet tjetër mund të trokas në derë dhe dërgoj fjali për të modifikuar.

Datetime:

Moduli DATETIME furnizon klasa për manipulimin të datave dhe dy mënyra: të thjeshta dhe komplekse. Ndërsa data dhe koha aritmetike është e mbështetur, fokusi i zbatimit është në nxjerrjen efikase të atributit për formatimin e prodhimit dhe manipulimit.

Random:

Është library ku mundemi ta krijojmë një object në çrast nxierr një numër të rastesishëm nga [0.0 , 1,0).

Sys:

Ky modul ofron qasje në disa variablave të përdorura ose të mbajtura nga përkthyesi dhe të funksioneve që ndërveprojnë me forcë me përkthyesin. Ajo është gjithmonë në dispozicion.

Decimal:

Moduli decimal siguron mbështetje për lundrues dhjetore pikë aritmetikë. Ajo ofron disa avantazhe mbi tipn float:

**Metodat themelore python:**

Input()/raw\_input():

Funksioni input() është funksion i Python që përdoret për të kërkuar nga shfrytëzuesi të dhëna. Këtë rast mesazhin që dëshiron me dërgu në server dhe të cilin e ruajmë në variablën message.

Print():

Deklarata print është zëvendësuar me një funksion print (), me argumente, fjalen për të zëvendësuar shumicën e sintaksës të veçantë të deklaratës së shkruar (PEP 3105).

String:

Tek kjo kemi metoda për list: nga vetë fjala ka formen e një liste. Pastaj kemi metoden Length() ne python shenohet “ len() “ per numrin e karaktereve ne string. Metoda split() është për ndarje.

Find()/Replace(): janë për gjetjen e nje karakteri ose string në një string ose zevendesim.

Upper()/Lower() : perdoret per konverim të stringut në shkronja të mëdha ose të vogla.

Convert:

Ka metodat e konvertimit nga stingu ne numra dhe anasjelltas duke u specializur ne tipin e numrave, pra mundet me qenë binear, integer, decimal, float ose heksadecimal.

**Metodat e krijuara:**

***IP:***

Metoda IP zabtohet kur nga klienti dergohet metoda IP si caseSensitive.

Kthen adresen e klientit.

***PORT:***

Metoda port kthen portin e klientit dhe gjndet perms metodes së gatshme ku na mundeson socket-i.

***ZANORE:***

Kjo metode e kthen numrin e zanoreve të një teksti.

Ketu është përdorur nje liste(dictionary sipas python-it) ne menyre qe secila shkronje e tekstit te krahasohet me shkronjat(karakteret) e list-ZANORE-ve. Po ne ate unazë kemi bërë një interator(numerues). Dhe saktesisht ai numerus është numri i zanoreve.

Dhe ne fund është formatuar mesazhi që do t’i kthehet klientit.

***PRINTO:***

Tek kjo metodë kemi kerkesen dhe tekstin ku ketu eshte formatuar teksti duke u larguar “PRINTO ” nga “PRINTO teksti qe do te printohet” ne qrastë na mbetet vetëm teksti që do të kthehet duke e rritur shkronja e pare.

***HOST:***

Tek kjo metodë është përdorur try{}except socket.error{}. Ku si metodë e gatëshme nga libraria e socket-it socket.gethostname().

***TIME:***

Tek kjo metodë është përdorur metoda e gatshme now() e libraries së datetime.

***KENO:***

Kjo metodë gjeneron random 20 numrat e qfardorshëm e cila është nën unazen e një for-it.

***KONVERTO:***

Kjo metodë e kemi deklaruar si një funksion të gjerë që në vete I ka dy parametra I pari e percakton njësinë nga cili lloj në cilin dhe i dyti është numrin që dëshirojmë t’a konvertojmë.

Pra në këtë metodë vetëm e bëjmë thirrjen e funksionit në fjalë.

***FAKTORIEL:***

Kthen numrin e një faktoriel numrit. P.sh. 5! =120.

Pra: n!=n\*(n-1)\* … \* 2 \* 1 {0!=1, 1!=1}

***FLOATNUMBER:***

Bënë konvertimin e numrit me bazë10 në float pointing number.

Float pointing number ka formatin:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Shenja Eksponenti Mantisa

ShenjaЄ{0, 1} Eksponenti {0, 1}8 Mantisa {0, 1}23

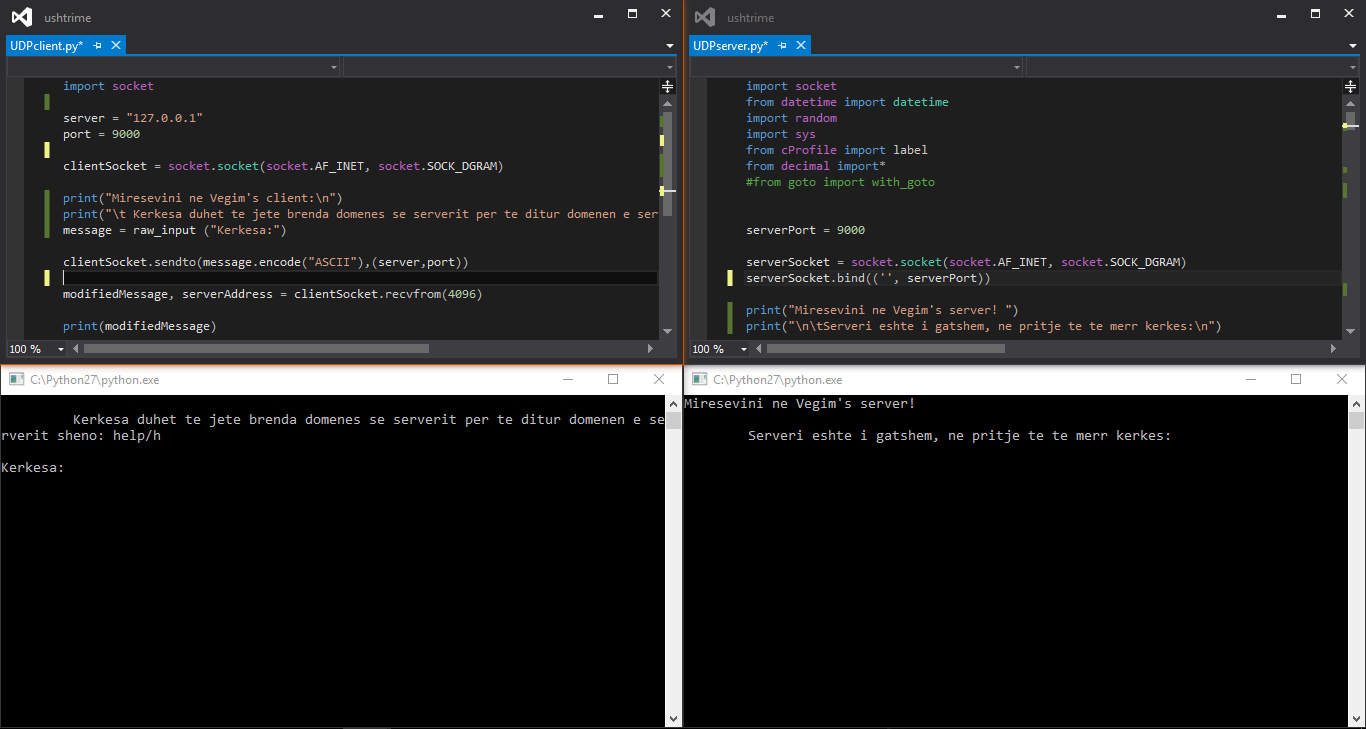
***ASCII:***

Kjo metodë çfarëdo karakteri e shëndrron në decimal pastaj në heksadecimal si dhe octal.

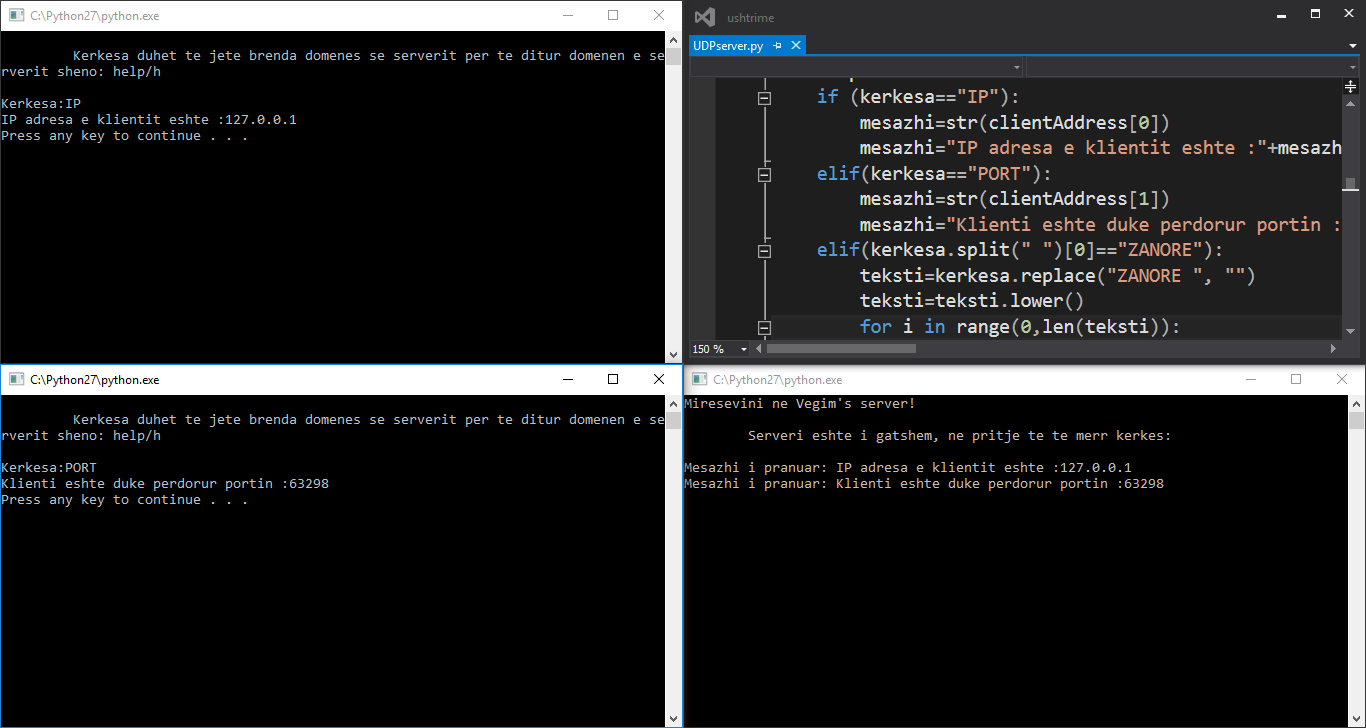
Shembull:

***Perciellja e zhvillimit te kodit:***

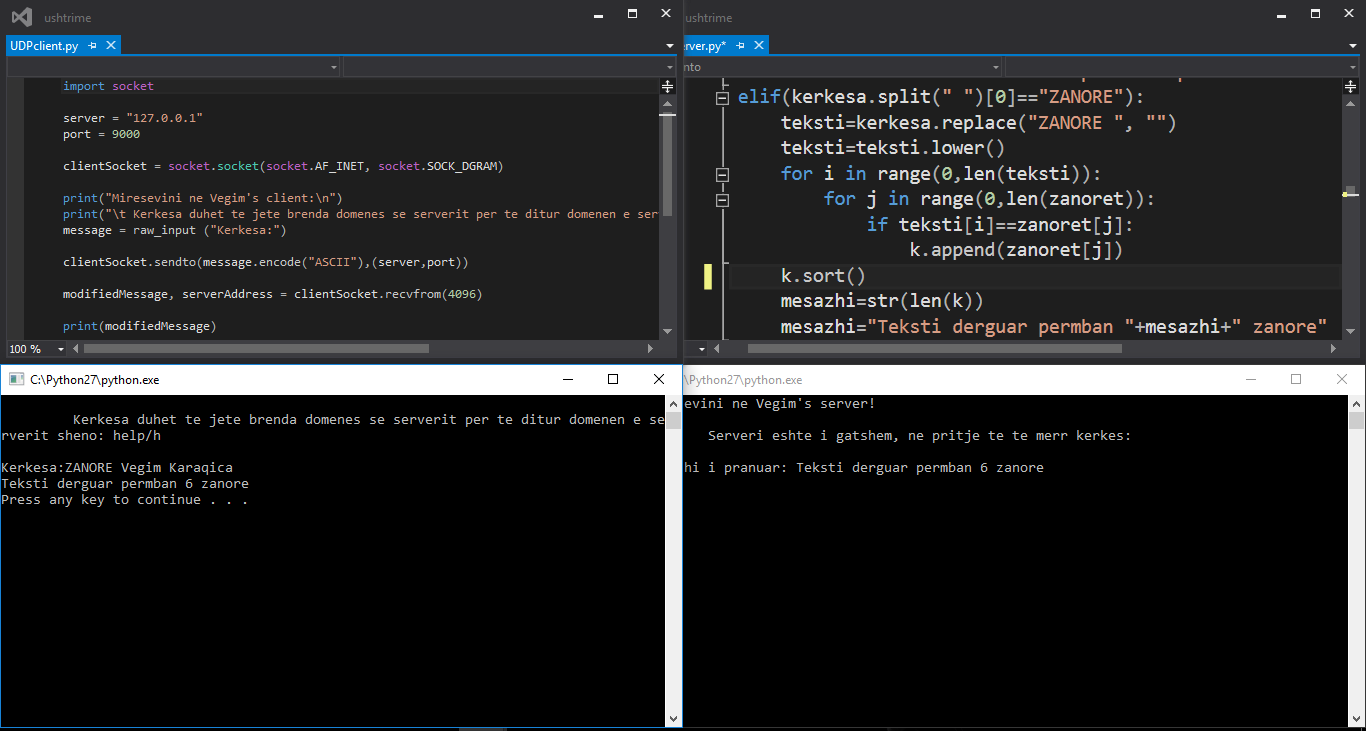
Zhvillimi i kodit te klientit dhe e serverit:



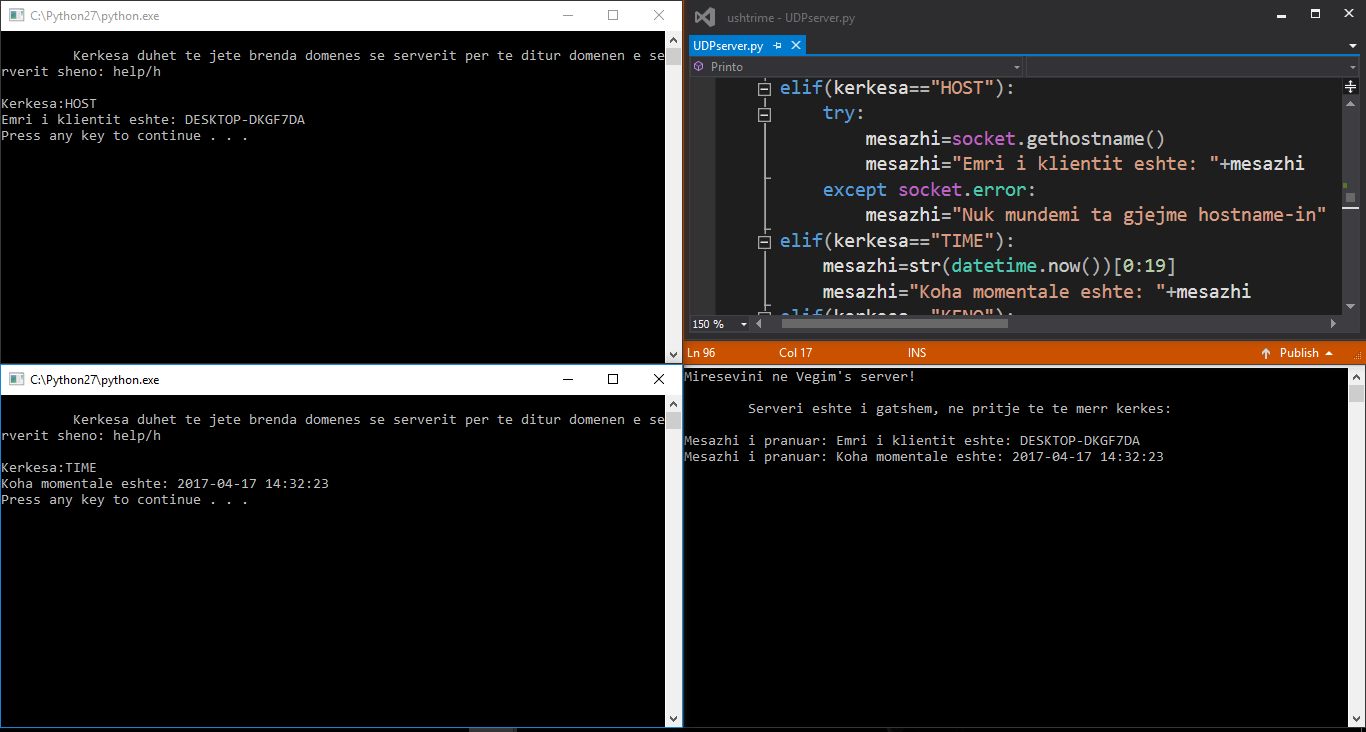
Kerkesa dhe pergjigj e kthyer nga metoda e IP dhe PORT



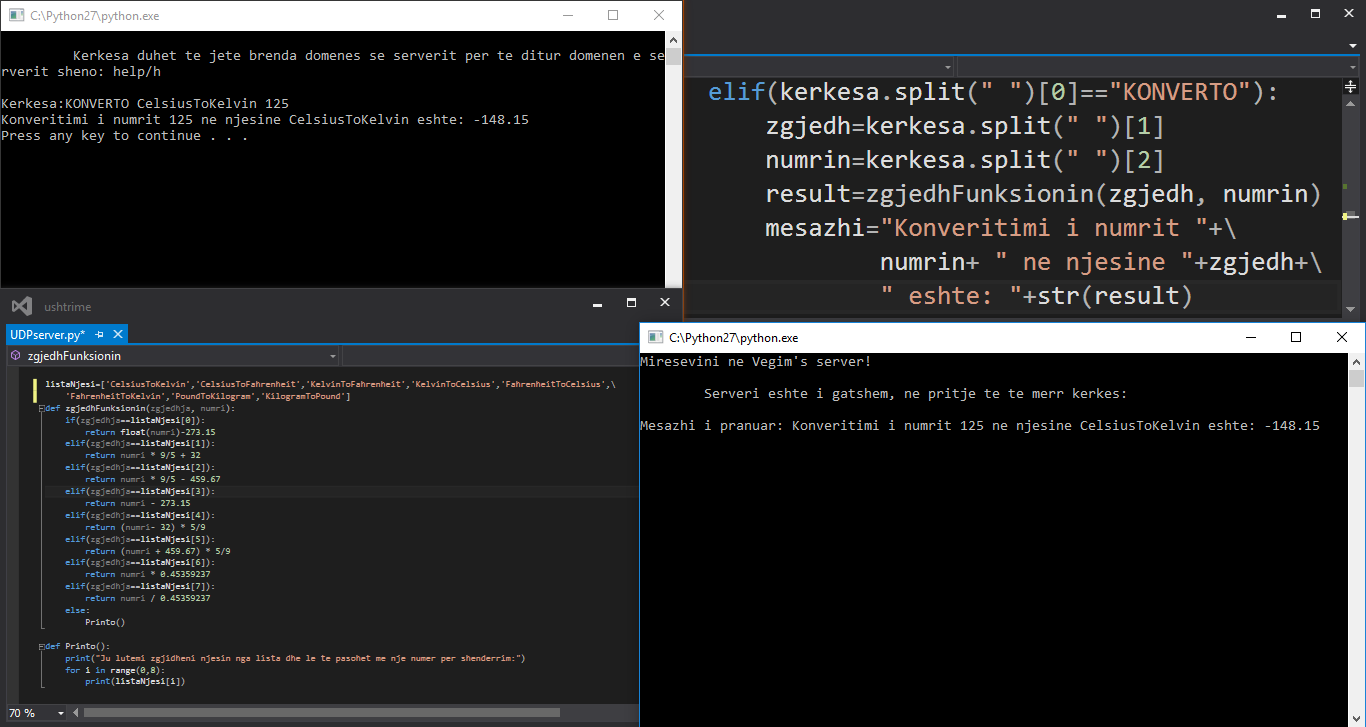
Kodi i server/client nën zhvillimin e metodes ZANORE:



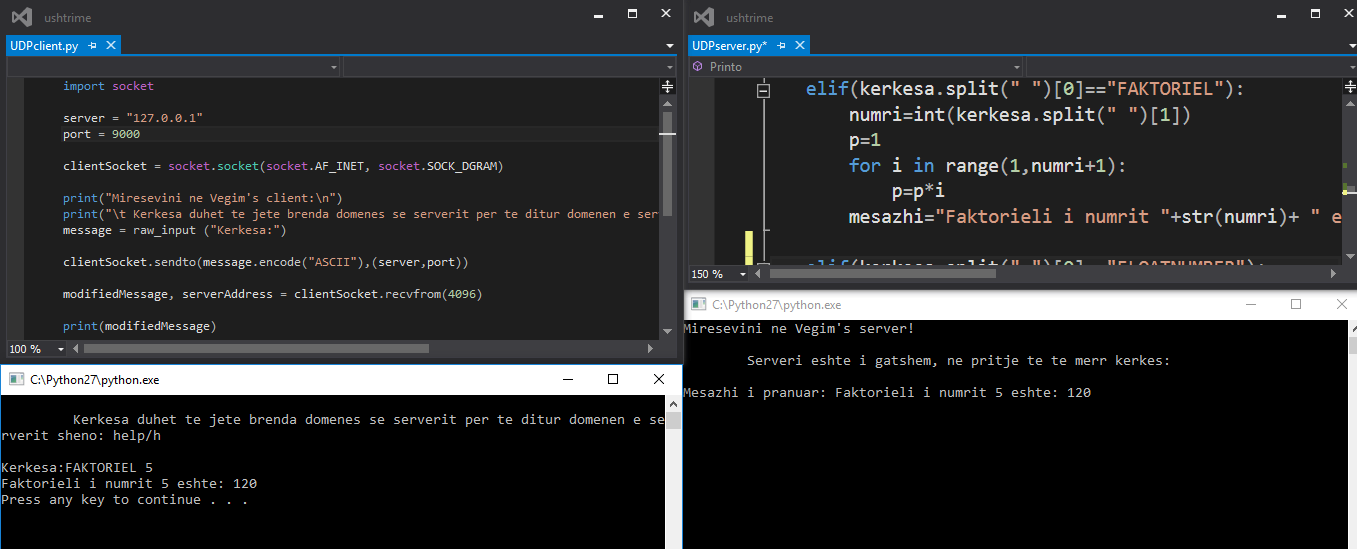
HOST metoda:



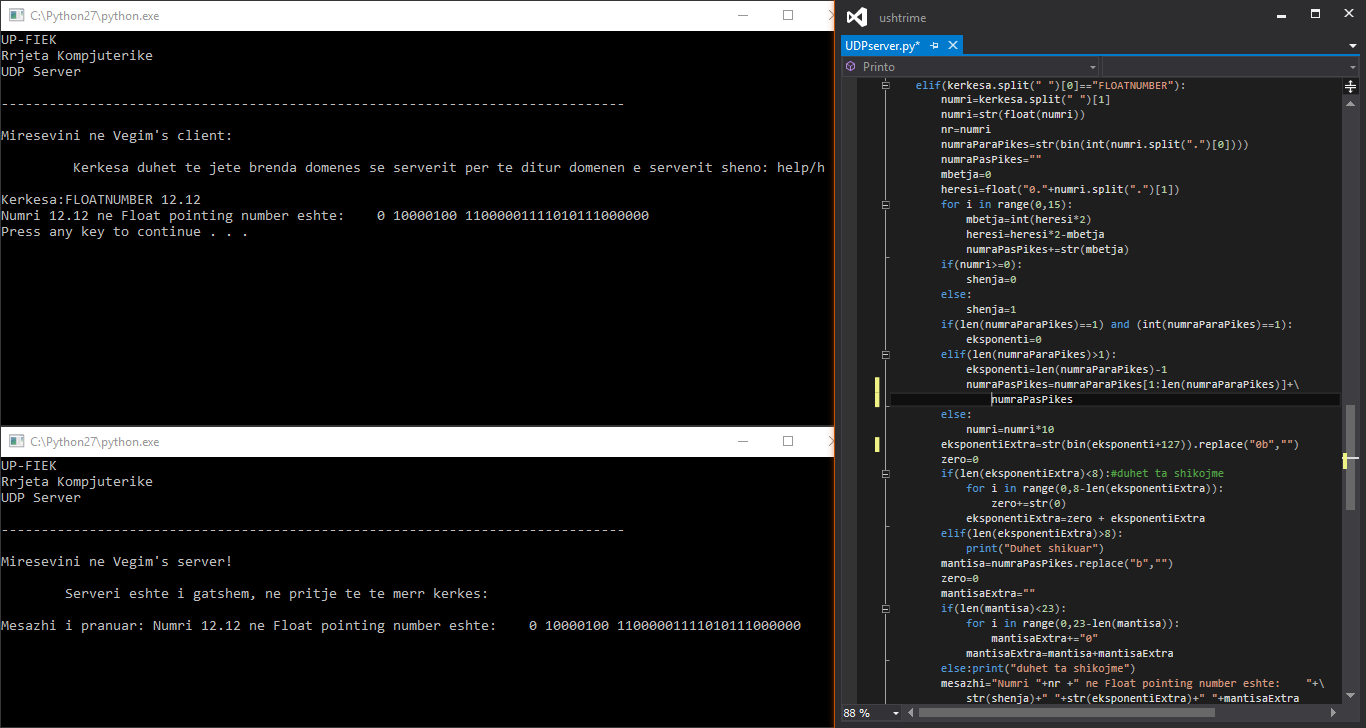
Konvertimi nga njesia🡪 njesia e “pakos” KONVERTO



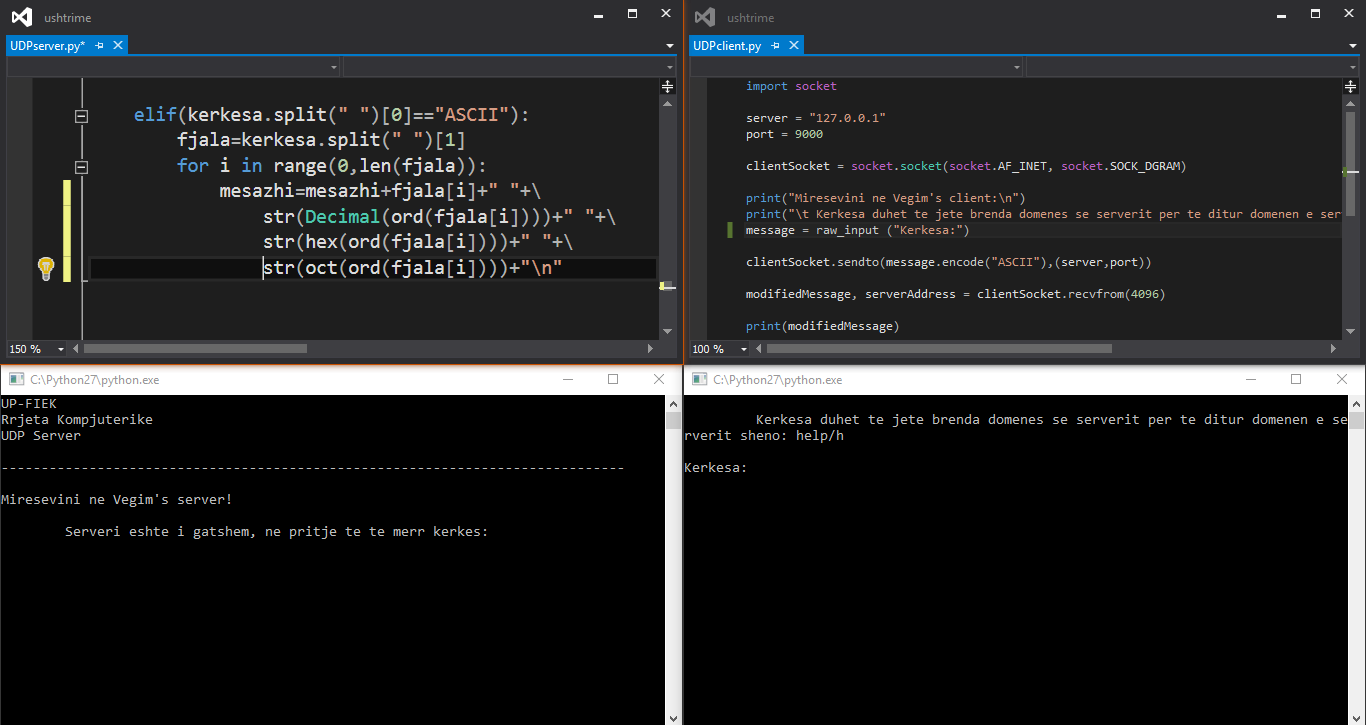
FAKTORIEL metoda e zhvilluar ne client dhe server:



FLOATNUMBER metoda e cila bënë shenderrimin e numrave sipas menyres se si tipi FLOAT ne gjuhet programuese manipulon me numra.



ASCII metoda:



1. 1. Më shumë rreth proceseve në lëndën e sistemeve operative apo mund të filloni edhe nga linku (https://en.wikipedia.org/wiki/Inter-process\_communication).

   [↑](#footnote-ref-1)