```
public class Server {
private ServerSocket servsocket;
private int porto;
private Banco b;
public Server (int porto){
this.porto = porto;
this.b = new Banco();}
public void startServer(){
try {
this.servsocket = new ServerSocket(this.porto);
while(true){
Socket socket = servsocket.accept();
ServerWorker sw = new ServerWorker(socket, b);
new Thread(sw).start();}
} catch (IOException e) {
e.printStackTrace();}
}
```

```
public class ServerWorker implements Runnable {
private Socket socket;
private Banco b;
public ServerWorker (Socket socket, Banco bank){
this.socket = socket; this.b = bank;}
public void run() {
try{
ObjectInputStream in = new ObjectInputStream(socket.getInputStream());
ObjectOutputStream out = new ObjectOutputStream(socket.getOutputStream());
OpsBanco op;
while(true){
double saldo = 0; int id = 0; double valor = 0; op = (OpsBanco) in.readObject();
try{
switch(op){
case CRIAR_CONTA:
System.out.println(OpsBanco.CRIAR_CONTA)
saldo = in.readDouble(); id = b.criarConta(saldo);
out.writeInt(id); out.flush(); break;
case FECHAR_CONTA:
System.out.println(OpsBanco.FECHAR_CONTA);
id = in.readInt(); saldo = b.fecharConta(id);
out.writeInt(OpsBanco.OK.ordinal()); out.writeDouble(saldo); out.flush(); break;
default:
out.writeInt(OpsBanco.OP_INVALIDA.ordinal());
out.flush();}
} catch(ContaInvalida ci){
out.writeInt(OpsBanco.CONTA_INVALIDA.ordinal()); out.flush();}
}
} catch (IOException ioe) {
try{
socket.shutdownOutput(); socket.shutdownInput(); socket.close();}
catch(Exception e){
e.printStackTrace();    }
} catch (ClassNotFoundException ce){
ce.printStackTrace();  }
}
```