# UNIVERSITETI I PRISHTINËS Fakulteti i Inxhinierisë Elektrike dhe Kompjuterike



## **RAPORT**

Lënda: Rrjetat Kompjuterike

Projekti 1: Dizajnimi Klient-Server

Studenti: Arbena Musa Nr. i ID: 170714100093 Emri i projektit: Projekti 1-Dizajnimi Klient-Server

Emri i studentit: Arbena Musa

Vegla e përdorur: Visual Studio - Python

3.6.3 (v3.6.3:2c5fed8, Oct 3 2017, 18:11:49) [MSC v.1900 64 bit (AMD64)]

Sistemi operativ: Windows 10 Pro, Version: 1803 Lista e metodave të implementuara: IPADRESA,

NUMRIIPORTIT,

BASHKETINGELLORE,

PRINTIMI,

EMRIIKOMPJUTERIT,

KOHA, LOJA,

FIBONACCI, KONVERTIMI.

## Përmbajtja

Hyrje

Përshkrimi i projektit

Programimi Klient-Server

TCP protocol

UDP protocol

Përshkrimi i metodave

Metoda IPADRESA

Metoda NUMRIIPORTIT

Metoda BASHKETINGELLORE

Metoda PRINTIMI

Metoda EMRIIKLIENTIT

Metoda KOHA

Metoda BASHKETINGELLORE

Metoda FIBONACCI

Metoda KONVERTIMI

Metoda VERSIONIIPYTHON

Metoda VERSIONIIOS

Testimi

Testimi TCP protocol

Testimi UDP protocol

Përmbledhje e rezultateve të testimit

## Hyrje

## Përshkrimi i projektit

Ky projekt përshkruan dizajnimin, implementimin dhe testimin e komunikimit klient-server përmes socket programming në gjuhën programuese Python. Protokolli i përdorur do të jetë FIEK protokolli i cili u mundëson serverëve dhe klientëve komunikimin përmes dy versioneve të tij, si TCP-protocol dhe si UDP-protocol. Serveri përmban 11 kërkesa (metoda) të cilave mund t'iu referohet klienti. Për secilën kërkesë të klientit server kthen përgjigjen përkatëse mbrapsht te klienti.

## Protokolli FIEK-TCP

Protokolli TCP (Transmission Control Protocol) është një nga protokollet kryesore. Gjatë komunikimit mes dy pikave bëhet kontrollimi i transportit të paketave. Çdo humbje e paketave në rrjet detektohet dhe ridërgohet nga dërguesi te marrësi. Të dhënat pranohen nga marrësi në të njejtin rend siç janë dërguar nga dërguesi.

#### Protokolli FIEK-UDP

Protokolli UDP (User Datagram Protocol) transmeton të dhënat si datagrams, mes dy pikave të ndryshme të cilat nuk kanë nevojë të konektohen paraprakisht. Nuk kontrollohet transporti i paketave, pra në rast të humbjes marrësit nuk i ridërgohen përsëri.

## Programimi Klient-Server

Socket është një endpoint (pikë fundore) që mundëson komunikimin e dy proceseve brenda të njejtit kompjuter apo mes kompjuterëve të ndryshëm. Socket paraqet një mënyrë virtuale të komunikimit të dy kompjuterëve në rrjet. Socket hap lidhjet në rrjet që të ju mundësojë aplikacioneve të shkruajnë dhe të lexojnë nga njëri tjetri përmes rrjetës. Një Socket identifikohet në mënyrë unike nga dy komponente,IP adresa dhe numri i portit.

Klienti e ka mundësinë ta zgjedhë vetë IP adersën dhe numri e portit, apo të perdorë ato që caktohen në kod

```
print("A doni te caktoni vete serverin dhe portin qe deshironi te perdorni?")
pergjigja=input().upper()

Dif(pergjigja=="PO"):
    print("Jep emrin e serverit:")
    servername=input().lower()
    print("Jep numerin e portit")
    p=input()
    port=int(p)

else:
    servername='localhost'
    port=12000
```

## TCP protocol

```
serversocket=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
serversocket.bind((servername, serverport))
```

Dy funksionet që përdoren në komunikim janë .sendall() që merr si parameter vlerën që duhet të dërgohet tek pika tjetër e komunikimit dhe .recv() që ka si parameter gjatësinë në byte të vlerës që duhet të mirret nga pika tjetër.

## UDP protocol

```
serversocket=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
serversocket.bind((servername, serverport))
```

Dy funksionet që përdoren në komunikim janë .sendto() që merr si parametër të parë vlerën që duhet dërguar tek pika tjetër e komunikimit , ndësi si parameter të dyte e ka adresën e pikes tjetër.Dhe .recvform() që ka si parameter gjatësinë në byte të vlerës që duhet të mirret nga pika tjetër.

#### Përshkrimi i metodave

Pas vendosjes së komunikimit mes serverit dhe klientit, klientit i mundesohet thirrja e metodave:

```
while True:

print('Operacioni (IPADRESA, NUMRIIPORTIT, BASHKETINGELLORE, PRINTIMI, EMRIIKOMPJUTERIT, KOHA, LOJA, FIBONACCI, KONVERTIMI, VERSIONIIPYTHON, VERSIONIIOS)?')

var=input().upper().encode()

if var==':

break
```

```
Ky eshte programi FIEK-TCP Client.
Klienti eshte gati per te komunikuar me serverin.

Operacioni (IPADRESA, NUMRIIPORTIT, BASHKETINGELLORE, PRINTIMI, EMRIIKOMPJUTERIT, KOHA, LOJA, FIBONACCI, KONVERTIMI)?
```

Metoda qe thirret nga klienti kalon ne server.

```
Bdef clientthread(socketKlienti):
    try:
    while True:
        opsioni = socketKlienti.recv(1024).decode()
        kerkesat(opsioni)
        socketKlienti.close()
    except:
    print("Error")
```

Metoda që e zgjedh klienti merret si input, enkodohet dhe dërgohet te server. Nga ana tjetër server është i gatshëm për të pranuar kërkesa. Kërkesa që e pranon serveri dekodohet dhe kalon si parametë i metodës kerkesa().

Metoda kerkesa() merr si parameter metodën e kërkuar nga klienti.

```
def kerkesat(op):
    opi=op
    op = op.split()
     if(op[0]=="IPADRESA"):
         IPADRESA()
     elif(op[0]=="NUMRIIPORTIT"):
         NUMRIIPORTIT()
     elif(op[0] == "BASHKETINGELLORE"):
         BASHKETINGELLORE(opi[17:])
     elif(op[0]=="PRINTIMI"):
         PRINTIMI(opi[9:])
     elif(op[0]=="EMRIIKOMPJUTERIT"):
         EMRIIKOMPJUTERIT()
     elif(op[0]=="KOHA"):
         KOHA()
     elif(op[0]=="LOJA"):
         LOJA()
     elif(op[0]=="FIBONACCI"):
         FIBONACCI(op[1])
     elif(op[0]=="KONVERTIMI"):
         KONVERTIMI(op[1],op[2])
     elif(op[0]=="VERSIONIIPYTHON"):
         VERSIONIIPYTHON()
     elif(op[0]=="VERSIONIIOS"):
        VERSIONIIOS()
         NOFUN()
```

Varësisht nga zgjedhja e klientit thirret metoda përkatëse. Metoda kërkesa e merr tërë zgjedhjen e klinetit dhe e ndan përmes split(). Argumenti i parë i fituar është gjithmonë emri i metodës. Kur metoda jepet së bashku me parameter atëherë argumenti i dyte (ose i tretë) është parametri i metodës që thirret.

#### Metoda IPADRESA

```
def IPADRESA():
    z="IP Adresa e klientit eshte: "+ addr[0]
    socketKlienti.sendall(str.encode(str(z)))
```

Kjo metodë merr anëtarin e parë të addr, në të cilin është ruajtur IP adresa e klientit, e enkodon dhe e dergon tek klienti.

#### Metoda NUMRIIPORTIT

```
Didef NUMRIIPORTIT():
    z="Klienti eshte duke perdorur portin "+ str(addr[1])
    socketKlienti.sendall(str.encode(str(z)))
```

Kjo metodë merr anëtarin e dyte të addr, në të cilin është ruajtur numri i portit në të cilin është konektuar klienti,

e enkodon dhe e dergon te klienti.

## Metoda BASHKETINGELLORE

Serveri merr nga klienti tekstin si parameter nga i cili duhet llogaritur numri i bashkëtingëlloreve. Teksti i marrë dekodohet, kthehet në shkronja të vogla dhe secili karakter kontrollohet se a është shkronjë bashkëtingëllore. Numri i shkronjave bashkëtingëllore dërgohet te klienti si string i enkoduar.

#### Metoda PRINTIMI

```
def PRINTIMI(a):
                                               Ky funksion merr nga klienti tekstin si parameter i cili
      socketKlienti.sendall(str.encode(a))
                                               duhet të printohet. Tekstin e marrë e enkodon përsëri dhe ia
dërgon klientit.
```

## Metoda EMRIIKLIENTIT

```
EMRIIKOMPJUTERIT():
z=socket.getfqdn(servername)
socketKlienti.sendall(str.encode(str(z)))
```

Me anë të kësaj metode dërgohet tek klienti emrin e makinës që përdorë klienti. Ky emër mirret nga .getfqdn(hostname) që jep emrin e pajisjes hostuese.

#### Metoda KOHA

Importohet libraria datetime.

```
z=(y.strftime("%d")+'.'+y.strftime("%m")+'.'+y.strftime("%y")+' '+y.strftime("%I")+':'+y.strftime("%N")+':'+y.strftime("%S")+' '+y.strftime("%p"))
socketKlienti.sendall(str.encode(z))
```

Kjo metodë merr të dhënat aktuale të kohes. Pas përpunimit në string, dërgohen tek klienti.

#### Metoda LOJA

import random

```
Importohet libraria ramdom.
from random import choice
```

```
LOJA():
sequence = [i for i in range(50)]
for _ in range(7):
    selection = choice(sequence)
    z=z+str(selection)+' '
z=z+')jane 7 numra te rastesishem nga 49.
socketKlienti.sendall(str.encode(z))
```

Kjo metodë ruan në varg numrat nga 1 deri në 50 pa përfshier 50. Zgjedhen random 7 numra prej vargut dhe i dergohen klientit.

#### Metoda FIBONACCI

```
_

def FIBONACCI(n):
     a,b = 0,1
     for i in range(n):
         a,b = b, a+b
     socketKlienti.sendall(str.encode(str(a)))
```

Kjo metodë merr si parameter indeksin e numrit të fibonaccit dhe e logaritë numrin përkatës dhe ia dërgon klientit.

#### Metoda KONVERTIMI

```
Ddef KONVERTINI(a,b):
    if(a=="KILOWATTOHORSEPOWER"):
        z=int(b)*1.34102209
    elif(a=="HORSEPOWERTOKILOWATT"):
        z=int(b)/1.34102209
    elif(a=="DEGREESTORADIANS"):
        z=int(b)*0.0174532925
    elif(a=="RADIANSTODEGREES"):
        z=int(b)/0.0174532925
    elif(a=="GALLONSTOLITERS"):
        z=int(b)*3.78541178
    elif(a=="LITERSTOGALLONS"):
        z=int(b)/3.78541178
    else:
        z="Nuk eshte shkruar mire kerkesa per konvertim."
    socketKlienti.sendall(str.encode(str(z)))
```

Kjo metodë merr dy parametra, i pari është lloji i konvertimit që duhet të kryhet, ndërsa parametri i dyte është vlera që duhet të konvertohet. Kthen tek klienti vlerën e caktuar.

## Metoda VERSIONIIPYTHON

```
def VERSIONIIPYTHON():
    z=sys.version
    socketKlienti.sendall(str.encode(z))
```

Kjo metodë kthen te klienti versionin e Pythonit që përdoret nga kompjuteri i cili përdoret.

## Metoda VERSIONIIOS

```
□def VERSIONIIOS():
    z=platform.machine()+" "+platform.platform()+" "+platform.node()+" "+platform.processor()
    socketKlienti.sendall(str.encode(z))
```

Kjo metodë jep të dhënat e kompjuterit, os, emrin e nyjës(emrin e kompjuterit) dhe karakteristikat e procesorit.

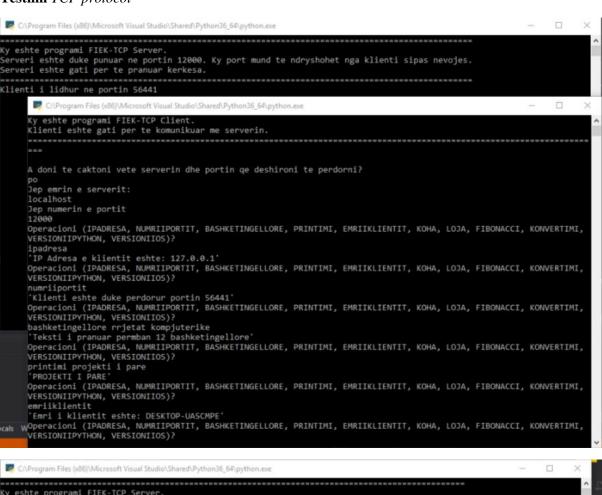
Në rastet kur nga klienti thirret një metodë jo ekzistuese, injorohet kërkesa.

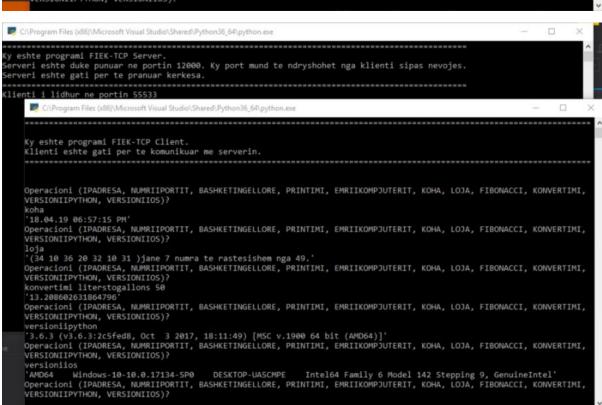
```
⊟def NOFUN():

z=

socketKlienti.sendall(str.encode(str(z)))
```

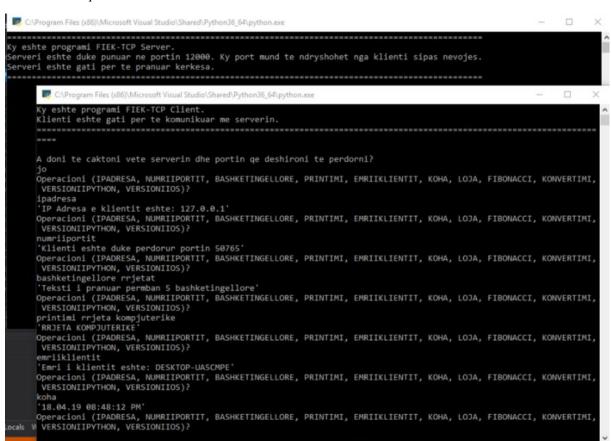
## **Testimi** TCP protocol

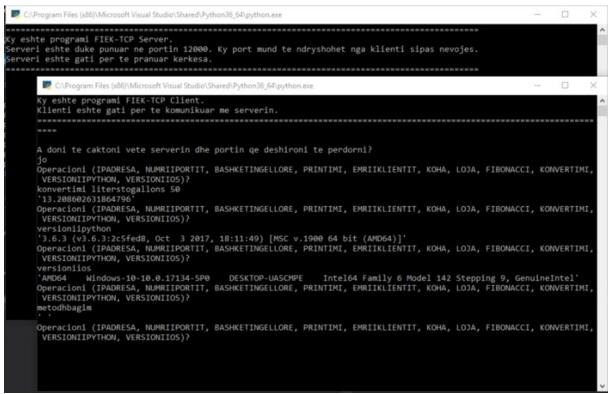


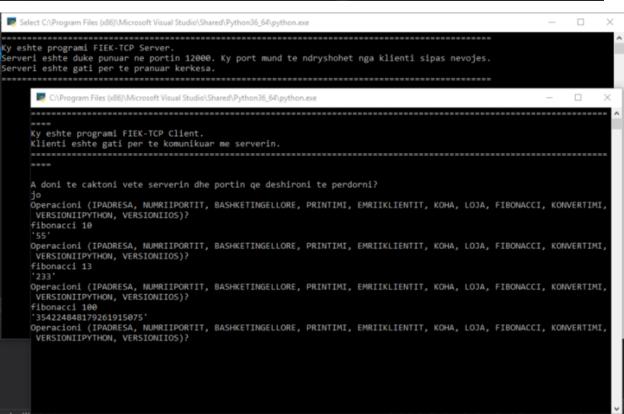


```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36_64\python.exe
                                                                                                                             (y eshte programi FIEK-TCP Server.
Serveri eshte duke punuar ne portin 12000. Ky port mund te ndryshohet nga klienti sipas nevojes.
Serveri eshte gati per te pranuar kerkesa.
                Klienti i lidhur ne portin 56672
       C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python36_64\python.exe
      Ky eshte programi FIEK-TCP Client.
Klienti eshte gati per te komunikuar me serverin.
      A doni te caktoni vete serverin dhe portin qe deshironi te perdorni?
      Operacioni (IPADRESA, NUMRIIPORTIT, BASHKETINGELLORE, PRINTIMI, EMRIIKLIENTIT, KOHA, LOJA, FIBONACCI, KONVERTIMI, VERSIONIIPYTHON, VERSIONIIOS)?
      fibonacci 10
      Operacioni (IPADRESA, NUMRIIPORTIT, BASHKETINGELLORE, PRINTIMI, EMRIIKLIENTIT, KOHA, LOJA, FIBONACCI, KONVERTIMI, VERSIONIIPYTHON, VERSIONIIOS)?
       '55
      fibonacci 6
      Operacioni (IPADRESA, NUMRIIPORTIT, BASHKETINGELLORE, PRINTIMI, EMRIIKLIENTIT, KOHA, LOJA, FIBONACCI, KONVERTIMI,
      VERSIONIIPYTHON, VERSIONIIOS)?
      fibonacci 15
      '610'
      Operacioni (IPADRESA, NUMRIIPORTIT, BASHKETINGELLORE, PRINTIMI, EMRIIKLIENTIT, KOHA, LOJA, FIBONACCI, KONVERTIMI, VERSIONIIPYTHON, VERSIONIIOS)?
```

#### Testimi UDP protocol







## Përmbledhje e rezultateve të testimit

Kërkesat e projektit janë përmbushur duke filluar nga implementimi i metodave. Siç shihet testimi kryhet në mënyrën e paraparë, dmth. të gjitha metodat ekzekutohen ashtu siç duhet. Poashtu edhe dy metodat shtesë ekzekutohen pa problem. Në rastet kur klienti dërgon ndonjë kërkesë joekzistenti, ajo thjesht injorohet dhe vazhdon ekzekutimi normalisht.

Në program është përfshirë edhe error handling përmes blloqeve try dhe except, i cili mundëson që çdo gabim eventual gjatë ekzekutimit të mos e prish stabilitetin e programit.

Tek server exception handling përdoret në dy raste:

```
try:
    serversocket.listen()
    socketKlienti, addr = serversocket.accept()
    print("Klienti i lidhur ne portin "+str(addr[1]))
    start_new_thread(clientthread, (socketKlienti, ))
except:
    print("Ka ndodhur nje gabim ne krijim e socketKlienti!")
```

```
try:
    while True:
        opsioni = socketKlienti.recv(128).decode()
        kerkesat(opsioni)
        socketKlienti.close()
except:
    print("Ka ndodhur nje gabim gjate marrjes se kerkeses nga klienti!")
```

## Multi-Threads

