FAKULTETI I INXHINIERISË ELEKTRIKE DHE KOMPJUTERIKE

DEPARTAMENTI I INXHINIERISË KOMPJUTERIKE



Projekti 1

**Data:** 10.04.2020

**Punoi:** Uran Lajçi

**Veglat e perdorura:** Sublime Text 3, Command Prompt

**Sistemi Operativ:** Windows 10 Pro

**ID:** 18071410065

**Vendi:** Peje, Kosove

**Email:** [uran.lajci@student.uni-pr.edu](mailto:uran.lajci@student.uni-pr.edu)

**Lenda:** Rrjeta kompjuterike

**Mentoret:** Blerim Rexha, Haxhi Lajqi

**Metodat:**

1. IPADDRESS
2. PORT
3. COUNT
4. REVERZ
5. PALINDROME
6. TIME
7. GAME
8. GCF
9. CONVERT
10. ANAGRAM
11. TITLE

# Hyrje

Ky projekt ka të bwjw me komunikimin midis një serveri dhe një klienti me anë të programimit me soketa. Soketat mundësojnë komunikimin midis dy proceseve në të njëjtën makinë apo në makina të ndryshme. Kur dy programet që e përbëjnë një Network Application ekzekutohen krijohen një proces i klientit dhe një proces i serverit.

Interesant që ky projekt të punohet dhe të diskutohet është se me anë të tij e kuptojmë krijimin e Network Application-eve dhe komunikimin midis një serveri dhe një klienti.

Sfida qëndron tek kuptimi i kodeve të shkruara në server dhe në klient, kuptimi si funksionon programimi me soketa, dallimi midis protokolleve TCP dhe UDP si dhe implementimi i metodave të ndryshme.

Problemi zgjidhet duke e krijuar lidhjen midis serverit dhe klientit dhe pastaj shkruarjen e metodave në server. Se pari jepet një input në klient që na tregon kërkesen e klientit dhe të dhënat me të cilat duhet të manipulohet, ky input enkodohet dhe dërgohet tek serveri. Në server me atë input kryhen funksionet e kërkuara.

Në këtë projekt kemi mundësuar kryerjen e 11 funksioneve të ndryshme. Këto metoda kryejnë manipulime me tekst, numra dhe kthejnë rezultate.

Përmbajtja

[Hyrje 2](#_Toc37445259)

[Lidhja midis TCPServerit.py dhe TCPKlientit.py 4](#_Toc37445260)

[Lidhja midis UDPServerit.py dhe UDPKlientit.py 4](#_Toc37445261)

[Sqarimi i metodave 5](#_Toc37445262)

[Metoda IPADDRESS 5](#_Toc37445263)

[Metoda PORT 5](#_Toc37445264)

[Metoda COUNT 5](#_Toc37445265)

[Metoda REVERZ 6](#_Toc37445266)

[Metoda PALINDROME 7](#_Toc37445267)

[Metoda TIME 7](#_Toc37445268)

[Metoda GAME 7](#_Toc37445269)

[Metoda GCF 8](#_Toc37445270)

[Metoda CONVERT 8](#_Toc37445271)

[Metoda ANAGRAM (metoda e parë shtesë) 9](#_Toc37445272)

[Metoda TITLE (metoda e dytë shtesë) 9](#_Toc37445273)

[Konkluzionet 10](#_Toc37445274)

# Lidhja midis TCPServerit.py dhe TCP Klientit.py

Për të komunikuar me anë të protokollit TCP njëherë duhet që të krijohet lidhja midis serverit dhe klientit.

Që të konektohemi në një server me anë të soketave duhet që së pari të importojmë librarinë e soketave (import socket). Pastaj e krijojmë një instancë të soketës me dy parametra. Parametri i parë është AF\_INET që i referohet adresës së familjes ipv4, kurse parametri i dytë është SOCK\_STRAM që do të thotë se lidhja është bërë përmes protokollit TCP. Libraria socket e ka një funksion sendall që na lejon të dërgojmë të dhëna te klienti.

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

Te klienti duhet që të importojmë librarinë e soketave, pastaj të krijojmë një objekt të soketës dhe të rezervojmë një port në makinën tonë. Objekti listen na e tregon sa kërkesa duhet të mbahen në pritje nëse serveri është i nxënë.

Kur të krijohet lidhja me anë të soketave prej klientit e dërgojmë një kërkesë të enkoduar në serever. Serveri e pranon kërkesën dhe në bazë të kushteve vendos se a është mesazhi valid dhe kryen funksione të caktuara dhe i kthen përgjigjie të enkoduar. Atë përgjigje nëse është valide klienti e dekodon dhe e printon rezultatin.

# Lidhja midis UDPServerit.py dhe UDPKlientit.py

Për ndryshe nga TCP protokolli, UDP protokolli është protokoll connectionless dhe non-stream. Kjo do të thotë se UDP serveri mund të pranoj kërkesa pa garantu që do të kthejë përgjigje.

Te UDP klienti duhet që së pari të shkruajmë modulin e soketës ( from socket import \* ) që na mundëson krijimin e soketave në programin tonë. Pastaj duhet që të krijojmë soketn e klientit.

clientSocket = socket(AF\_INET, SOCK\_DGRAM)

SOCK\_DGRAM na tregon se kemi UDP soket e jo TCP. Mesazhi i enkoduar dërgohet te serveri ku pastaj mirë një përgjigje e cila dekodohet dhe printohet.

Te UDP serveri ajo që ndryshon prej UDP klientit është rreshti i kodit i cili ia jep numrin e portit soketit te serverit.

serverSocket.bind((’’, serverPort))

Pastaj në server merret një mesazh nga klienti me të cilin kryhen funksione të ndryshme dhe kthehet përgjigjia e enkoduar për klientin.

# Sqarimi i metodave

Metodat janë të implementuara në server.

## Metoda IPADDRESS

Kjo metodë e kthen IP adresën e klientit. Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

Te UDPServeri

def IDADDRESS(address):

ip\_address = address[0]

serverS.sendto(("Përgjigjja: IP Adresa e klientit eshte: %s" %ip\_address).encode("utf-8"), address)

Aryja pse e kemi address[0] është që të marrin vetëm IP adresën e jo edhe portin.

Te TCPServeri

def IDADDRESS():

hostname = socket.gethostname()

ip\_address = socket.gethostbyname(hostname)

print('Kalkulimi nga serveri: ' + str(ip\_address))

client\_socket.send(("Përgjigjja: IP Adresa e klientit eshte: %s" %ip\_address).encode("utf-8"))

socket.gethostname() na e kthen emrin e sistemit ku ekzekutohet programi. Ky funksion i kombinuar ne kod me socket.gethostbyname(hostname) na e kthen IP e lokal hostit.

## Metoda PORT

Kjo metod e kthen PORTIN e klientit. Me poshte eshte kodi qe na duhet per te mar kete kerkese:

def PORT():

client\_socket.send(("Përgjigjja: Klienti eshte duke perdorur portin %s" %address[1]).encode("utf-8"))

Arsyja se pse e kemi address[1] eshte sepse address-a paraqet IP dhe Portin e klientit, kshtuqe ne kete rast neva na duhet vetem porti.

## Metoda COUNT

Kjo metodë i numron zanoret dhe bashkëtingëlloret në një tekst. Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def COUNT(r1):

f1 = r1

f1 = str(metoda.split(" ")[0])

def listToString(s):

str1 = ""

for ele in s:

str1 = str1 + ele + " "

return str1

f2 = metoda.split(" ")[1::]

f2 = listToString(f2)

vowels = 0

consonants = 0

for i in f2:

if(i == 'a' or i == 'e' or i == 'i' or i == 'o' or i == 'u'

or i == 'A' or i == 'E' or i == 'I' or i == 'O' or i == 'U'):

vowels = vowels + 1

elif i == ' ':

continue

else:

consonants = consonants + 1

client\_socket.send(("Përgjigjja: Teksti i pranuar permban %s zanore dhe %s bashketingëllore" %(vowels,consonants)).encode("utf-8"))

Së pari e kam ndarë pjesën e kërkesës e cila e përcakton metodën. Pastaj me f2 e kam marrë pjesën tjetër te mesazhit, të cilën nga lista e kam kthyer në string me anë te listToString(f2). Me anë te for i in f2 i ndajmë zanoret prej bashkëtingëlloreve dhe në fund i dërgojmë te klienti.

## Metoda REVERZ

Kjo metodë e kthen një fjalë amo fjalë në reverzë (i kthen kundër). Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def REVERZ(r1):

s2 = r1

def reverse(string):

string = string[::-1]

return string

def listToString(s):

str1 = ""

for ele in s:

str1 = str1 + ele + " "

return str1

s2 = metoda.split(" ")[1::]

s2 = listToString(s2)

s2 = reverse(s2).strip()

client\_socket.send(("Përgjigjja: Fjalia e kthyer ne reverz: %s " %s2).encode("utf-8"))

Se pari e marrim pjesën e mesazhit e cila nuk e përfshin kërkesën dhe e kthejmë në string. Me anë të funksionit reverse(string) e kthejmë fjalën-fjalinë në reverz. Strip() e kam përdorur që të mos paraqiten hapsirat në fund dhe fillim të fjalisë.

## Metoda PALINDROME

Kjo metodë na tregon se a është fjala apo fjalia palindrome( a është fjalia e njëjtë edhe kur të kthejmë në të kundërt). Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def PALINDROME(r1):

string = r1

def listToString(s):

str1 = ""

for ele in s:

str1 = str1 + ele

return str1

string = metoda.split(" ")[1::]

string = listToString(string)

if(string==string[::-1]):

client\_socket.send("Përgjigjja: Teksti i dhene eshte palindrome".encode("utf-8"))

else:

client\_socket.send("Përgjigjja: Teksti i dhene nuk eshte palindrome".encode("utf-8"))

Se pari e kemi marrë pjesën e mesazhit e cila nuk e përfshin kërkesën dhe e kemi kthyer në string me anë të funksionit listToString(string). Me anë të kushtit if(string==string[::-1]): e shohim se a është fjala-fjalia e dhënë e njëjtë me atë fjalë-fjali të kthyer në reverz. Nëse plotësohet kushti fjalia është palindorme, nëse jo nuk është palindrome.

## Metoda TIME

Kjo metodë na kthen kohën e atij momenti. Për këtë metodë duhet të importojmë librarinë import datetime. Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def TIME():

metoda = datetime.datetime.now()

metoda = metoda.strftime("%d.%m.%Y %I:%M:%S %p")

serverS.sendto(("Përgjigjja: Koha eshte %s " %metoda).encode("utf-8"),address)

Me ane te datetime.datetime.now() ne e marim kohen e sakt te momentit aktual, dhe me ane te metodes strftime("%d.%m.%Y %I:%M:%S %p") i marim formatin qe ne deshirojm.

## Metoda GAME

Kjo metodë na i kthen 5 numra të rastësishëm prej numrit 1 deri në numrin 35. Për këtë metodë duhet që të importojnë librarinë import random. Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def GAME():

randomlist = random.sample(range(1, 35), 5)

metoda = randomlist.sort()

metoda = str(tuple(randomlist))

serverS.sendto(("Përgjigjja: psh. %s pra 5 numra te rastesishem nga 35" %metoda).encode("utf-8"),address)

Funksioni random.sample(range(1, 35), 5) na i kthen 5 numra të ndryshëm. Me anë të sort() i sortojmë në mënyrë rritëse listën. Me anë të tuple(randomlist) e kthejmë listen në tuple.

## Metoda GCF

Kjo metoda na e kthen pjestuesin më të madh të përbashket midis dy numrave. Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def GCF(k1,k2):

x = k1

y = k2

if x > y:

smaller = y

else:

smaller = x

for i in range(1, smaller+1):

if((x % i == 0) and (y % i == 0)):

hcf = i

serverS.sendto(("Përgjigjja: Pjestuesi me i madh i perbashket i numrave %s dhe %s eshte %s" %(x,y,hcf)).encode("utf-8"),address)

Me anë të kushtit is x > y: e gjejmë numrin më të vogël prej x, y. Pastaj i kontrollojmë numrat me anë të for i in range(1, smaller+1): nga numri 1 deri te numri i fituar smaller+1 deri sa të gjendet numri i cili e plotëson kushtin if((x % i == 0) and (y % i == 0)): nga ku fitojmë pjestuesin më të madh të përbashkët.

## Metoda CONVERT

Kjo metodë na ofron 4 mundësi të konvertimit të numrave në njësi tjera. Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def CONVERT(opcioni1,numri1):

opcioni = opcioni1

numri = numri1

if opcioni == "cmToFeet":

fjaliaP = str(float(numri) / 30.48)

serverS.sendto(("Përgjigjja: Rezultati i konvertimit: %s ft" %fjaliaP).encode("utf-8"),address)

elif opcioni == "FeetToCm":

fjaliaP = str(float(numri) \* 30.48)

serverS.sendto(("Përgjigjja: Rezultati i konvertimit: %s cm" %fjaliaP).encode("utf-8"), address)

elif opcioni == "kmToMiles":

fjaliaP = str(float(numri) / 1.609)

serverS.sendto(("Përgjigjja: Rezultati i konvertimit: %s miles" %fjaliaP).encode("utf-8"), address)

elif opcioni == "MileToKm":

fjaliaP = str(float(numri) \* 1.609)

serverS.sendto(("Përgjigjja: Rezultati i konvertimit: %s km" %fjaliaP).encode("utf-8"), address)

else:

serverS.sendto(("Shenoni nje opsion valid").encode("utf-8"),address)

Me anë të kushteve e zgjedhim metodën e konvertimit dhe pastaj kur zgjedhet metoda kryhen veprimet dhe dërgohet mesazhi te klienti.

## Metoda ANAGRAM (metoda e pare shtes)

Kjo metodë na mundëson ta dimë se a janë fjalët e dhëna anagram (a i kanë fjalët shkronjat e barabarta dhe të njëjta por në radhitje të ndryshme). Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def ANAGRAM(a1,a2):

s1=a1

s2=a2

if(sorted(s1)==sorted(s2)):

serverS.sendto(bytes("Përgjigjja: Teksti i dhene eshte anagram.","utf-8"),address)

else:

serverS.sendto(bytes("Përgjigjja: Teksti i dhene nuk eshte anagram.","utf-8"),address)

Këtu brenda kushtit sortohen në radhitje alfabetike stringjet dhe krahasohen a janë të barabarta. Pastaj varësisht nga rezultati dërgohet mesazhi i enkoduar.

## Metoda TITLE (metoda e dyte shtes)

Kjo metodë na tregon se a i përshtatet fjala apo fjalia e dhënë rregullave të titullit (që të ketë çdo fjale shkronjën e parë të madhe (uppercase) kurse shkronjat tjera të vogla (lowercase). Më poshtë është kodi që na duhet për të marrë këtë kërkesë:

def TITLE(r1):

x = int(r1.istitle())

if x == 1:

serverS.sendto(("Përgjigjja: Teksti i permbahet rregullave te titullit").encode("utf-8"),address)

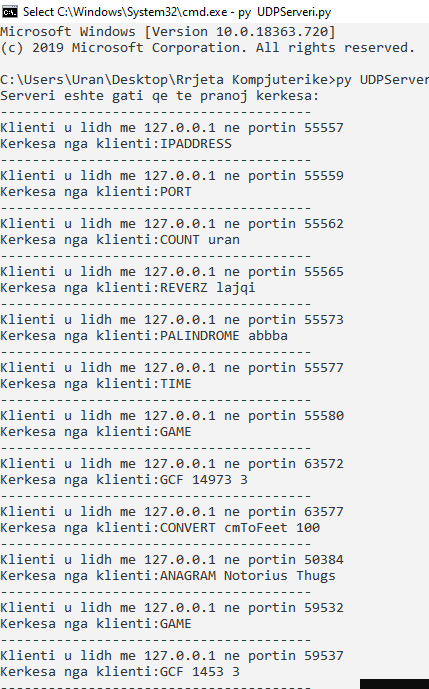
else:

serverS.sendto(("Përgjigjja: Teksti nuk i permbahet rregullave te titullit").encode("utf-8"),address)

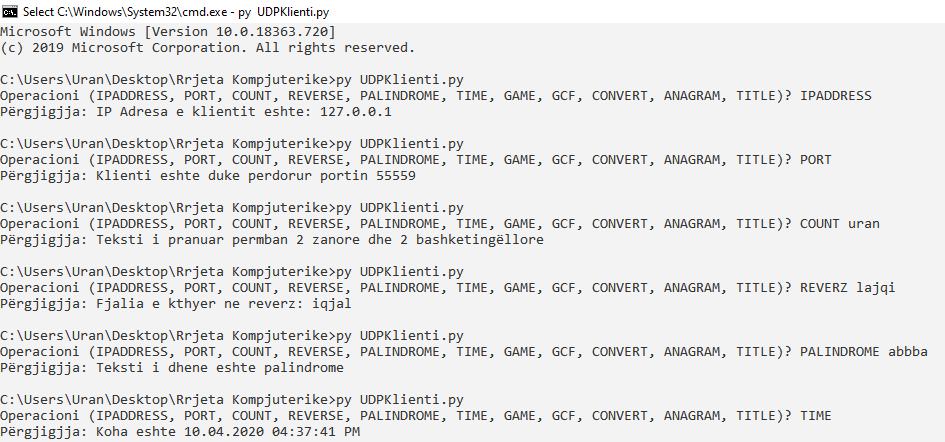
Funksioni istitle() na tregon se a i përmbahet mesazhi rregullave të titullit. Nëse mesazhi i përshtatet rregullave funksioni r1.istitle() na e kthen true e që kur e kthejmë në int() na e jep numrin 1. Kështu në bazë të rezultatit do të dërgohet mesazhi.

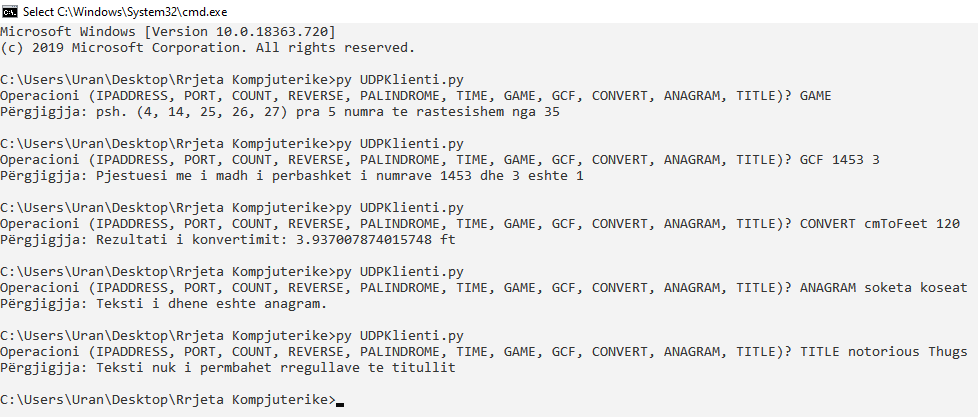
# Konkluzionet

UDPServeri.py

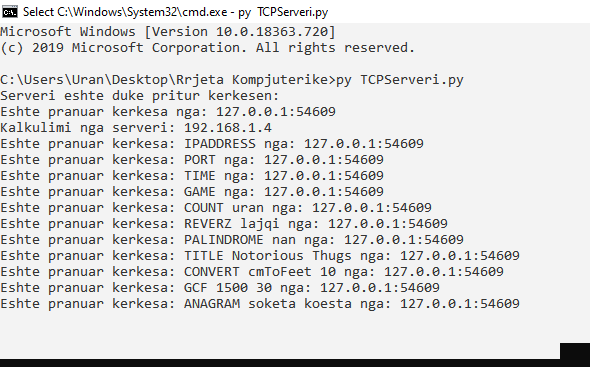


UDPKlienti.py





TCPServeri.py



TCPKlienti.py

