

# **3. PRAKTIKA: SOCKET-AK**

***2020-2021 IKATURTEA***

**Konputagailu Sareen Oinarriak**

Irakaslea: Gaizka Saldaña

SISTEMEN INGENIARITZA ETA AUTOMATIKA SAILA

## PRAKTIKAREN HELBURUA

Praktika honekin ikaslea komunikazio aplikazio baten funtzionamendua ulertzea nahi da, java socket-ak erabiliz. Horretarako, dagoeneko implementatutako aplikazio bat aztertu, exekutatu eta egokituko du.

Praktika honen lehen zatia praktika martxan jartzeko erabiltzen den eklipsearen garapen inguruneari buruzko sarrera laburra da. Bigarren zatian Java-ko socket programazioaren oinarriak definituko dira, azkenean sare batean konektatutako bi ordenagailuren arteko mezularitza aplikazioarekin lan egiteko.

## 1.- SORTU JAVA APLIKAZIO BAT ECLIPSE-rekin

Eclipse Open Source software plataforma bat da. Plataforma hau garapeneko ingurune integratuak (IDE) sortzeko erabiltzen da, hala nola Java IDE (Java Development Toolkit JDT). Eclipse IBM-k VisualAge-ren oinarriko gisa garatu zuen. Orain Eclipse Fundazioaren esku dago, irabazi asmorik gabeko erakunde independentea, plataforma irekiko komunitatea bultzatzen duena.

Eclipse <http://www.eclipse.org/downloads> webgunetik deskarga daiteke.

Exekutagarria **eclipse.exe** fitxategia da.

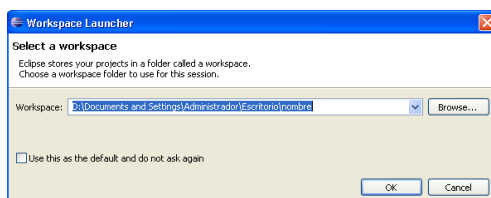


### **Laborategian aplikazioa instalatuta dago.**

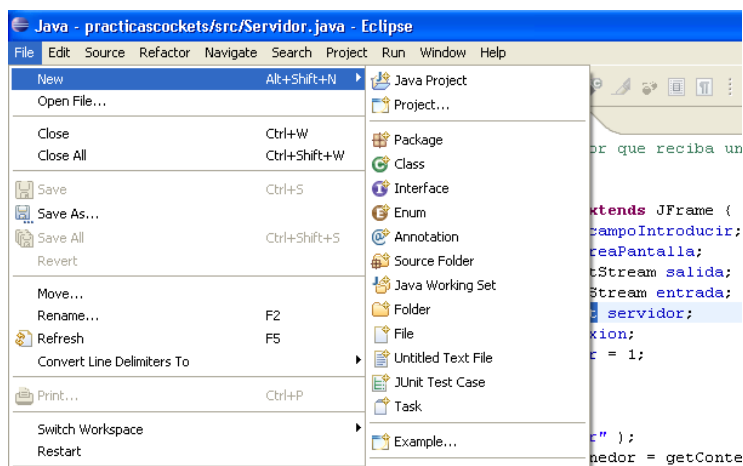
Aplikazioa exekutatzeko, lan espazioko direktorio bat eskatzen da saio honetarako. Direktorio bat sor dezakezu *escritorioan* zure inzialekin (Gogoratu zure lana gorde nahi baduzu, *escritorioa* ez dela tokia).

Adb: *D:\Documents and Settings\alumno\Escritorio\izena*.

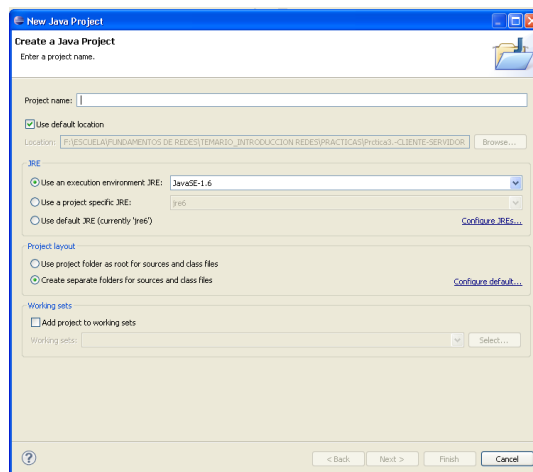
"OK" sakatuko da jarraitzeko:



Eclipse-ko programa guztiek "Proiektua" sortzea eskatzen dute, horretarako aukeren menuan hautatu behar da: **"File" - "New" - "Java Project"**

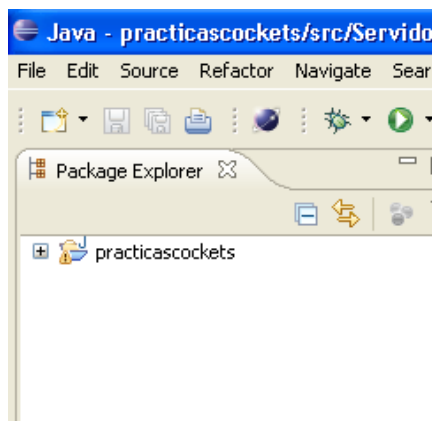


Elkarrizketa-menua agertzen da proiektuaren izena zehazteko:

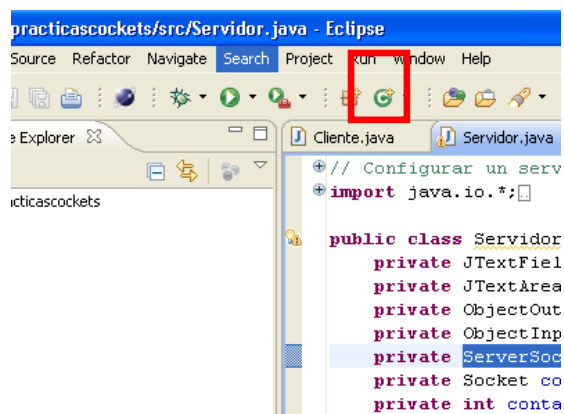
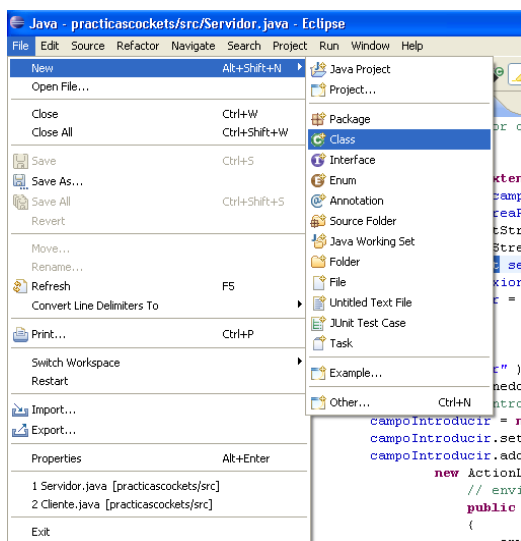


"Proiektuaren izena" testu-eremuan, jarri izena eta utzi gainerako aukera guztiak elkarrizketan, lehenetsitako balioekin. "**Finish**" botoia sakatuko dugu.

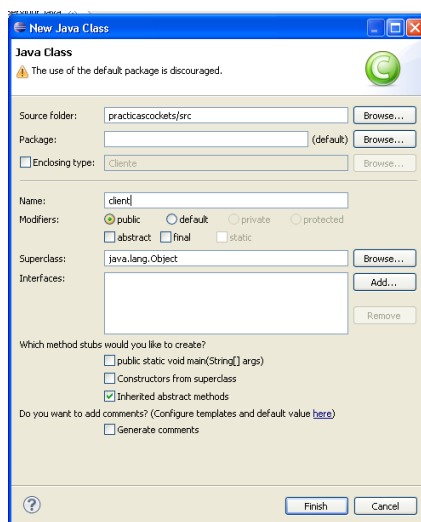
Orain "**Package**" leihoan sortu berri den proiektua agertzen da:



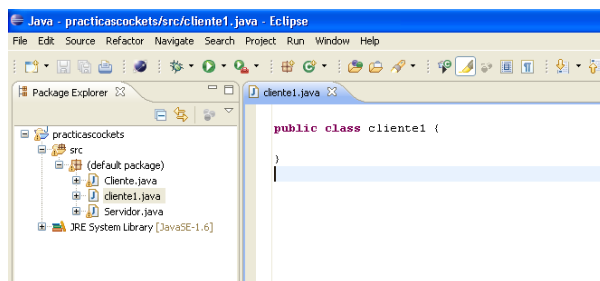
Java programa guztiek gutxienez klase bat behar dute. Klase bat sortzeko klasea aukeren menuan edo eklipse ikonoaren barran hautatuta egin daiteke.



Agertzen den elkarrizketan, klasearen izena definitu behar duzu (Bezero klase bat eta beste Zerbitzari bat definitu). Elkarrizketan beste datuak balio lehenetsiekin geratzeko dira.



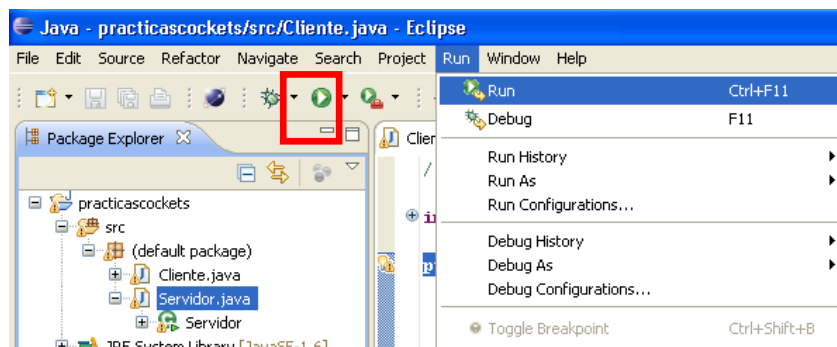
"Finish" botoia sakatzen da eta klasea kodetzeko fitxategia lortzen da.



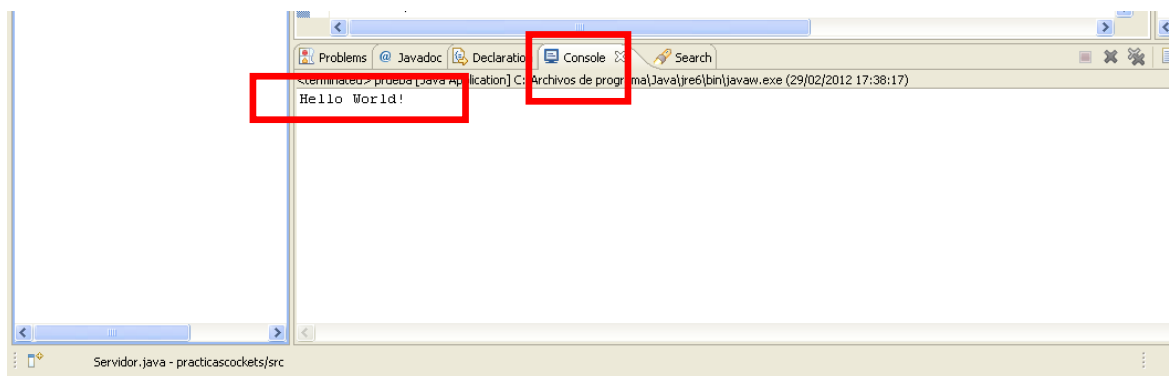
Ingurunea probatu ahal izateko, jarri honako kodea klasean

```
public static void main(String[] args) {
    // Display "Hello World!"
    System.out.println("Hello World!");
}
```

Aukeratu klasea eta exekutatu aukera menuan edo dagokion ikonoarekin.

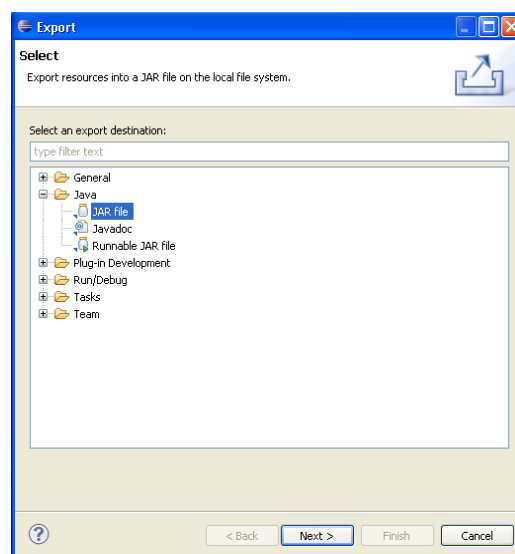
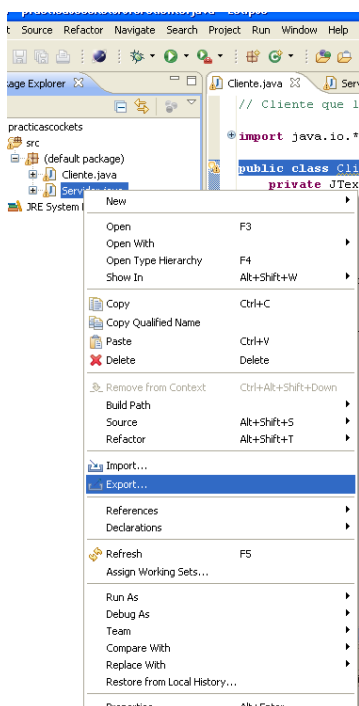


Kodetze akatsik ez badago, exekuzioaren emaitza behean agertzen den "Console" izeneko Eclipse leihoan ikusiko duzu.

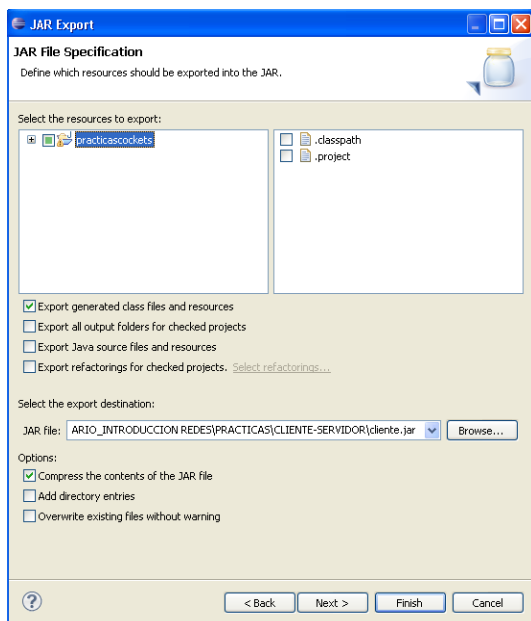


## NOLA SORTU EXEKUTAGARRIA

Eclipse-tik jar "exekutagarria" sortzeko, klasea hautatuko dugu eta eskuineko botoia sakatuta goitibeherako menua agertuko da eta bertan **"Export"** hautatuko dugu.



Agertzen den leihoan "java" zuhaitza bistaratzen da eta **"Runnable JAR file"** aukeratuko dugu. Aukeratu "Next". Ondorengo goitibeherako menua agertzen da non fitxategia izendatzen den eta irteerako direktorioa esleitzen zaion. "Finish" hautatuta fitxategi exekutagarria sortuko da.



## 2.- SOCKET-AK JAVA-N

Socket bat garraio geruzan prozesuen arteko interkomunikazioa ahalbidetzen duen datu sarrera-irteera interfazea da. **IP helbide + TCP ataka** zenbaki baten konbinazioa helbide gisa erabiltzen da.

Socketak prozesu bati beste prozesu batekin hitz egitea (informazioa igortzea edo jasotzea) ahalbidetzen duten prozesuen arteko komunikazio puntuak edo mekanismoak baino ez dira, nahiz eta prozesu horiek makina desberdinetan egon. Socketek prozesuen arteko komunikazioa ahalbidetzen dute, telefonoek pertsonen arteko komunikazioa ahalbidetzen duten bezala.

Parte hartzen duten prozesuen bidez socket bat erreferentziatzeko modua fitxategiak erreferentziatzeko erabiltzen den mota bereko deskribatzaile baten bidez egiten da. Socket eragiketak modu askotako fitxategien eragiketen antza du:

- Beharrezkoa denean socket bat **sortzen da**, eta sistema eragileak balio txikiko zenbaki oso bat eskaintzen du (socket deskriptorea, fitxategi deskribatzailearen antzekoa), geroago erreferentzia egiteko erabiltzen dena.
- Socket datuak transferitzeko eragiketak fitxategiak **irakurtzeko eta idazteko** eragiketak bezalakoak dira.
- Komunikazioak amaitzen direnean socket bat **ixten** da fitxategi gisa.

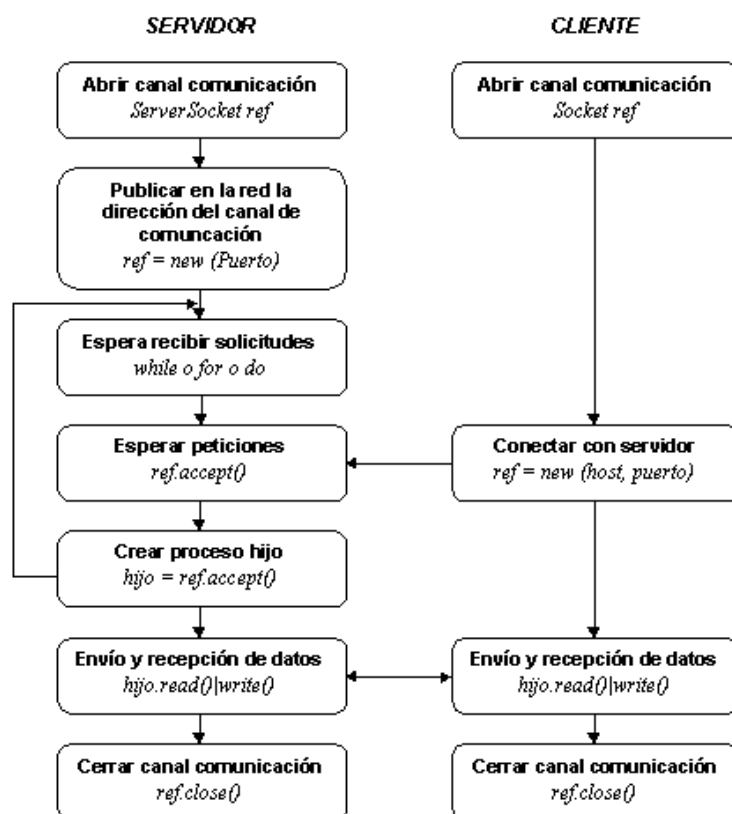
Socket bidez prozesuen arteko komunikazioa **CLIENT-SERVER** filosofian oinarritzen da: komunikazio honetako prozesu batek **zerbitzariaren prozesu gisa funtzionatuko du bezeroaren prozesuak bere izena ezagutuko duen socket bat sortuko duena**, zerbitzariaren prozesuarekin "hitz egiteko" gai izango dena izendatutako socket horretarako konexioaren bidez.

Bezeroaren prozesuak **izenik gabeko socket bat sortzen du**, itzulerako balioa deskribatzailea izanik, bertan irakurri edo idatziko da, norabide biko komunikazioa ahalbidetuz, socketen ezaugarria eta hodietatik desberdina dena, edo norabide bakarreko komunikazio kanalak makina bereko prozesuen artean. Socket bidez komunikatzeko mekanismoak urrats hauek ditu:

- I. Zerbitzariaren prozesuak izendatutako socket bat sortzen du eta konexioaren zain dago.
- II. **Bezeroaren** prozesuak izenik gabeko socket bat **sortzen** du.
- III. **Bezeroaren** prozesuak **konexio eskaera egiten dio zerbitzariaren** socketari.
- IV. Bezeroak konexioa bere socketaren bidez egiten du (zerbitzariaren socketarekin lotzen da) zerbitzariaren prozesuak jatorrizko izendatutako zerbitzariaren socket-a mantentzen duen bitartean.

Komunikazio mota honetan oso ohikoa da haurren prozesua abiaraztea, behin konexioa eginda, bezeroaren prozesuarekin informazioa trukatzear arduratzen dena, zerbitzari nagusiak prozesuak konexioak onartzen jarraitzen duen bitartean.

Hurrengo diagramak socketen funtzionamendua modu sinplean erakusten du.



## JAVAKO SOCKETAK PROGRAMATZEA

Java-k Internet bezalako sareetarako programak eraikitzeke mekanismo indartsuak baina errazak eskaintzen ditu.

Zozketa serie bat erabiltzen dute komunikazio puntua finkatzeko, eskuragarri dagoen ataka jakin bateko urruneko makina batera konektatzeko, bertan entzuteko, bertan irakurtzeko edo idazteko eta argitaratzeko informazioa eta, azkenik, deskonektatzeko.

## SORTU ZERBITZARI SOCKETA

Java zerbitzari bat **sortzeko ServerSocket** klasea eman. Instantziatzerakoan portu zenbaki bat pasatzen den eraikitzaile bat erabiltzen da. Ataka zenbaki hau 1 eta 65535 arteko edozein zenbaki oso izan daiteke. Gogoratu zer ikusi zenuen klaseko portuei buruz. Ezin dira bi zerbitzari izan portu / zerbitzu bera zerbitzatzen dutenak ordenagailu berean.

```
ServerSocket socket = new ServerSocket (xxxxxx);
```

## BEZERO BAT ONARTU

Zerbitzaria sortu ondoren, **bezeroen konexioak zerbitzatzen hasteko** eskatuko zaio. Horretarako, **accept()** metodoa deitzen da. Metodo hau bezero batek konektatu arte itxarongo du, eta **Socket**-a itzuliko du, hau da, bezero horrekin duen konexioa.

```
Socket cliente = socket.accept();
```

Hainbat bezero onar daitezke aldi berean (ez dira adibidean ezarri)



## LORTU InputStream eta/edo OutputStream

Bezeroan konexioaren deskribatzailea duzu. Horretatik lortu behar dira **OutputStream** edo **InputStream** datuen sarrera eta irteerako kanalak **getOutputStream()** edo **getInputStream()** metodoekin. **OutputStream** klasea bezeroari datuak bidaltzeko erabiltzen da. **InputStream** klasea bezeroaren datuak irakurtzeko erabiltzen da.

```
InputStream entrada = cliente.getInputStream();  
OutputStream salida = cliente.getOutputStream();
```

**InputStream** eta / edo **OutputStream** bat sortzen da aplikazioaren beharren arabera.

Bi klase horien metodoek byte-ak soilik bidaltzen dituzte. Ohikoa da metodo egokiagoak dituzten beste zenbait datu sarrera / irteera klase eraikitzea:

- Mota normalak (zenbaki osoak, mugikorak, kateak) bidali edo jaso nahi badituzu, **DataInputStream** eta **DataOutputStream** klaseak dituzu. Klase hauek **InputStream** eta **OutputStream** onartzen dituen eraikitzailea dute hurrenez hurren.

```
DataInputStream entradaDatos = new DataInputStream (entrada);  
DataOutputStream salidaDatos = new DataOutputStream (salida);
```

- Klase osoak propioak bidali edo jaso nahi badituzu, **ObjectInputStream** eta **ObjectOutputStream** klaseak dituzu. Aurrekoetan bezala, **InputStream** eta **OutputStream** onartzen dituen eraikitzailea erabiltzen da

```
ObjectInputStream entradaObjetos = new ObjectInputStream (entrada);  
ObjectOutputStream salidaObjetos = new ObjectOutputStream (salida);
```

Klase berri hauek **readObject()** eta **writeObject()** erabiltzeko metodo intuitiboagoak dituzte.

## SOCKET-A IXTEN

Socket bat ixteko, deitu **close()** funtziora.

```
cliente.close(); // Con esto se cierra la conexión con el cliente .  
socket.close(); // Con esto se cierra el socket servidor, ya no atendemos más conexiones.
```

## BEZEROAREN SOCKETA SORTU

**Socket** klasea bezeroaren **socket** bat sortzeko erabiltzen da. **Nahikoa da zein makina konektatu (IP) eta konektatu behar duen ataka zein den adieraztea.** Zerbitzariak zerbitzatzen duen ataka berdina izan behar du.

```
Socket socket = new Socket ("localhost", 35557);
```

Gainerakoa zerbitzarian egiten den moduan egiten da.

### 3.- PRAKTIKA: XAT BAT JAVAN SOCKETEKIN

Praktikarekin batera **Client.txt** eta **Server.txt** bi testu fitxategi dituzu Java-n idatzitako txataren **Client** eta **Server** klaseekin.

1. Aztertu fitxategi hauek eta **bilatu aipatutako java klaseak**.
2. **1. galdera:** Egiaztatu eta kodean aurkitu bloke-diagramako urratsak jarraitzen direla socket-ak sortzeko.
3. **2. galdera:** Zer ataka erabiltzen du zerbitzariak? Zer ataka erabiltzen du bezeroak?
4. **3. galdera:** Eclipse IDE ingurunea erabiliz, konpilatu eta exekutatu klase hauek **zure ordenagailuan**. Zein exekutatu behar da lehen: zerbitzaria ala bezeroa?
5. **4. galdera:** Erabili **netstat -n** komandoa ezarritako konexioak ikusteko eta arazorik izanez gero. Aldatu programa beharrezkoa bada. Zer ataka zenbaki erabiltzen du gure zerbitzariak? Zer ataka zenbaki erabiltzen du gure bezeroak? Zein IP helbidetan dago gure zerbitzaria?
6. **5. galdera:** aldatu ikasgelan beste ordenagailu batekin konektatzeko beharrezkoa dena. Aztertu konexioa **netstat**-ekin. Azaldu egindako aldaketak eta horien zergatia. Zer ataka zenbaki erabiltzen du gure zerbitzariak? Zer ataka zenbaki erabiltzen du gure bezeroak? Zein IP helbidetan dago gure zerbitzaria? Eta bezeroa (Zerbitzarian begiratua)? Bezero bat baino gehiago konektatu ahal dira zerbitzari berberera?