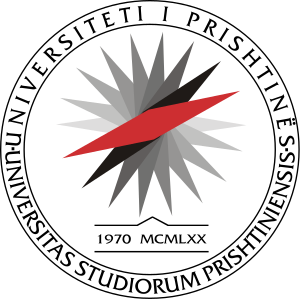
**UNIVERSITETI I PRISHTINËS**

**Fakulteti i Inxhinierisë Elektrike dhe Kompjuterike**



**Lënda: Rrjeta Kompjuterike**

**Titulli: Programimi me soketa**

**Studenti:** Eduard Spahija **Mentori:** Haxhi Lajqi

**ID:** 170714100077

**Data: 18.04.2019**

Përmbajtja

[1. Hyrje e shkurtër rreth projektit 3](#_Toc6522841)

[2. Ekzekutimi i FIEK-TCP Serverit dhe FIEK-TCP Klientit 4](#_Toc6522842)

[3. Metodat e protokolit FIEK-TCP 5](#_Toc6522843)

[3.1 IPADRESA 5](#_Toc6522844)

[3.2 NUMRIIPORTIT 5](#_Toc6522845)

[3.3 BASHKETINGELLORE 6](#_Toc6522846)

[3.4 PRINTIMI 6](#_Toc6522847)

[3.5 EMRIIKOMPJUTERIT 7](#_Toc6522848)

[3.5 KOHA 7](#_Toc6522849)

[3.6 LOJA 7](#_Toc6522850)

[3.8 FIBONACCI 7](#_Toc6522851)

[3.9 KONVERTIMI 8](#_Toc6522852)

[3.10 DECIMALTOBINARY 9](#_Toc6522853)

[3.11 DECIMALTOHEKSADECIMAL 9](#_Toc6522854)

[4. Ekzekutimi i FIEK-UDP Serverit dhe FIEK-UDP Klientit 10](#_Toc6522855)

[5. Metodat e protokolit FIEK-UDP 10](#_Toc6522856)

[6. Përmbledhje e rezultatit të testimit 11](#_Toc6522857)

# 1. Hyrje e shkurtër rreth projektit

Projekti përmban programet:

* FIEK-TCP Klienti
* FIEK-TCP Serveri
* FIEK-UDP Klienti
* FIEK-UDP Serveri

Veglat e përdorura për zhvillim të projektit:

* JetBrains PyCharm Community Edition 2019.1.1 x64

Sistemi Operativ ku është bërë testimi:

* Windows 10 Pro N

Metodat e implementuara në projekt

* Të obliguara:
  + - IPADRESA
    - NUMRIIPORTIT
    - BASHKETINGELLORE
    - PRINTIMI
    - EMRIIKOMPJUTERIT
    - KOHA
    - LOJA
    - FIBONACCI
    - KONVERTIMI
* Shtesë
  + - DECIMALTOBINARY
    - DECIMALTOHEKSADECIMAL

# Ekzekutimi i FIEK-TCP Serverit dhe FIEK-TCP Klientit

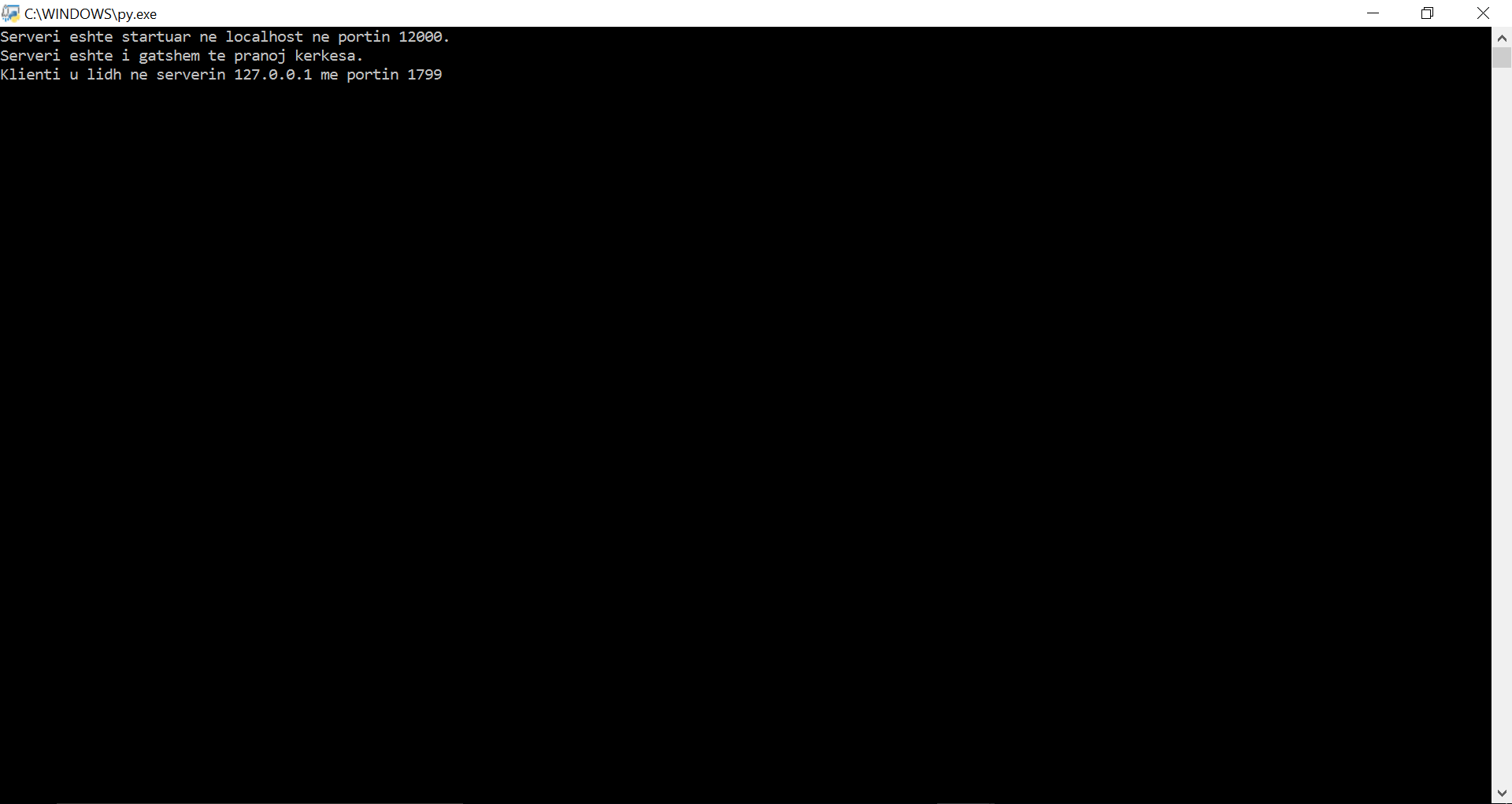
Versionet e Klientëve e Serverëve janë zhvilluar në Python Console Application.

Përmes tyre është paraqitur komunikimi i Klientit me Server përmes TCP protokollit, duke përdorur Soketat. Më poshtë është paraqitur procedura e testimit:

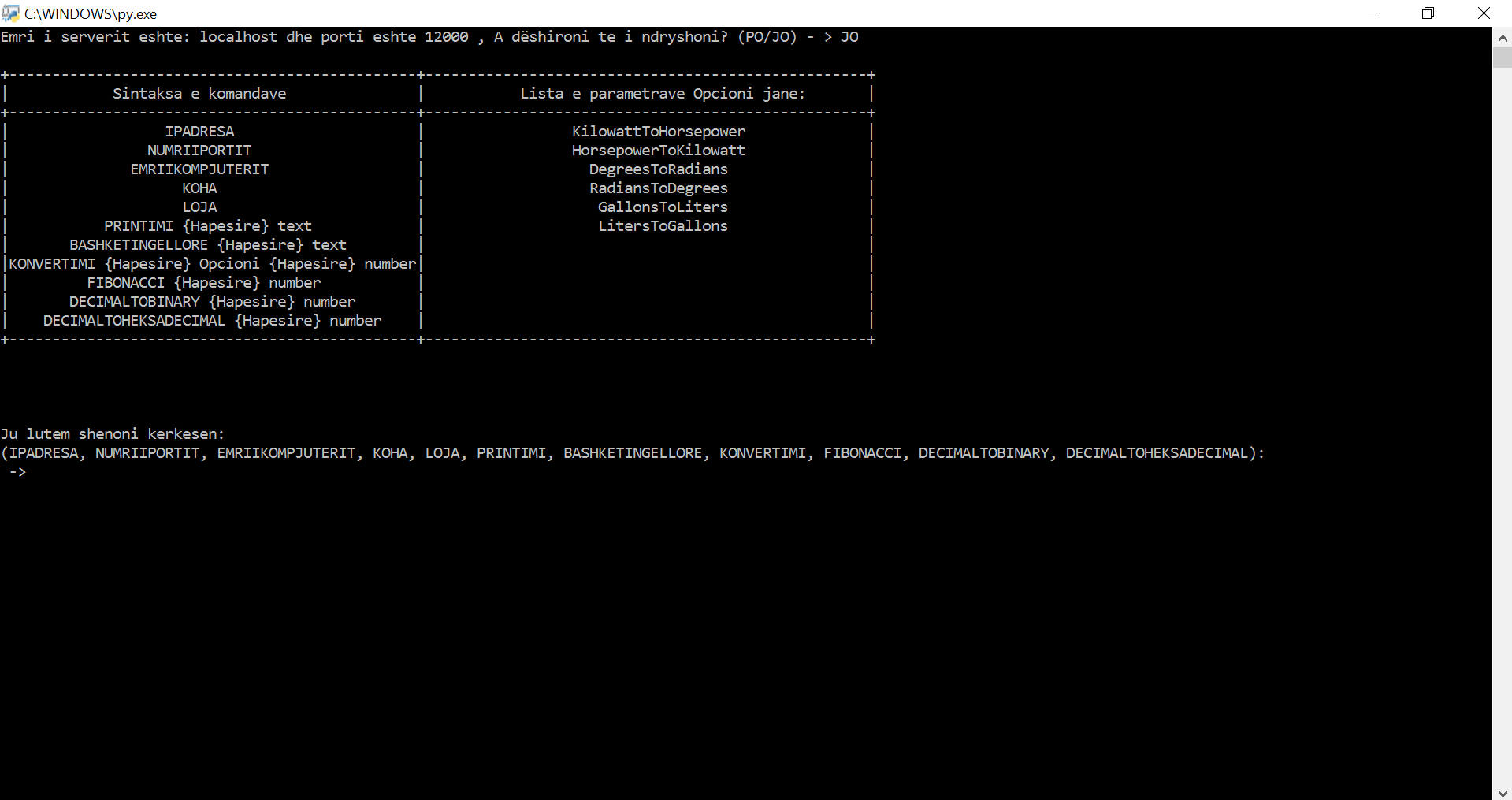
Pas ekzekutimit të TCP Serverit dhe TCP Klientit shfaqen rezulltatet në ekran me mesazhet si në vazhdim:

Së pari duhet të ekzekutohet Serveri e pastaj Klienti.

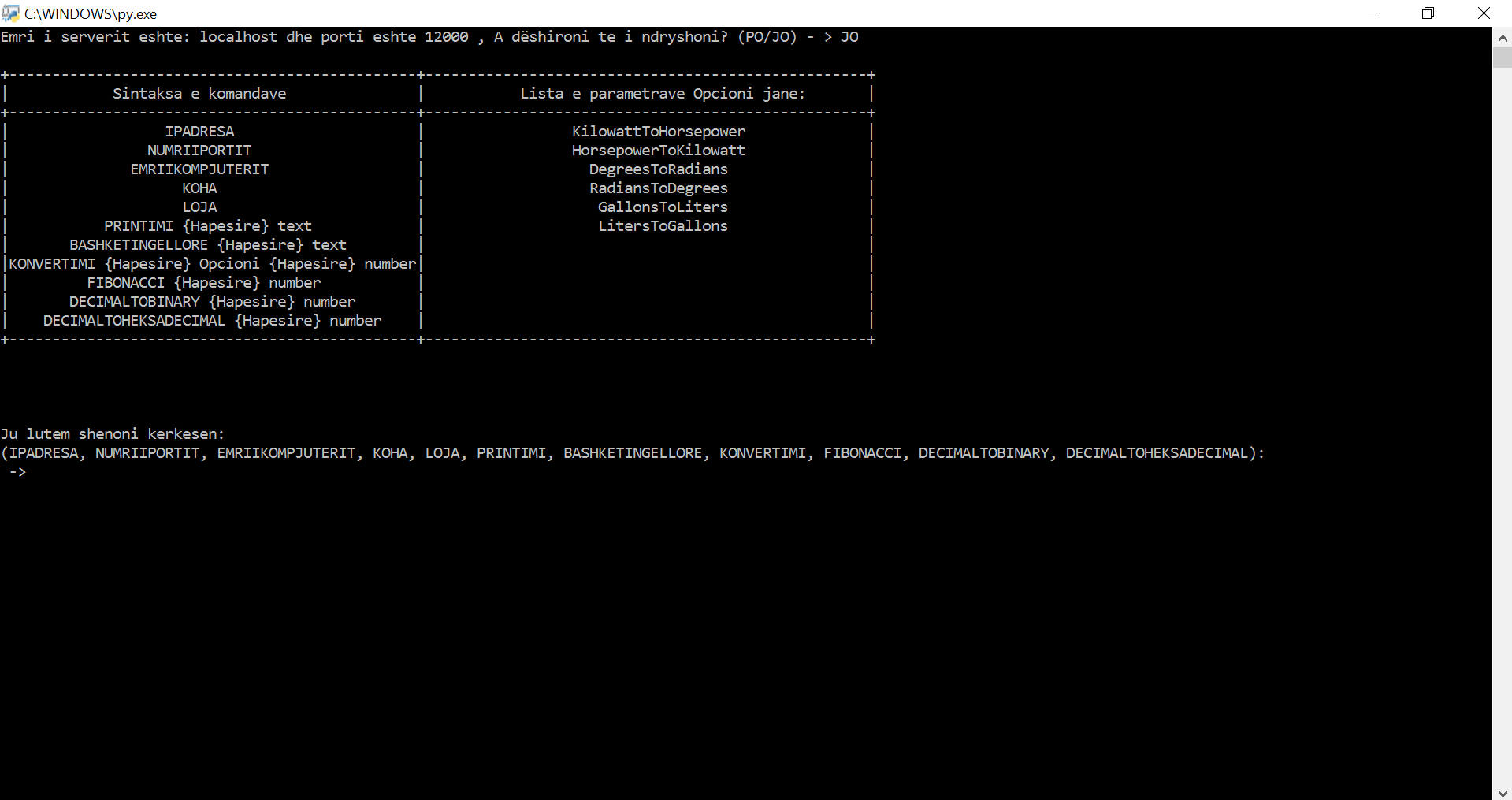
FIEK-TCP SERVERI



FEK-TCP KLIENTI



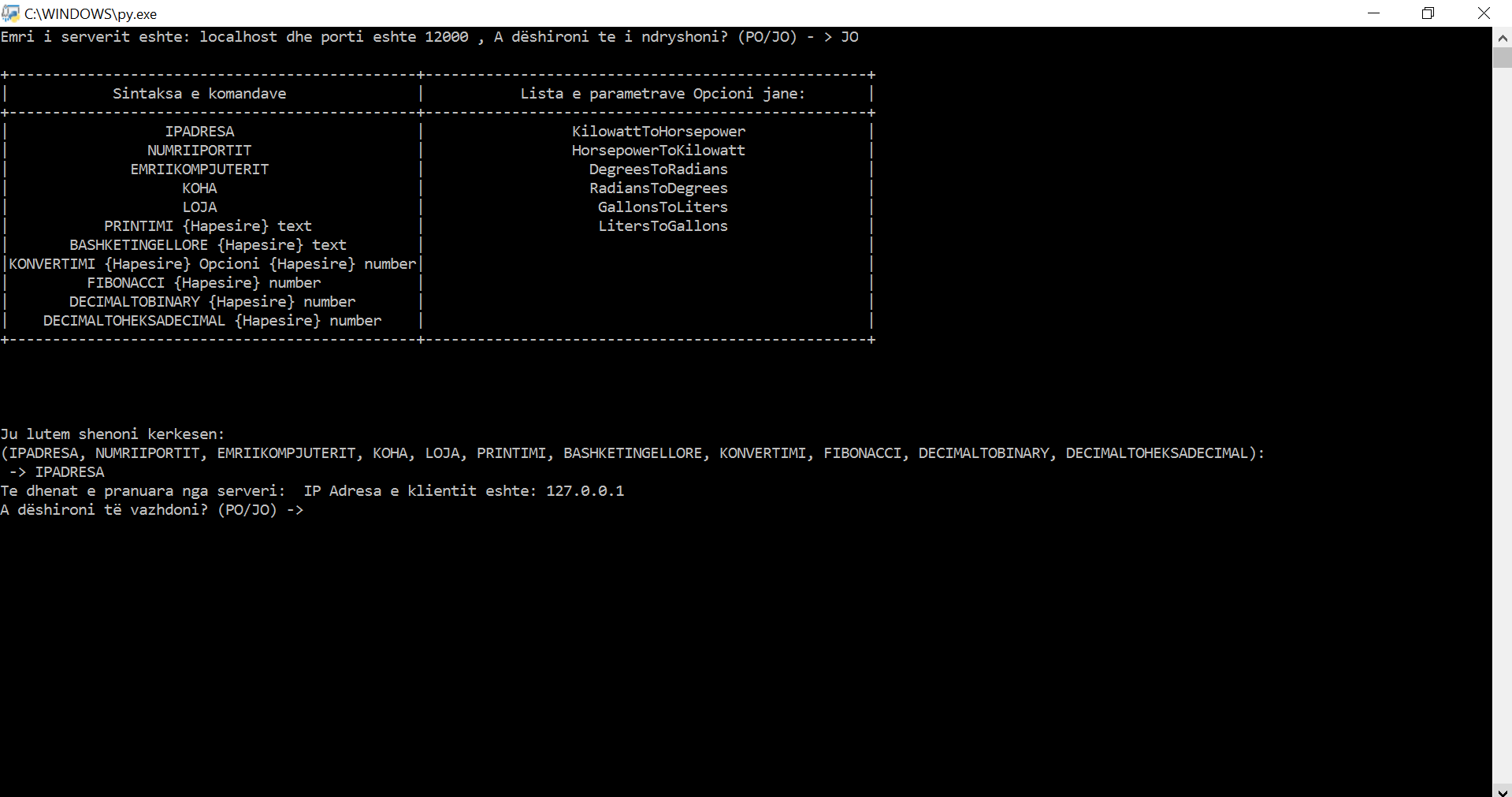
Pastaj vazhdojmë me testimin e komandave, ku paraprakisht shfaqen sintaksa e kërkesave bashkë me sintaksat dhe parametrat e caktuara. Vetëm për kërkesat e tilla serveri mund të kthej rezultat, në të kundërt kthen përgjigje si Kerkesa eshte jo valide!



# Metodat e protokolit FIEK-TCP

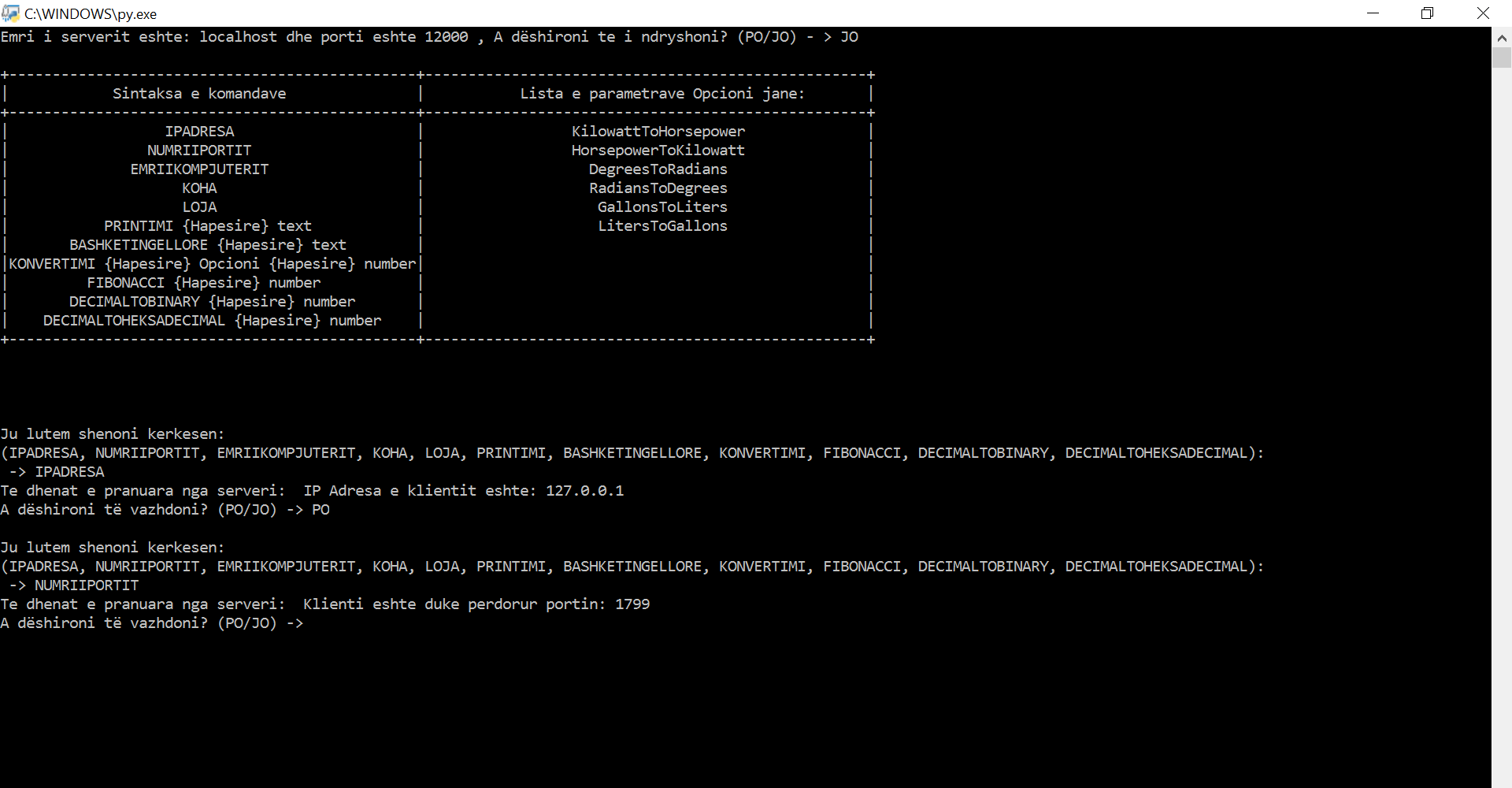
## 3.1 IPADRESA

Për të ditur IP Adresën e **Klientit**, klienti dërgon kërkesën te **Serveri** me emrin *“*IPADRESA*”,* me ç’rast serveri pranon kërkesën e klientit dhe i kthen atij IP Adresën e klientit, sikurse shihet në vijim:



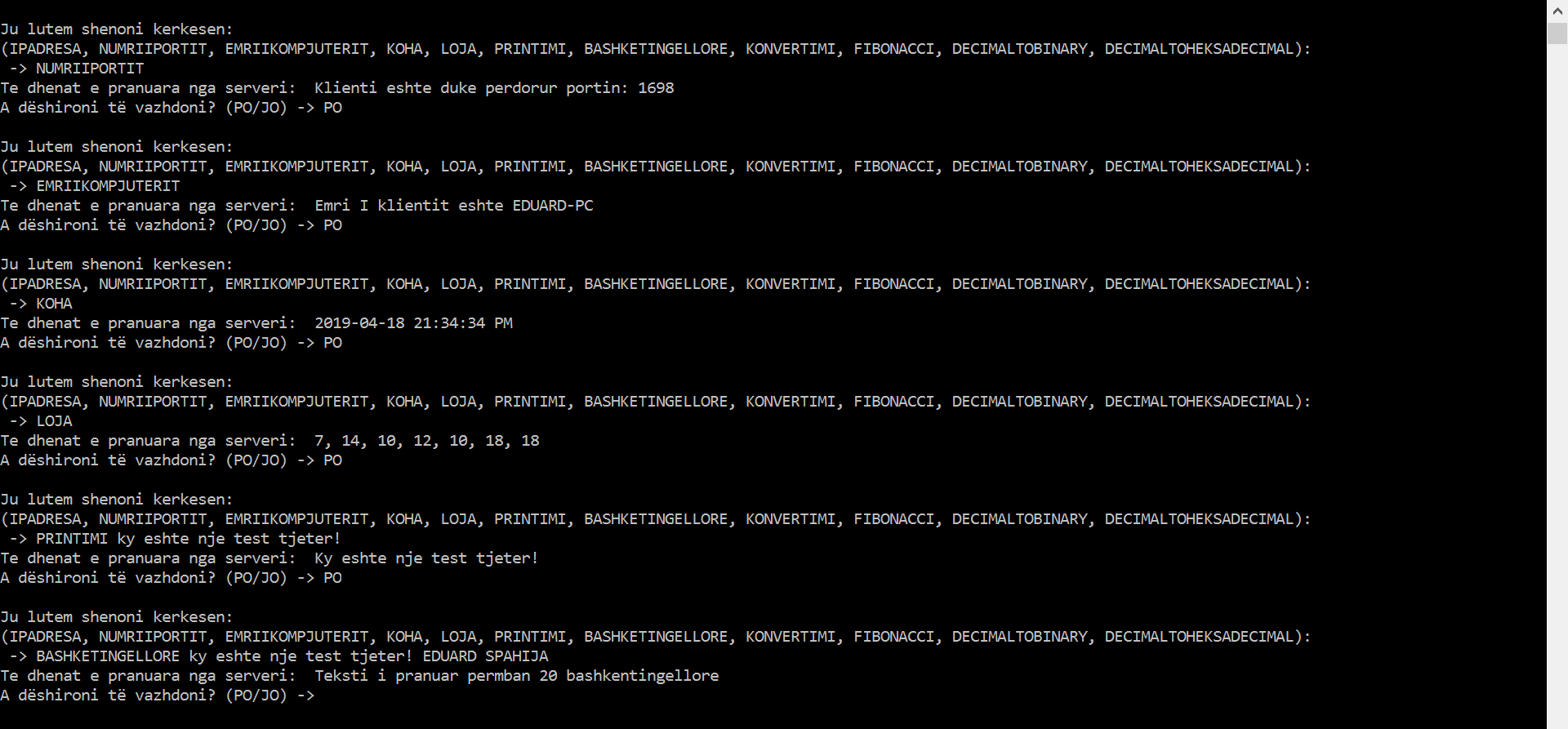
3.2 NUMRIIPORTIT

Për të ditur Portin e komunikimit të **Klientit**, klienti dërgon kërkesën te **Serveri** me emrin *“*NUMRIIPORTIT*”,* me ç’rast serveri pranon kërkesën e klientit dhe i kthen atij Portin me të cilin ai komunikon me server, sikurse shihet në vijim:



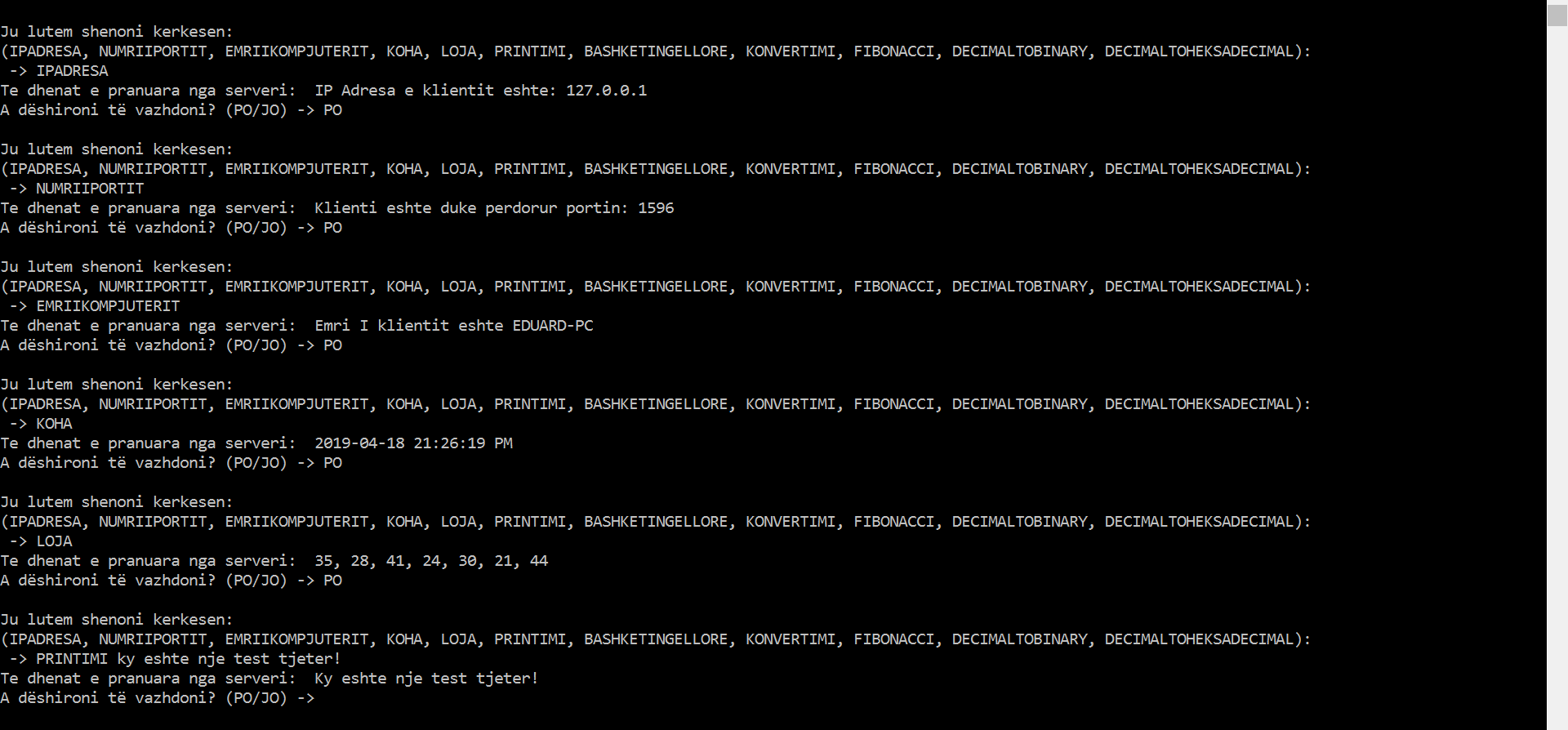
* 1. BASHKETINGELLORE

Përmes kësaj metode **Klienti** kërkon nga **Serveri** të gjej numrin e bashkëtingëlloreve që përmban një tekst dhe **Serveri** kthen numrin e bashkëtingëlloreve

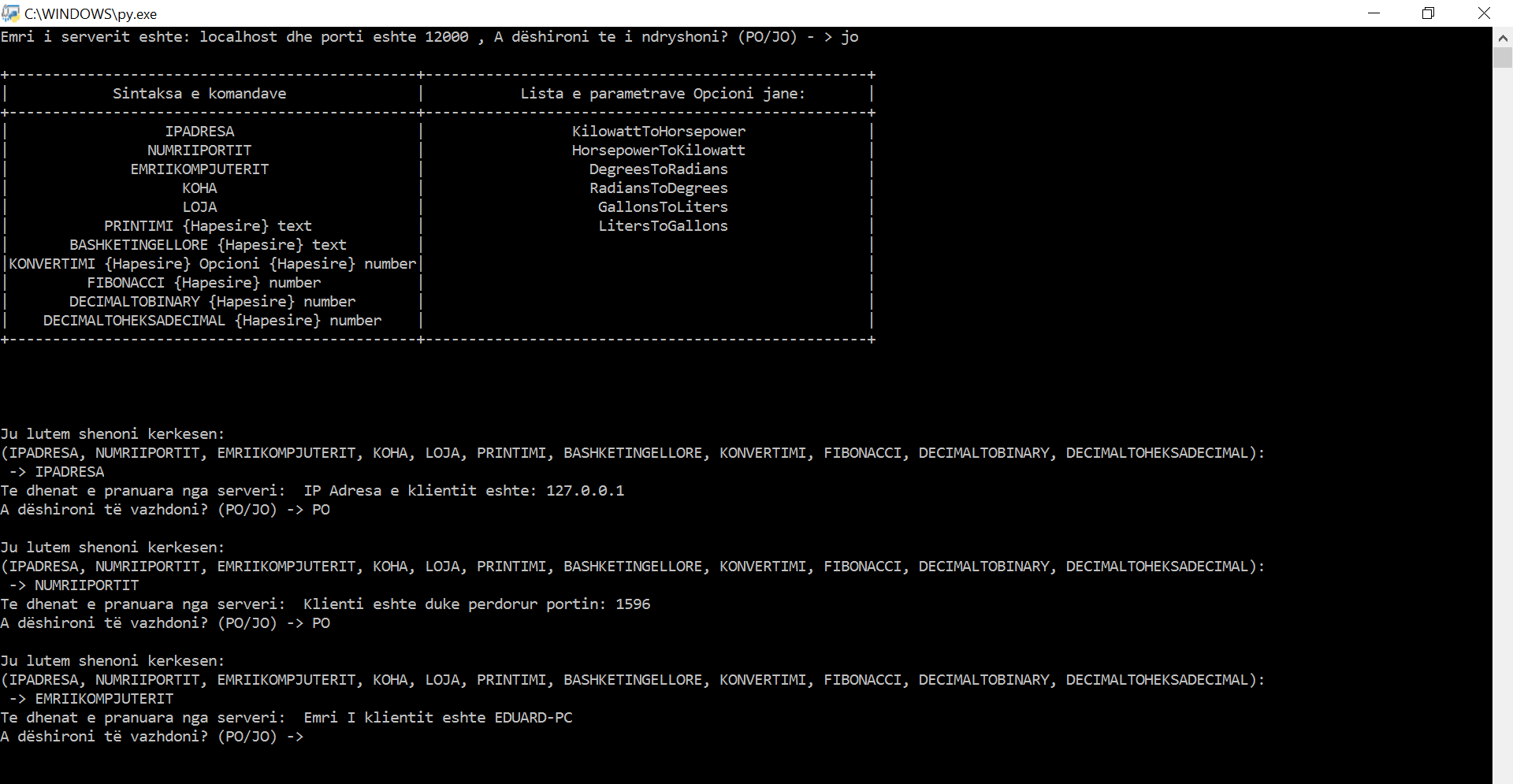


* 1. PRINTIMI

Përmes metodës *“*PRINTIMI*”* **Serveri** i dërgon **Klientit** kërkesën për shënimin e tekstit të dëshiruar nga **Klienti** dhe pastaj mesazhin e njëjtë që Klienti ka shtypur, Serveri ia kthen prapë Klientit. Figura në vazhdim:

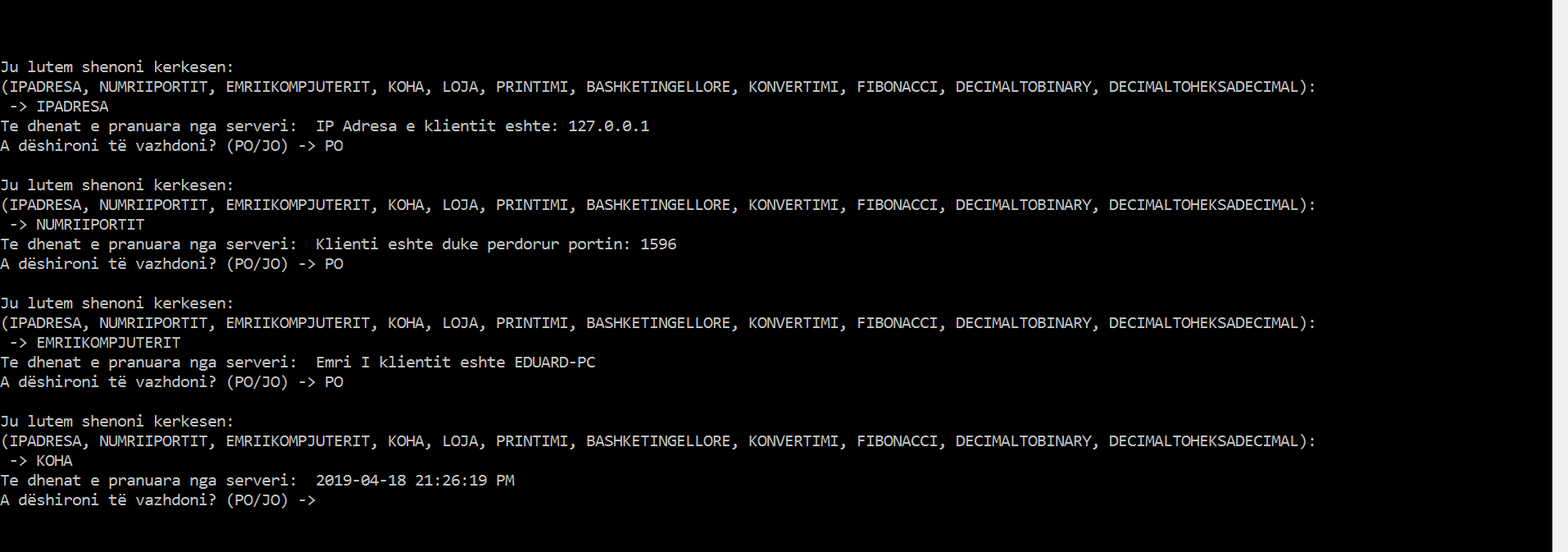


3.5 EMRIIKOMPJUTERIT

Kjo është metodë që **Serveri** i kthen **Klientit** emrin e kompjuterit të **Klientit**, e në rast se ai nuk dihet nga ana e **Serverit**, **Serveri** kthen mesazhin “Emri i klientit nuk dihet”, sikurse në figurën më poshtë:  
  


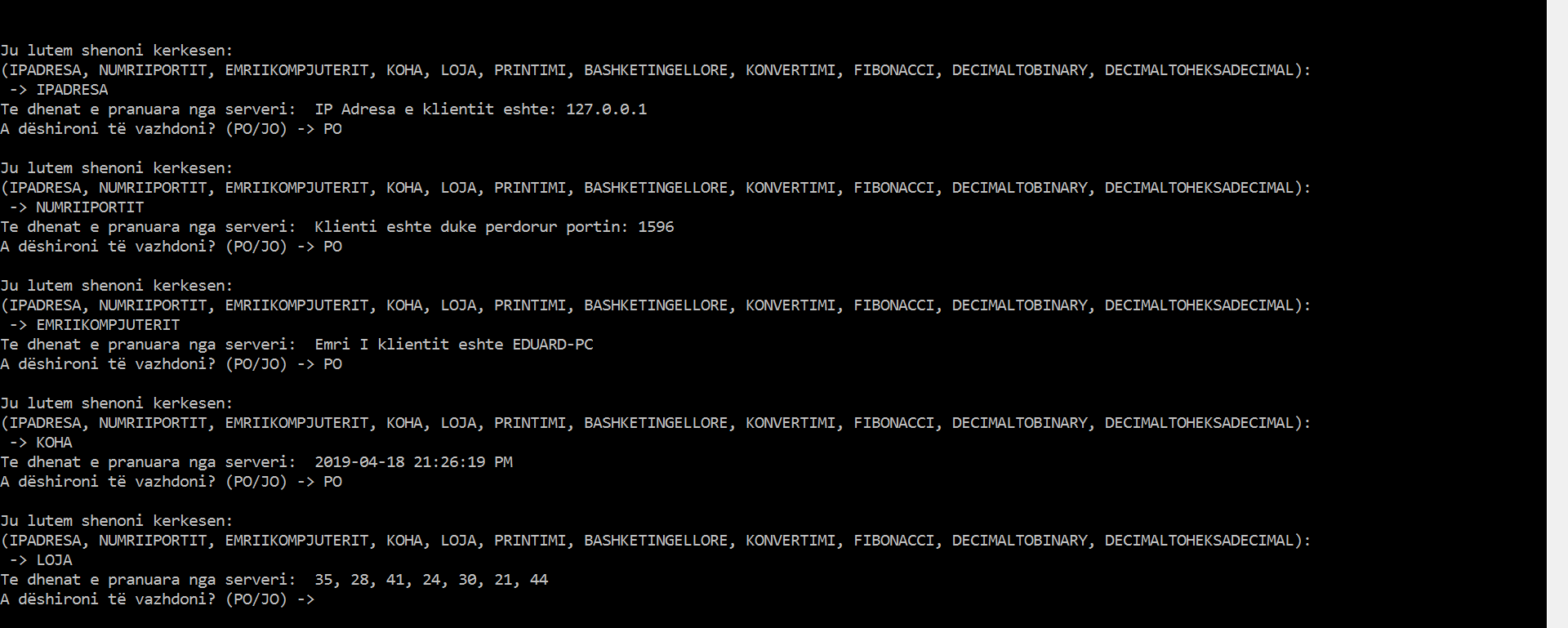
* 1. KOHA

Metoda “KOHA” i kthen **Klientit** datën dhe kohën aktuale të **Serverit**, sikurse në figurën në vijim:



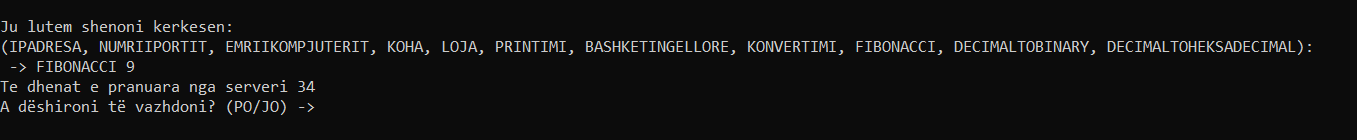
* 1. LOJA

Kjo metodë kthen 7 numra random nga rangu 1-50.



3.8 FIBONACCI

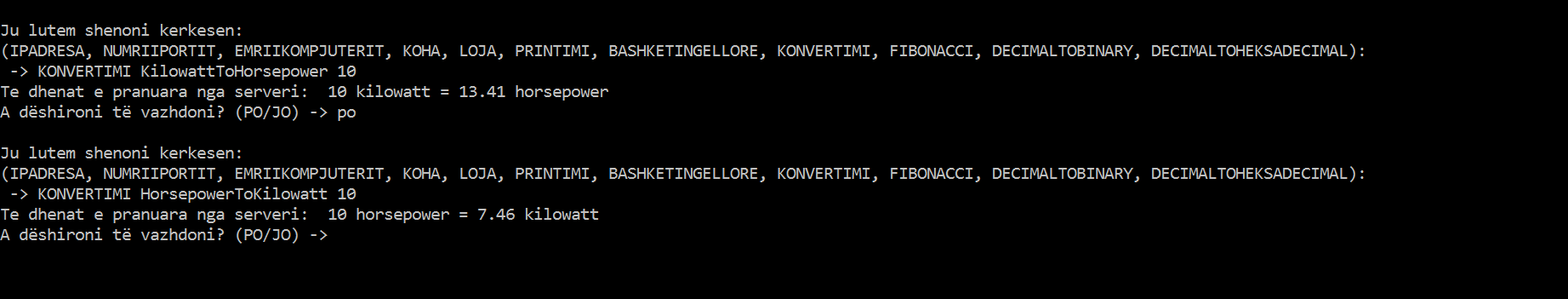
Kjo metodë llogaritë FIBONACIN të një numri.



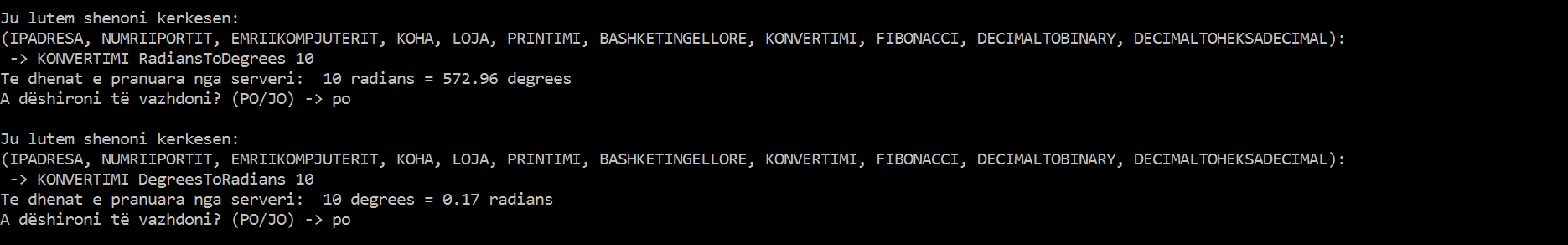
3.9 KONVERTIMI

Kjo metodë konverton një njësi në tjetrën. Opcionet e mundshme që janë implementuar me këtë protocol janë:

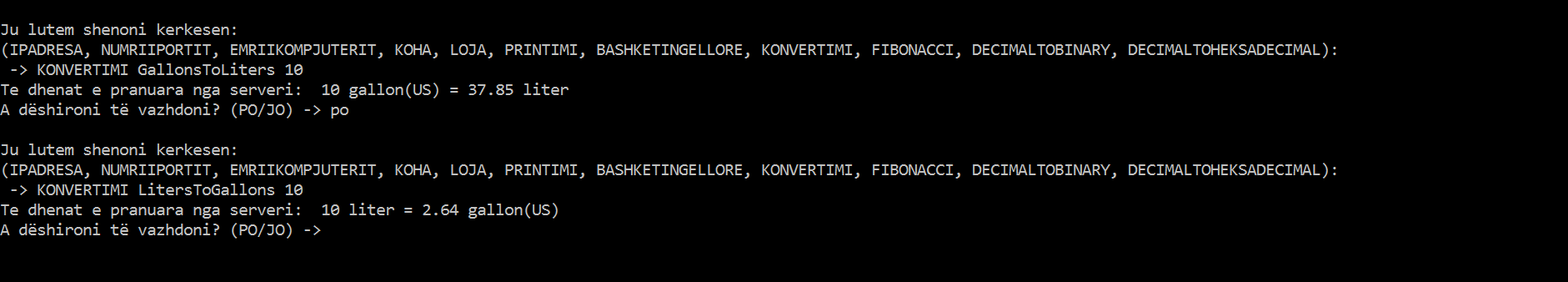
* KilowattToHorsepower
* HorsepowerToKilowatt



* DegreesToRadians
* RadiansToDegrees

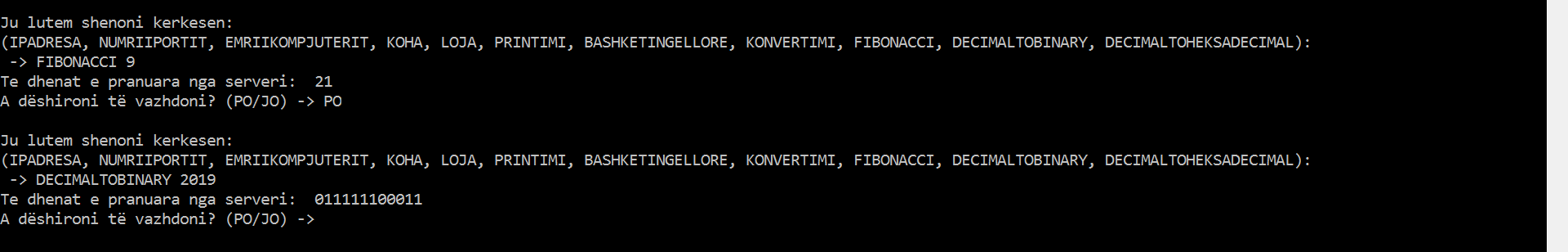


* GallonsToLiters
* LitersToGallons



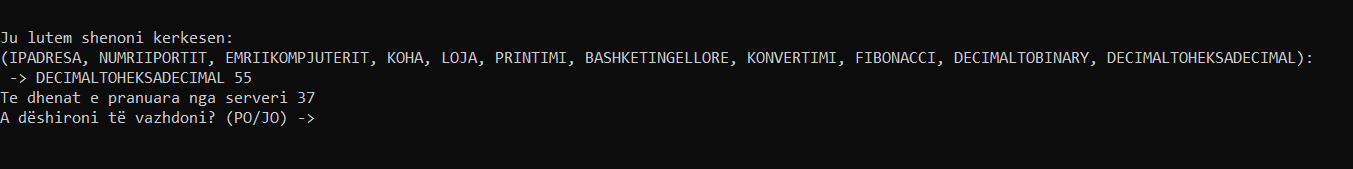
3.10 DECIMALTOBINARY

Kjo metodë shëndron numrin decimal në binar.

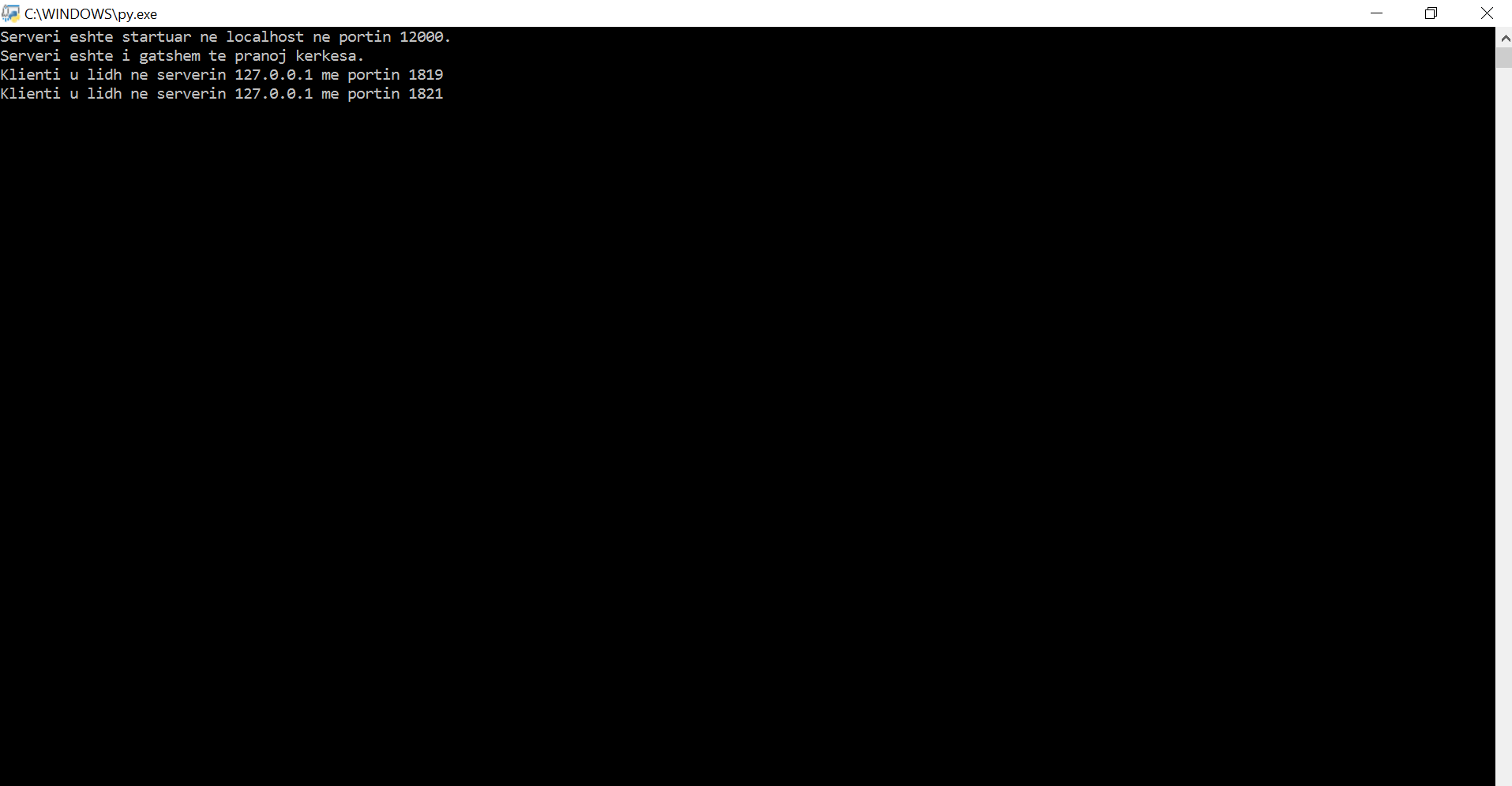


3.11 DECIMALTOHEKSADECIMAL

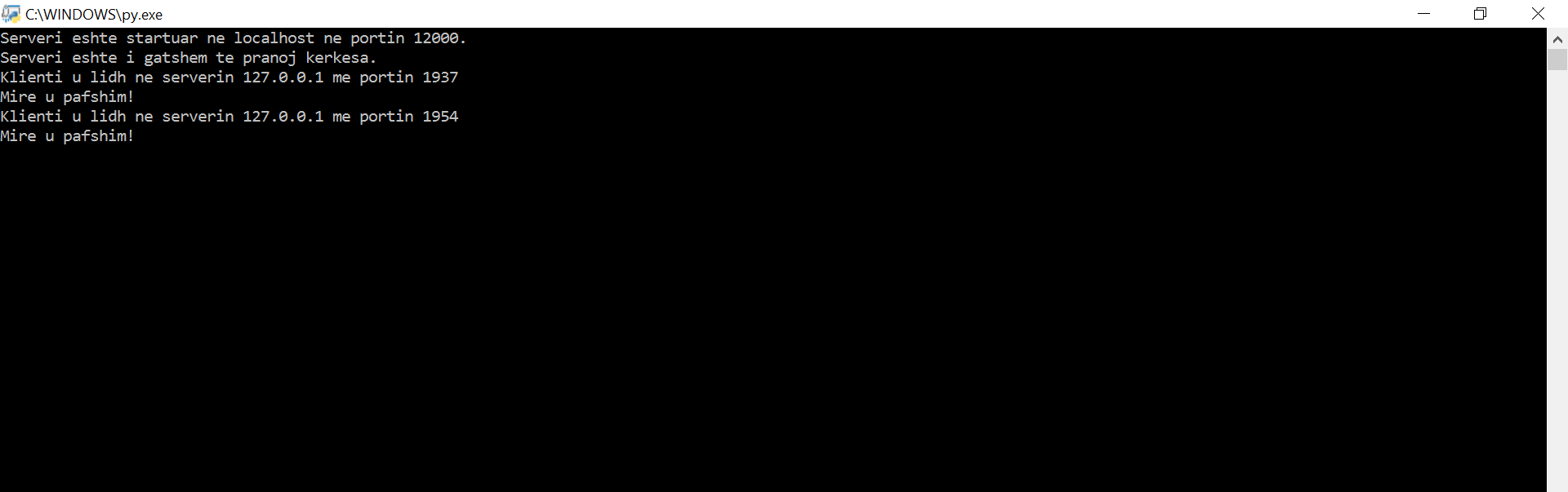
Kjo metodë shëndron numrin decimal në heksadecimal.



Gjithashtu **Serveri** mund të pranojë kërkesa nga shumë **Klient** siq shihet në vazhdim:



Ndërsa lidhja e klientit e serverit përfundon pas mesazhit te Klienti “A dëshironi të vazhdoni?” dhe përgjigjes së tij me JO, siq shihet në vijim:



# Ekzekutimi i FIEK-UDP Serverit dhe FIEK-UDP Klientit

Versionet e Serverëve e Klientëve janë zhvilluar në Python Console Application.

Përmes tyre është paraqitur komunikimi i Klientit me Server përmes UDP protokollit, duke përdorur Soketat. Më poshtë është paraqitur procedura e testimit:

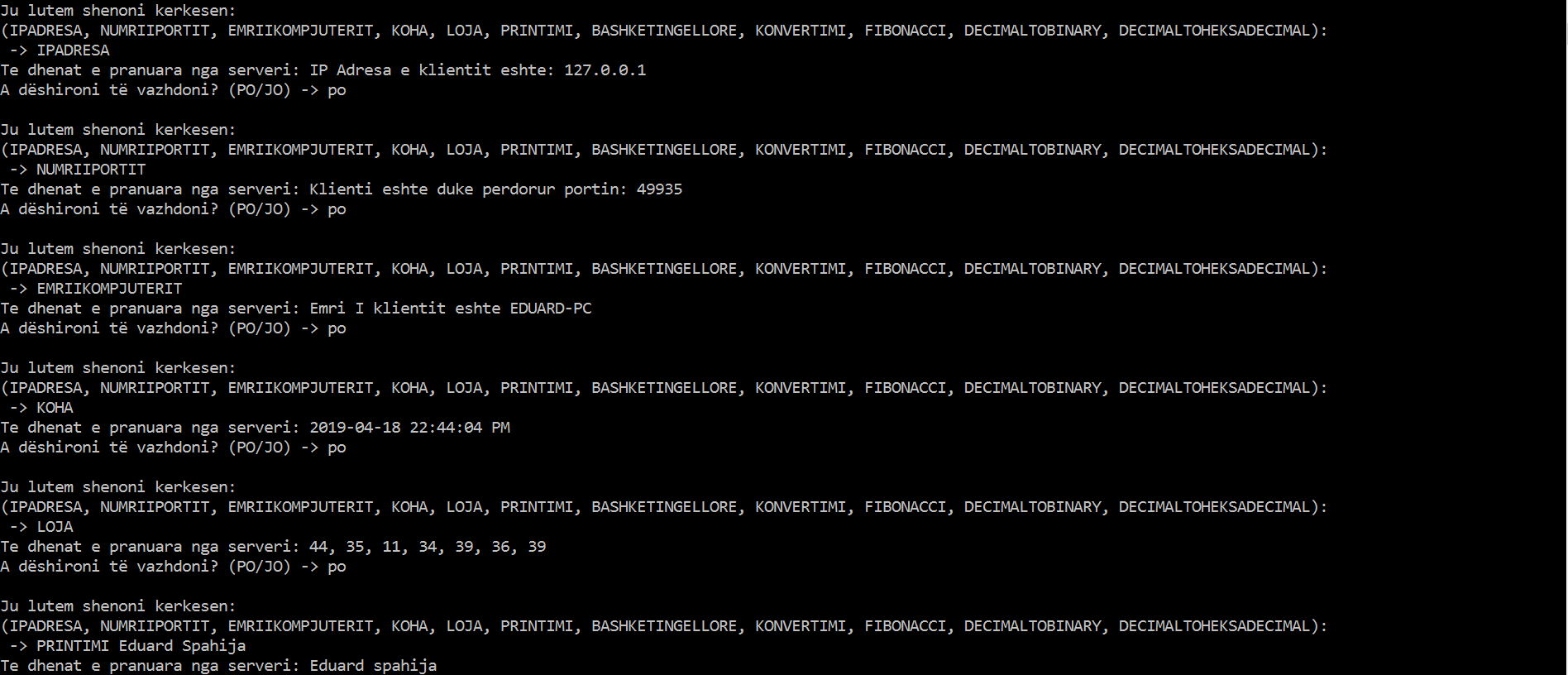
Pas ekzekutimit të UDP\_Serverit dhe UDP Klientit shfaqen rezulultatet në ekran, me mesazhet përkatëse:

Së pari duhet të ekzekutohet Serveri e pastaj Klienti.

# Metodat e protokolit FIEK-UDP

Metodat e protokolit FIEK UDP janë të njëjta me ato të protokolit FIEK TCP

Në vazhdim po testojmë disa nga metodat:



# Përmbledhje e rezultatit të testimit

Me anë të testimit të punës së Serverit dhe Klientit TCP dhe UDP vijmë në përfundim se ata janë duke funksionuar ashtu si jane definuar me projekt

Të gjitha metodat kthejnë përgjigjet e duhura dhe nuk vërehet ndonjë defekt.

Pra programimi Socket është një mënyrë për të lidhur dy nyje në rrjet për të komunikuar me njëri-tjetrin. Një socket (nyje) dëgjon në një port të caktuar në një IP, ndërsa socket-i (nyja) tjetër lidhet dhe dërgon kërkesa te tjetri dhe keshtu formohet një lidhje. Serveri formon socket-in (nyjen) qe i dëgjon apo është në pritje të kerkesave nga klienti ndërsa klienti i dërgon kërkesa  serverit.

TCP-Transmission Control Protocol, garanton se marrësi do të marrë pakon. Marrësi i dërgon mesazhet përsëri dërguesit duke thënë se i ka marrë mesazhet. Nëse dërguesi nuk merr një përgjigje të saktë, do të ridërgoj paketat për të siguruar që marrësi i ka marrë ato. Paketat gjithashtu kontrollohen për gabime. TCP është e gjitha në lidhje me këtë besueshmëri - paketat e dërguara me TCP janë gjurmuar kështu që asnjë e dhënë nuk humbet ose nuk prishet në transmetim.

UDP-User Datagram Protocol, paketat i dërgohen vetëm tek marrësi. Dërguesi nuk do të presë për t'u siguruar që marrësi ka marrë paketën - do të vazhdojë të dërgojë paketat e ardhshme. Nëse jeni marrësi dhe ju mungon ndonë jpako UDP, ju nuk mund t'i kërkoni këto pako përsëri. Nuk ka asnjë garanci që po merr të gjitha paketat dhe nuk ka asnjë mënyrë për të kërkuar përsëri një pako nëse e humbisni, por humbja e gjithë kësaj lartësie do të thotë që kompjuterat mund të komunikojnë më shpejt. UDP përdoret kur shpejtësia është e dëshirueshme dhe korrigjimi i gabimit nuk është i nevojshëm. Për shembull, UDP përdoret shpesh për transmetime të drejtpërdrejta dhe lojëra online.