

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: Blerta Krasniqi

Asistenti : Msc. Edon Mustafa

Data : 18/04/2017

Permbajtja

**1.Veglat e perdorura, versionet e tyre dhe lista e metodave te implementuara3**

**2.Hyrje4**

2.1 Pershkrim I shkurter rreth projektit4

2.2 Si punon arkitektura klient/server?4

2.3 Pershkrimi I klientit dhe serverit4

**3.Realizimi I TCP Klientit dhe serverit permes kodit** ………………………………………………………………………………**5**

3.1 Realizimi I TCPserver…………………..…………………………………………………………………………………………….5

3.2 Realizimi I TCPclient………………………………………………………………………………………………………………….6

**4.Realizimi I UDP Klientit dhe serverit permes kodit** …………………………………………………………………………….7

4.1 Realizimi I UDPserver…………………..…………………………………………………………………………………………..7

4.2 Realizimi I UDPclient………………………………………………………………………………………………………………..8.

**5.Pershkrimi I metodave dhe rezultatet pas tetstimit**.……………………………………………………………………….11

5.1 Pershkrimi i metodave pa hyrje11

5.1.1 Metoda IP…………………………………………………………………………………………………………………………….11

5.1.2 Metoda PORT………………………………………………………………………………………………………………………12

5.1.3 Metoda HOST………………………………………………………………………………………………………………………13

5.1.4 Metoda TIME……………………………………………………………………………………………………………………….15

5.1.5 Metoda KENO………………………………………………………………………………………………………………………16

5.2 Pershkrimi I metodave me hyrje……………………………………………………………………………………………………17

5.2.1 Metoda ZANORE…………………………………………………………………………………………………………………17

5.2.2 Metoda PRINTO………………………………………………………………………………………………………………….18

5.2.3 Metoda FAKTORIEL……………………………………………………………………………………………………………..19

5.2.4 Metoda KONVERTO…………………………………………………………………………………………………………….20

**6.Pershkrimi I metodave personale dhe rezultatet pas tetstimit**.…………………………………………………….23

6.1 Realizimi I metodes permes GUI23

6.1.1 Metoda MACA………..………………………………………………………………………………………………………….23

6.2 Realizimi I metodave te tjera23

6.1.1 Metoda POEM.…………………………………………………………………………………………………………….27

6.1.2 Metoda GUESS..…………………………………………………………………………………………………………….28

Veglat e perdorura, versionet e tyre dhe lista e metodave te implementuara

Sistemi operativ:

* Windows 10

Veglat e perdorura dhe versionet e tyre:

* Microsoft Visual Studio 2015 Ultimate
* Python 2.7

Libarite e Python:

* Socket
* Random
* Dat**e**Time
* Cprofile
* Decimal
* Tkinter

Lista e metodave te implementuara:

* IP
* PORT
* ZANORE
* PRINTO
* HOST
* TIME
* KENO
* FAKTORIEL
* KONVERTO

Lista e metodave personale te implementuara:

* MACA
* POEM
* GUESS

## 2.Hyrje

# 2.1 Pershkrim I shkurter rreth projektit

Projekti perbehet nga sistemi programimi I klienti dhe serverit te cilet komunikojne permes FIEK Protokollit.

Komunikimi ndermjet klientit dhe serverit behen permes socket-ave te cilat mundesojne komunikim e tyre ne portin default 9000 ose nje port te vecante te cilin mund ta caktoje klienti.

FIEK Protokolli eshte nje protokoll I thjeshte I komunikimit I cili per baze ka fjalen kyqe ne fillim te kerkesen e cila percakton metoden te cilen do ta kthej server. Ajo ka dy versione:

1. FIEK-TCP
2. FIEK-UDP

Ky protokoll ju pergjigjet vetem ketyre kerkesave: IP, PORT, ZANORE, PRINTO, HOST, TIME, KENO, FAKTORIEL, KONVERTO te cilat dergohen nga klienti ne server dhe per te cilat serveri kthen nga nje pergjigje te vecante, klienti se bashku me kerkesen mund te dergoje edhe input varesisht nga kerkesa.

# 2.2 Si punon arkitektura klient/server?

**Serveri** njëjtë sikurse edhe kompjuterët e tjerë është një makinë procesorike e fuqishme. Detyrat e kësaj makinë janë që të shërbejnë klientët. Serveri është console aplikacion që lidhet me klient përmes socketave dhe për secilen lidhje me klient krijon një thread të ri. Serveri dallon llojet e kërkesave që vijnë nga klienti dhe për secilin lloj ka përgjigje të veqantë. Pas seciles përgjigje mbyllet soketi.

# 2.3 Pershkrimi I klientit dhe serverit

FIEK-TCP:

Se pari vendoset nje lidhje(segment) ne mes te klientit dhe serverit permes socket-ave ne portin e caktuar(default 9000). Pastaj klienti permes command line e dergon kerkesen tek server, I cili nese kerkesa eshte valide I pergjigjet kerkeses specifike te klientit dhe me pas mbyll lidhjen me klientin. Serveri eshte ne gjendje te lidhet me me shume kliente ne te njejten kohe dhe ti kthej pergjigje klienteve ne menyre paralele.

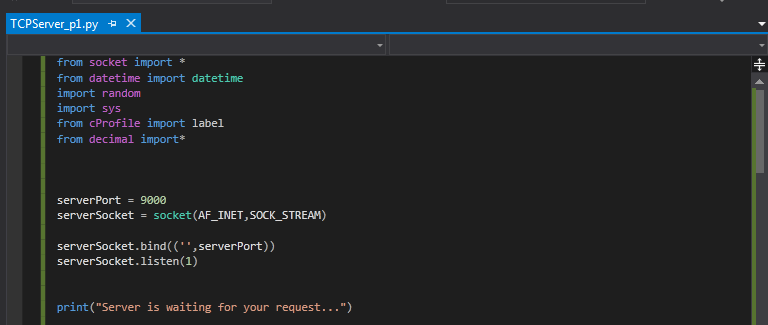
FIEK-UDP:

Ne versionin UDP te prokollit FIEK komunikimi realizohet vetem permes UDP datagram-ve, ku me c’rast nuk krijohet nje lidhje(segment) mes klientit dhe serverit por komunikimi fillon drejt. Klienti dergon kerkesen tek server permes nje UDP datagram. Pasi qe kerkesa te validohet ne server ajo poashtu kthehet tek klienti permes UDP datagram-it. Protokolli FIEK limiton qe klienti te dergon vetem nje kerkese per datagram.

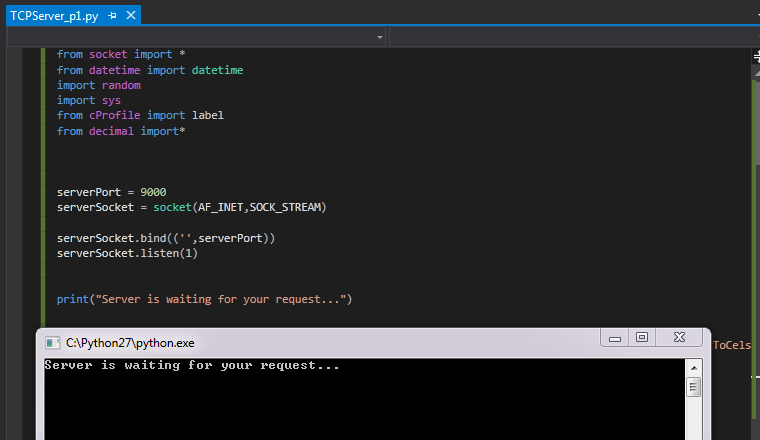
## 3. Realizimi I TCP klientit dhe serverit permes kodit

# 3.1 Realizimi I TCPserver

Se pari deklarohet variabla statike serverPort e cila perdoret gjate ekzekutimit te programit , dhe sherbejne per tu lidhur me klientin.

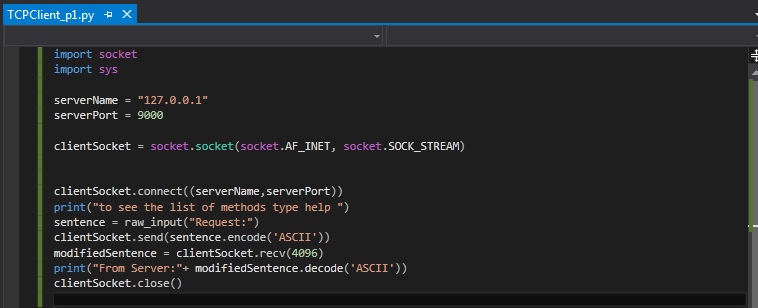


Pas ekzekutimit serveri eshte I gatshem per pranimin e kerkesave nga klienti

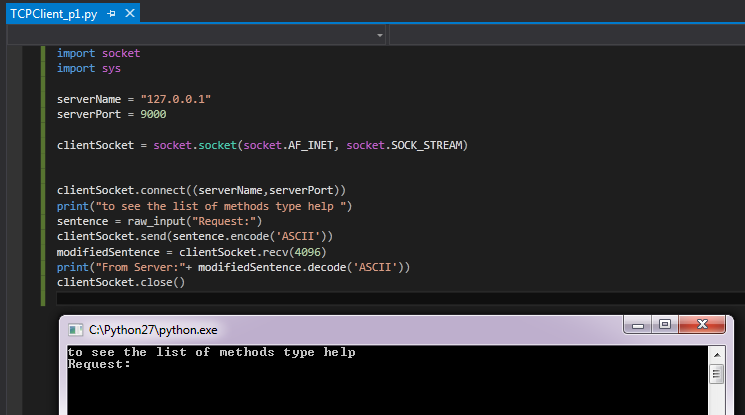


# 4.1 Realizimi I TCPclient

Ne fillim se pari deklarohen variabla serverName dhe serverPort me te cilat do te mundesojme lidhjen e klientit ne server dhe permes te ciles kryhet komunikimi.



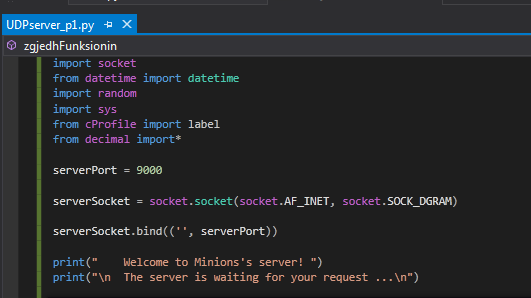
Pas ekzekutimit te TCPclient , klienti mund te beje kerkesen dhe pergjigjien e merr nga TCPserver



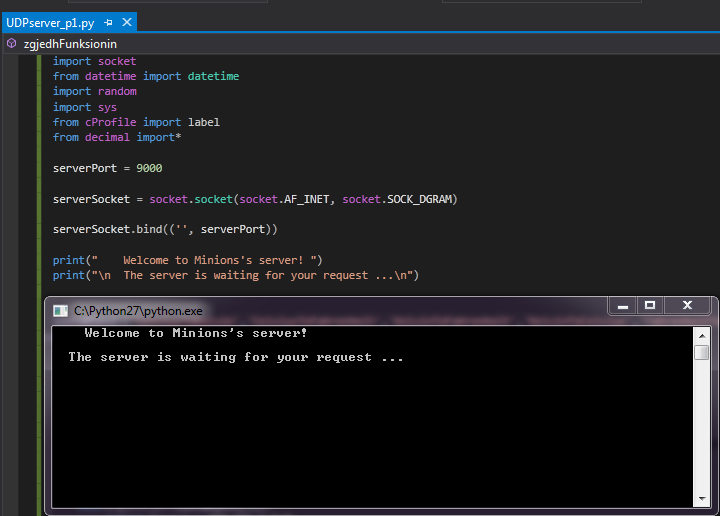
## 4.Realizimi I UDP klientit dhe serverit permes kodit

# 4.1 Realizimi I UDPserver

Se pari deklarohen variablat statike te cilat perdoren gjate ekzekutimit te programit , dhe sherbejne per tu lidhur me klientin.

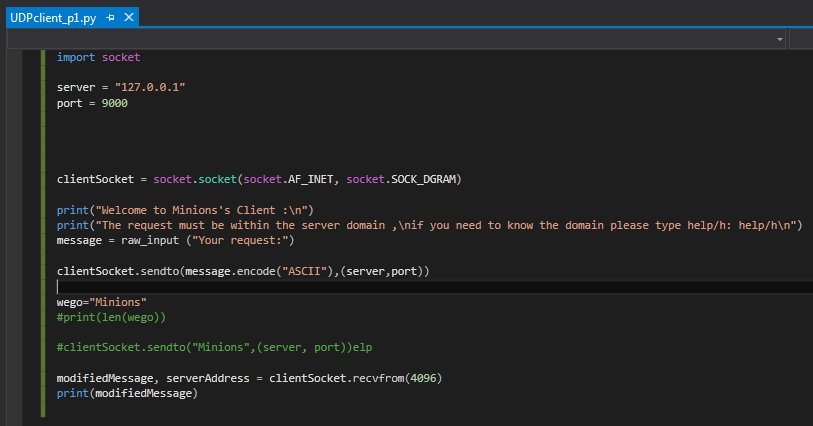


Pas ekzekutimit serveri eshte I gatshem per pranimin e kerkesave nga klienti

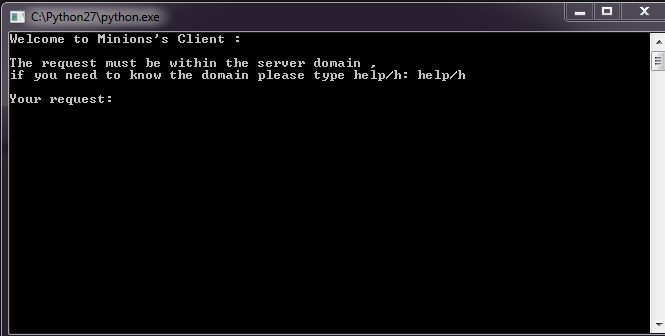


# 4.1 Realizimi I UDPclient

Ne fillim se pari deklarohen variabla server dhe port me te cilat do te mundesojme lidhjen e klientit ne server dhe permes te ciles kryhet komunikimi.



Pas ekzekutimit te UDPclient , klienti mund te beje kerkesen dhe pergjigjien e merr nga UDPserver



Per implementimin e metodave se pari deklarohet nje unaze while True: ne UDPserver, mbrenda se ciles realizohet kodi per metodat.

Fillimisht per ti mundesuar klientit njohjen e domenit eshte realizuar kerkesa HELP mbrenda unazes while True: e cila mundeson njohjen e kerkesave per te cilat UDPserver mund te kthej pergjigjie.

Realizimi permes kodit:

else: mesazhi="\n++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n"+\

"Lista e metodave eshte brenda domenes:\n"+\

"IP\n"+"PORT\n"+"ZANORE\n"+"PRINTO\n"+"HOST\n"+"TIME\n"+\

"KENO\n"+"Konverto\n"+"\tCelsiusToKelvin\n"+"\tCelsiusToFahrenheit\n"+\

"\tKelvinToFahrenheit\n"+"\tKelvinToCelsius\n"+"\tFahrenheitToCelsius\n"+\

"\tFahrenheitToKelvin\n"+"\tPoundToKilogram\n"+"\tKilogramToPound\n"+\

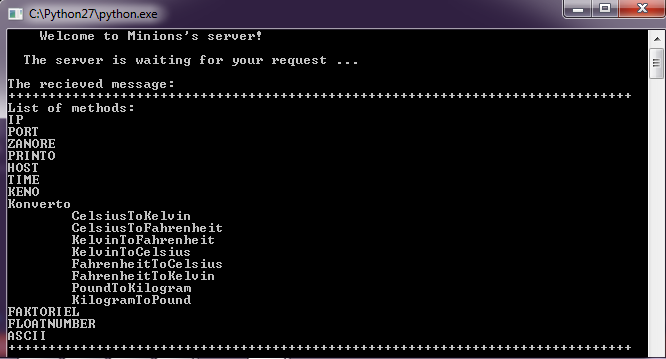
"FAKTORIEL\n"+"FLOATNUMBER\n"+"ASCII\n"+\

"++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n"

print("Mesazhi i pranuar: " + mesazhi)

serverSocket.sendto(mesazhi, clientAddress)

Pas pranimit te kerkeses HELP ne server :



Pergjigjia qe kthehet nga serveri, pasi klienti te beje kerkesen Help :



## 5.Pershkrimi i metodave dhe rezultatet pas testimit

# 5.1 Pershkrimi i metodave pa hyrje

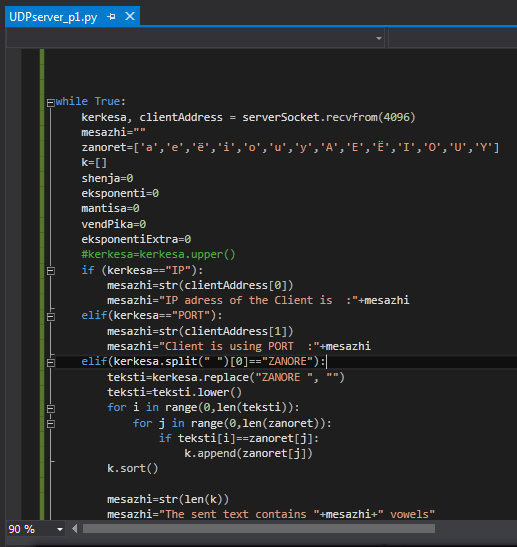
* + 1. Metoda IP

Request : IP

Pergjigjia nga serveri : Përcakton dhe kthen IP adresën e klientit në formë dhjetore me pike në një tekst mesazh p.sh. (10.10.7.251)

* Një mesazh jo me i gjate se 128 karaktere qe përmban IP adresën e klientit.
* Mesazhi nuk guxon te përmbaj NULL karakterin

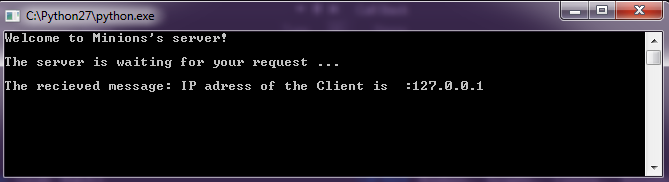
Realizimi permes kodit:

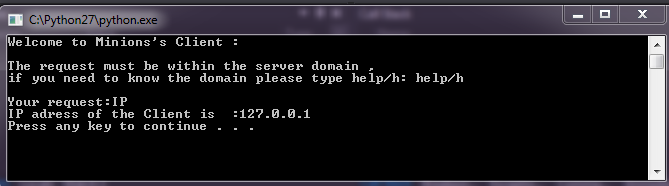


Shpjegim I kodit:Permes metodes clientAddress serveri merr ip e klientit dhe e kthen prap te ai.

Rezultatet pas testimit

Pasi klienti te beje kerkesen IP I kthehet pergjigjia nga serveri





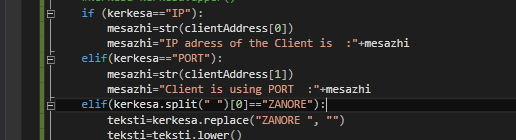
* + 1. Metoda PORT

Request : PORT

Pergjigjia nga serveri : Përcakton dhe kthen portin e klientit. (Ky duhet te jete porti I klientit dhe jo porti I serverit.)

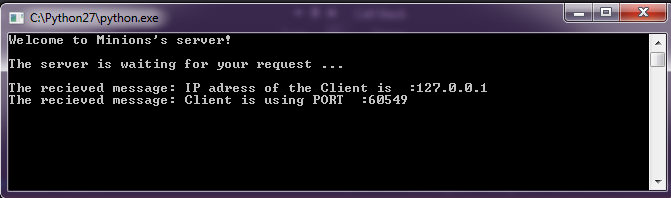
* Një mesazh jo me i gjate se 128 karaktere qe përmban IP adresën e klientit.
* Mesazhi nuk guxon te përmbaj NULL karakterin

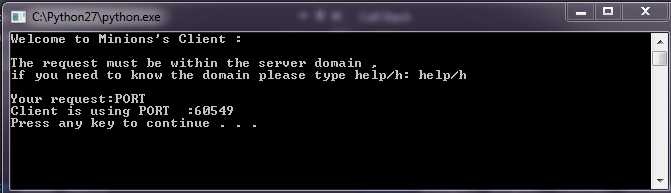
Realizimi permes kodit:



Rezultatet pas testimit

Pasi klienti te beje kerkesen PORT I kthehet pergjigjia nga serveri





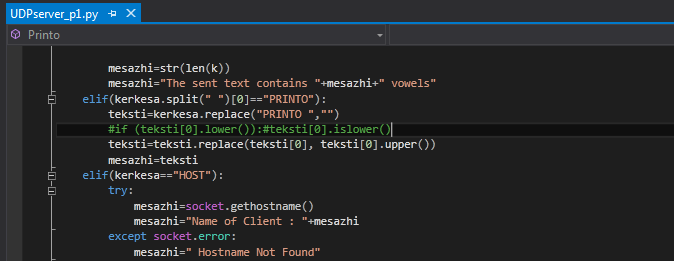
* + 1. Metoda HOST

Request : HOST

Pergjigjia nga serveri : Kërkon emrin e hostit dhe e kthen atë. Nëse emri i hostit nuk mund te përcaktohet (gjendet) atëherë duhet te kthehet një mesazh dhe te tregohet se emri i hostit nuk mund te gjendet

* Një mesazh jo me i gjate se 128 karaktere qe përmban IP adresën e klientit.
* Mesazhi nuk guxon te përmbaj NULL karakterin

Realizimi permes kodit:



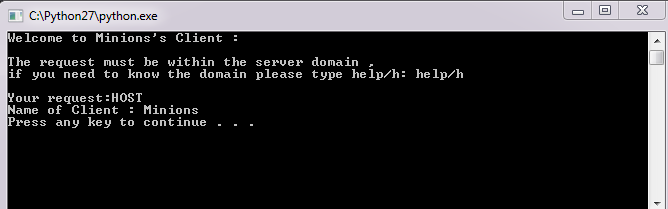
Shpjegimi I kodit:

Me ane te metodes .GetHostName() mirret emri i hostit i cili pastaj kthehet tek klienti, ne qofte se marrje e emrit te hostit ka qene e pasukseshme klienti lajmerohet qe emri i hostit nuk eshte marre.

Rezultatet pas testimit

Pasi klienti te beje kerkesen HOST I kthehet pergjigjia nga serveri





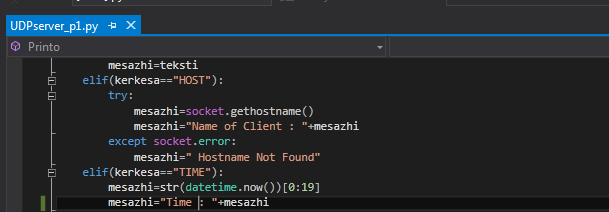
* + 1. Metoda TIME

Request : TIME

Pergjigjia nga serveri : Përcakton kohen aktuale ne server dhe e dërgon atë tek klienti si format te

lexueshme për njerëzit.

* Një mesazh jo me i gjate se 128 karaktere qe përmban kohen e serverit.
* Mesazhi nuk guxon te përmbaj NULL karakterin.

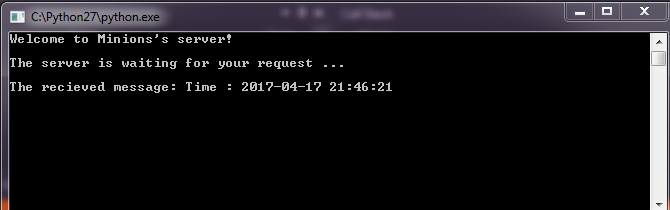
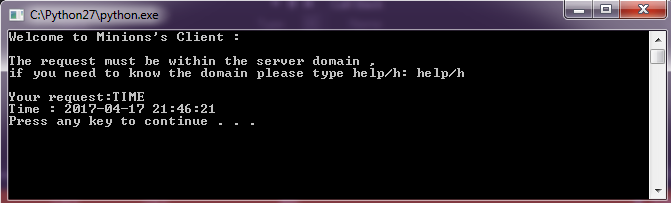
Realizimi permes kodit: 

Shpjegimi I kodit:

Permes klases DateTime mirret koha aktuale e sistemit dhe kthehet ne string ne formatin “HH:mm:ss tt

Rezultatet pas testimit

Pasi klienti te beje kerkesen TIME I kthehet pergjigjia nga serveri

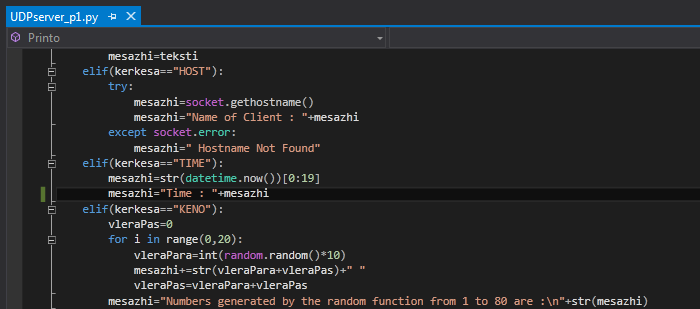
 

* + 1. Metoda KENO

Request : KENO

Pergjigjia nga serveri : Kthen 20 numra nga rangu [1,80].

* Një mesazh jo me i gjate se 128 karaktere qe përmban kohen e serverit.
* Mesazhi nuk guxon te përmbaj NULL karakterin.

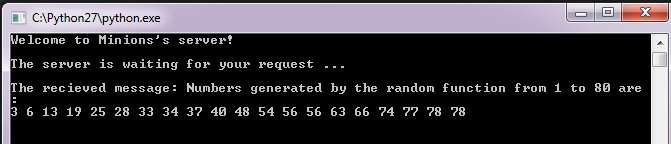
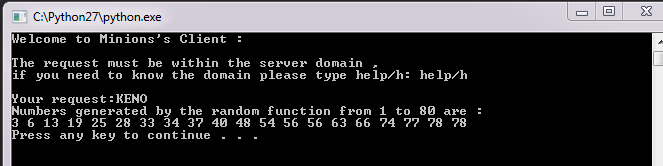
Realizimi permes kodit: 

Shpjegimi I kodit:

Klasa Random perdoret per gjenerimin e numrave te rastesishem te cilat ruhen ne nje array te tipit int dhe permes unazes for gjenerehen 20 numra random nga 1 deri 80 permes 20 cikleve . ku ne secilin cikil shikohet se a eshte numer I perseritur ne vargun e numrave te meparshem qe te mos perseritet.

Rezultatet pas testimit

Pasi klienti te beje kerkesen KENO I kthehet pergjigjia nga serveri

# 5.2 Pershkrimi i metodave me hyrje

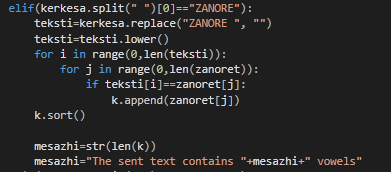
5.2.1 Metoda ZANORE

Request : ZANORE{HAPESIRE}tekst

Pergjigjia nga serveri : Gjen numrin e zanoreve ne tekst dhe kthen pergjigjien

* Një mesazh jo me i gjate se 128 karaktere qe përmban kohen e serverit.
* Mesazhi nuk guxon te përmbaj NULL karakterin.

Realizimi permes kodit:

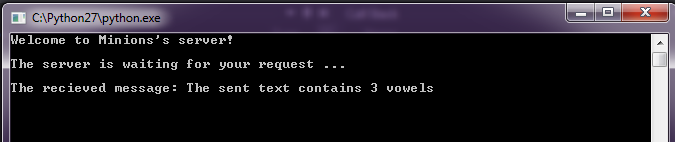
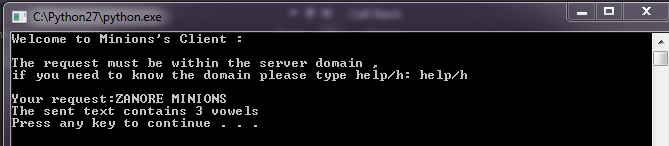
C:\Users\SINKOPA\Desktop\18009582_1757915510901747_2057160418_n.png 

Shpjegimi I kodit:

Fillimisht deklarohet vargu I zanoreve “zanoret”,pasi klienti te jep fjalen serveri e krahason nese ne fjalen e dhene nga klienti ndodhet ndonje karakter I vargut “zanoret” dhe rrit variablen per nje.Ne fund kthen numrin e zanoreve qe ndodhen ne fjalen e dhene nga klienti.

Rezultatet pas testimit

Pasi klienti te beje kerkesen ZANORE FJALE I kthehet pergjigjia nga serveri

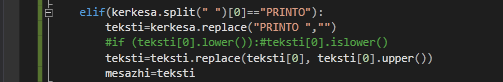
 

5.2.2 Metoda PRINTO

Request : PRINTO

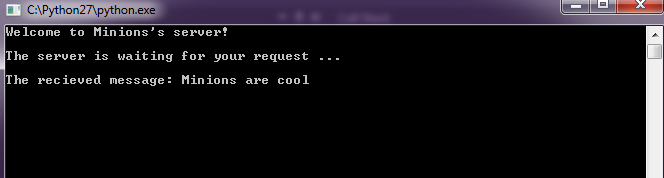
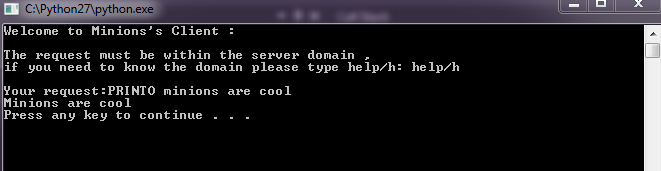
Pergjigjia nga serveri : Kthen fjalinë e shtypur ne tekst.Hapësirat ne fillim dhe ne fund te fjalisë nuk duhet te kthehen

Realizimi permes kodit:



Rezultatet pas testimit

Pasi klienti te beje kerkesen PRINTO FJALINE I kthehet pergjigjia nga serveri

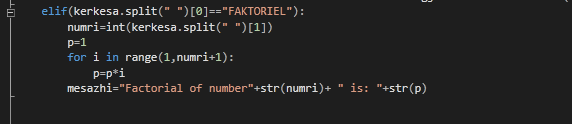
5.2.3 Metoda FAKTORIEL

Request : FAKTORIEL{HAPESIRE}numer

Pergjigjia nga serveri : Gjen faktorielin si rezultat i parametrit të dhënë hyrës. Numër i plotë psh.

FAKTORIEL 5 kthen si rezultat 120

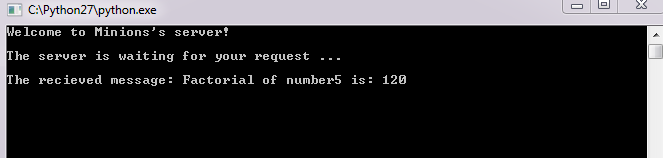
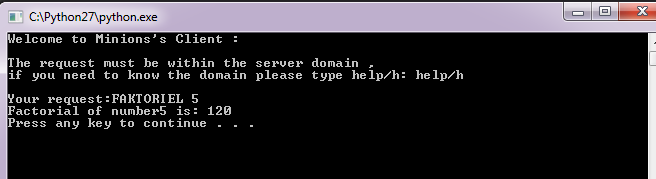
Realizimi permes kodit:



Shpjegimi I kodit:Sipas nje unaze for e cilva fillon nga 1 deri te nr qe e jep klienti dhe llogaritet si p\*I

Rezultatet pas testimit

Pasi klienti te beje kerkesen FAKTORIEL 5 I kthehet pergjigjia nga serveri

5.2.4 Metoda KONVERTO

Request : KONVERTO{HAPESIRE}opcioni{HAPESIRE}numer

Pergjigjia nga serveri : Kthen si rezultat konvertimin e opcioneve varësisht opcionit të zgjedhur.

* Lista e parametrave opcioni janë:
* CelsiusToKelvin
* CelsiusToFahrenheit
* KelvinToFahrenheit
* KelvinToCelsius
* FahrenheitToCelsius
* FahrenheitToKelvin
* PoundToKilogram
* KilogramToPound

Realizimi permes kodit:

listaNjesi=['CelsiusToKelvin','CelsiusToFahrenheit','KelvinToFahrenheit','KelvinToCelsius','FahrenheitToCelsius','FahrenheitToKelvin','PoundToKilogram','KilogramToPound']

def zgjedhFunksionin(zgjedhja, numri):

if(zgjedhja==listaNjesi[0]):

return float(numri)-273.15

elif(zgjedhja==listaNjesi[1]):

return numri \* 9/5 + 32

elif(zgjedhja==listaNjesi[2]):

return numri \* 9/5 - 459.67

elif(zgjedhja==listaNjesi[3]):

return numri - 273.15

elif(zgjedhja==listaNjesi[4]):

return (numri- 32) \* 5/9

elif(zgjedhja==listaNjesi[5]):

return (numri + 459.67) \* 5/9

elif(zgjedhja==listaNjesi[6]):

return numri \* 0.45359237

elif(zgjedhja==listaNjesi[7]):

return numri / 0.45359237

else:

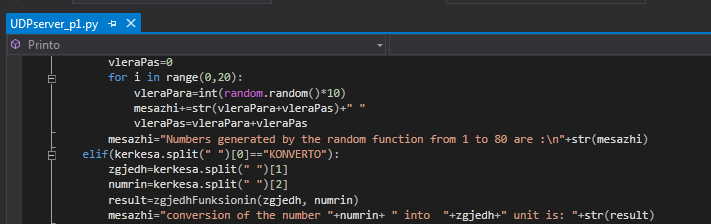
Printo()

def Printo():

print("Please choose a measure unit from the list and type a number you want to convert :")

for i in range(0,8):

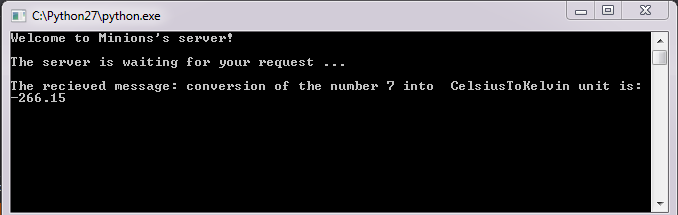
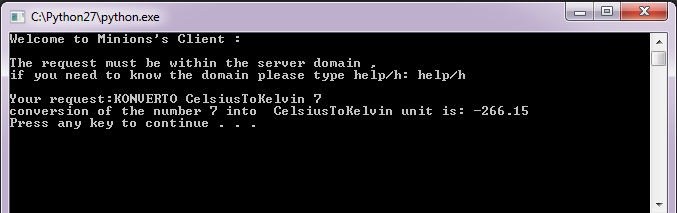
print(listaNjesi[i])



Shpjegimi I kodit:Varsishet se cilen nga mundesit e dhena per konvertim e zgjedh klienti I kthehet pergjigjia nga serveri.

Rezultatet pas testimit

Pasi klienti te beje kerkesen I kthehet pergjigjia nga serveri

## 6.Pershkrimi i metodave personale dhe rezultatet pas testimit

# Pershkrimi i metodes e realizuar ne GUI

* + 1. Metoda MACA

Kerkesa :MACA

Pergjigjia:

Realizimi permes kodit:

import Tkinter

okToPressReturn = True

hunger = 100

day = 0

#Eshte thirur paketi Tkinter e cila mundeson perdorimin e GUI

def startGame(event):

global okToPressReturn

if okToPressReturn == False:

pass

else:

startLabel.config(text="")

updateHunger()

updateDay()

updateDisplay()

okToPressReturn = False

def updateDisplay():

global hunger

global day

if hunger <= 50:

catPic.config(image = hungryphoto)

else:

catPic.config(image = normalphoto)

hungerLabel.config(text="Hunger: " + str(hunger))

dayLabel.config(text="day: " + str(day))

catPic.after(100, updateDisplay)

def updateHunger():

global hunger

hunger -= 1

if isAlive():

hungerLabel.after(500, updateHunger)

def updateDay():

global day

day += 1

if isAlive():

dayLabel.after(5000, updateDay)

def feed():

global hunger

if isAlive():

if hunger <= 95:

hunger += 20

else:

hunger -=20

def isAlive():

global hunger

if hunger <= 0:

startLabel.config(text="GAME OVER! YOU KILLED HIM/HER/IT!")

return False

else:

return True

root = Tkinter.Tk()

root.title("Stay Alive!")

root.geometry("500x300")

startLabel = Tkinter.Label(root, text="Press 'Return' to start!", font=('Helvetica', 12))

startLabel.pack()

hungerLabel = Tkinter.Label(root, text="Hunger: " + str(hunger), font=('Helvetica', 12))

hungerLabel.pack()

dayLabel = Tkinter.Label(root, text="Day: " + str(day), font=('Helvetica', 12))

dayLabel.pack()

hungryphoto = Tkinter.PhotoImage(file="hungry.gif")

normalphoto = Tkinter.PhotoImage(file="normal.gif")

catPic = Tkinter.Label(root, image=normalphoto)

catPic.pack()

btnFeed = Tkinter.Button(root, text="Feed Me", command=feed)

btnFeed.pack()

root.bind('<Return>', startGame)

root.mainloop()

Shpejgimi I kodit:

import Tkinter //Importojme librarine tkinter e cila mundeson perdorimin e nderfaqes se perdoruesit

root = Tkinter.Tk() // Definojme objektin root te klases Tkinter e cila eshte pjese e librarise tkinter

root.title("Stay Alive!") // Definojme titullin e dritares se GUI-t

root.geometry("500x300") // Definojme madhesine e dritares qe na shfaqet

startLabel = Tkinter.Label(root, text="Press 'Return' to start!", font=('Helvetica', 12)) //Definojme objektin startLabel te klases Label. Atributi text definon tekstin qe do shfaqet ne labele dhe font- fonti i cili do perdoret.

startLabel.pack() // Vendosim labelen ne dritaren qe na shfaqet

hungryphoto = Tkinter.PhotoImage(file="hungry.gif") // Definojme objektin hungryphoto te klases PhotoImage, e cila na mundeson vendosjen e imazhit me emer "hungry" ne dritaren qe shfaqet

catPic = Tkinter.Label(root, image=normalphoto) // Definojme labelen dhe ne atributin image te tij vendosim objektin e imazhit qe deshirojme te shfaqet

btnFeed = Tkinter.Button(root, text="Feed Me", command=feed) // Definojme objektin btnFeed te klases Button. Permes atibutit text definojme tekstin qe deshirojme te na shfaqet ne buton dhe ne atributin command e vendosim funksionin qe deshirojme ta thirrim pasi klikojme butonin.

root.bind('<Return>', startGame) // Funksioni bind() si parameter te pare ka ndonje tast te tastieres, kurse parametri i dyte definon funksionin qe deshirojme ta thirrim pasi klikojme ne ate tast te tastieres.

// startGame eshte funksioni i pare qe thirret pasi klikojme ndonje tast ne tastiere. Parametri event percakton qe ky funksion thirret vetem kur klikohet tasti.

def startGame(event):

global okToPressReturn

if okToPressReturn == False:

pass

else:

startLabel.config(text="")

updateHunger()

updateDay()

updateDisplay()

okToPressReturn = False

startLabel.config(text="") // Per editim te permbajtjes se elementeve te GUI-t definojme funksionin config()

// Permes funksionit updateDisplay() mundesohet ndryshimi i imazhit ne vartesi nga vlera e parametrit hungry.

def updateDisplay():

global hunger

global day

if hunger <= 50:

catPic.config(image = hungryphoto)

else:

catPic.config(image = normalphoto)

hungerLabel.config(text="Hunger: " + str(hunger))

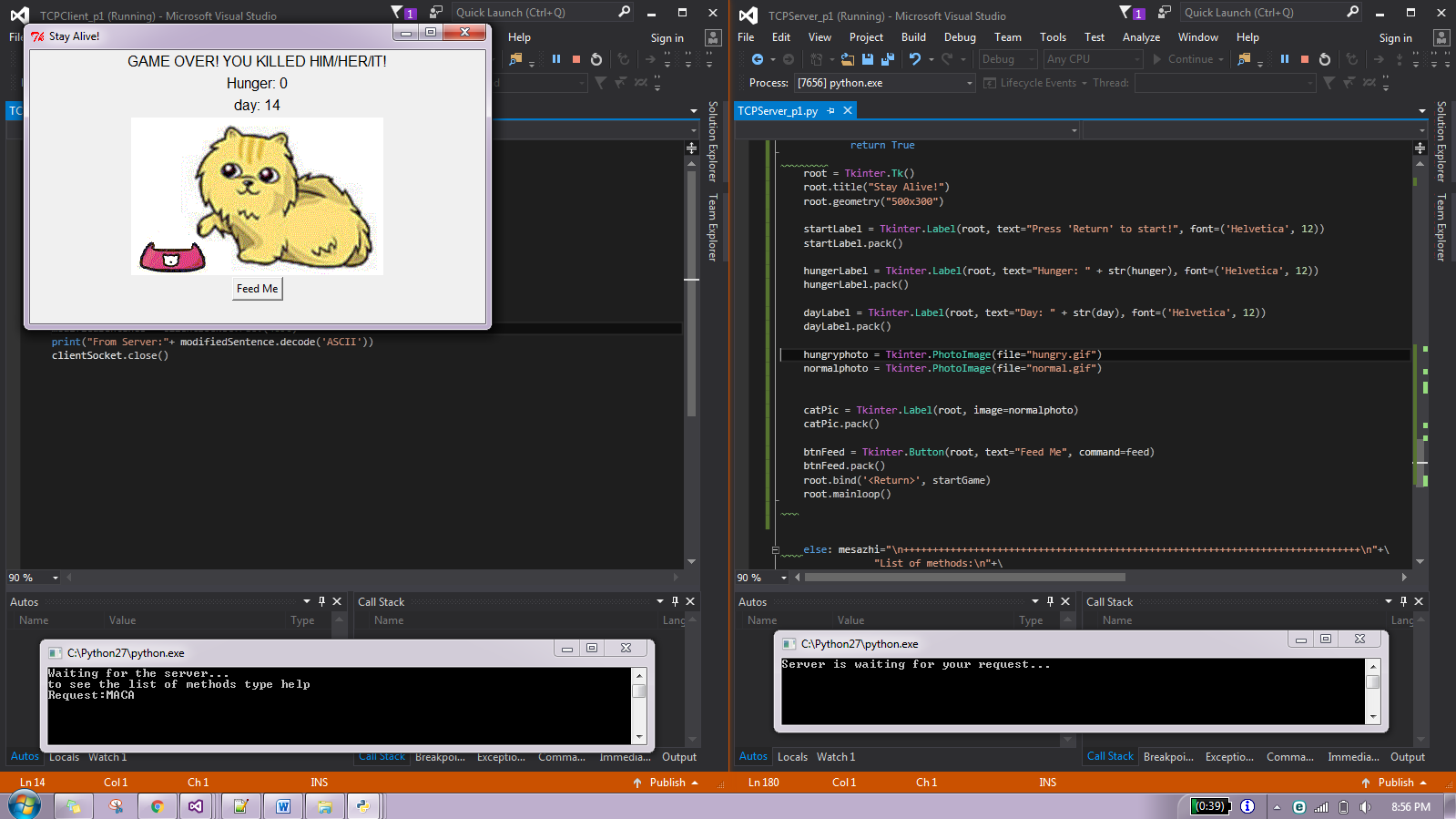
dayLabel.config(text="day: " + str(day))

catPic.after(100, updateDisplay)

catPic.after(100, updateDisplay) // Funksioni after mundeson thirrjen e funksionit pas disa milisekondave, ne kete rast funksioni updateDisplay thirret pas cdo 100milisekonda

Rezultatet pas testimit

Pasi klienti te beje kerkesen I kthehet pergjigjia nga serveri



# Pershkrimi i metodave te tjera

* + 1. Metoda POEM

Kerkesa :POEM

Realizimi permes kodit:

import random //importojme librarine random e cila pastaj na mundeson perdorimin e funksioneve te cilat ndodhen ne kuader

te kesaj librarie

random\_number = random.randint(0,100) // Ne menyre random marrim nje vlere nga 0 deri 100

#print(random\_number)

random\_number

failed\_attempts=0

x=True

while(x==True): // Blloku i komandave do ekzekutohet derisa vlera e variables x eshte True

guess\_number = int(input("Guess the number")) // Variable guess\_number merr nje vlere integer nga inputi

if(guess\_number>random\_number):

print("The number you've guessed is greater than the picked one") // Teksti i cili shfaqet nese numri qe kemi dhene eshte me i madh se ai i gjeneruar automatikisht

failed\_attempts+=1

elif(guess\_number<random\_number):

print("The number you've guessed is less than the picked one") // Teksti i cili shfaqet nese numri qe kemi dhene eshte me i vogel se ai i gjeneruar automatikisht

failed\_attempts+=1

else:

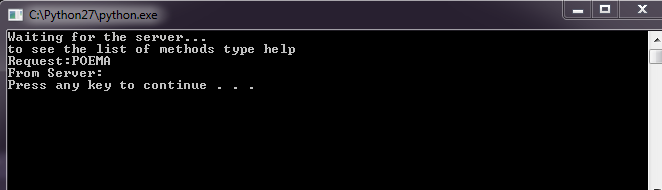
print("You found it !") // Nese e gjejme vleren e cila eshte gjeneru automatikisht vlera e variables x behet False dhe nuk ekzekutohet me unaza while

x=False

print("You tried %d"%failed\_attempts+"times")

Rezultatet pas testimit

Pasi klienti te beje kerkesen I kthehet pergjigjia nga serveri

******

******

* + 1. Metoda GUESS

Kerkesa :GUESS

Realizimi permes kodit:

from random import choice, randint // Nga libraria random importojme vetem metodat choice dhe randint

mbiemra = "brave clam delightful faithful gentle happy old young".split() // Definojme variablen mbiemra te tipit string. Funksioni split ndan fjaline ne fjale kur hase ne hapesire dhe merr formatin e listes

foljet = "wants love hears live burns brings becomes grows".split() // Definojme variablen foljet te tipit string

emrat = "sadness life books songs picture music unicorn long happiness joy ".split() // Definojme variablen emrat te tipit string

kryefjalet="nobody he she somebody ".split() // Definojme variablen kryefjalet te tipit string

// Definojme funksionin poema()

def poema():

print("The poem:")

for i in range(0,randint(5,10)):

// Per arsye se numri i antareve ne variablen kryefjalet eshte 4, permes randint(0,3) na mundesohet marrja e njerit prej anetareve.

// E njejta logjike perdoret edhe ne rastin e variablave tjera

print(kryefjalet[randint(0,3)]+" is "+mbiemra[randint(0,7)]+" and "+foljet[randint(0,6)]+" "+emrat[randint(0,9)])

// Thirrim funksionin poema()

poema()

Rezultatet pas testimit

Pasi klienti te beje kerkesen I kthehet pergjigjia nga serveri

