Soluzione Esercizio 2

```java
// PCStatement.java
public class PCStatement {
  public final int senderPort;
  public final String statement;
  public final double item;

  public PCStatement(int sp, String st, double it) {
    senderPort = sp; statement = st; item = it;
  }
}
```

```java
// PCList.java
import java.net.*;
import java.io.*;
import java.util.*;
import com.thoughtworks.xstream.*;
import java.sql.*;

 public class PCList {
    private static int receivePort, sendPort;

 public static void configure(int rp, int sp) {
    receivePort = rp; sendPort = sp;
  }

  public static void send(PCStatement o) {
    try (Socket sock = new Socket("localhost", sendPort);
      DataOutputStream dos =
        new DataOutputStream(sock.getOutputStream());
    ) {  dos.writeUTF((new XStream()).toXML(o));
    } catch (IOException e) { e.printStackTrace();}
    System.out.println("- invio a " + sendPort);
  }

  public static PCStatement receive() {
    PCStatement o = null;
    try (
      ServerSocket servsock = new ServerSocket(receivePort);
      Socket sock = servsock.accept();
      DataInputStream dis =
         new DataInputStream(sock.getInputStream());
    ) { o = (PCStatement)(new XStream()).fromXML(dis.readUTF());
    } catch (IOException e) { e.printStackTrace();}
    System.out.println("- ricevo");
    return o;
  }

  private static ArrayList read() {
```

```
38      try {
39        Connection co = DriverManager.getConnection("jdbc:mysql://localhost:3306/pclist",
"root","");
40        PreparedStatement ps = co.prepareStatement("SELECT xmlArrayList FROM pclist WHERE
receivePort = ?");
41        ps.setInt(1,receivePort);
42        ResultSet rs = ps.executeQuery();
43        if (rs.next()) {
44          String x = rs.getString("xmlarraylist");
45          return (ArrayList) (new XStream().fromXML(x));
46        }
47      } catch (SQLException e) { System.err.println(e.getMessage());}
48      return null;
49    }
50
51    private static void write(ArrayList a) {
52      String x = (new XStream()).toXML(a);
53      try (
54        Connection co = DriverManager.getConnection("jdbc:mysql://localhost:3306/pclist",
"root","");
55        PreparedStatement ps = co.prepareStatement("UPDATE pclist SET xmlarraylist = ? WHERE
receiveport = ?");
56      ) {
57        ps.setInt(2,receivePort); ps.setString(1, x);
58        System.out.println("rows affected: " + ps.executeUpdate());
59      } catch (SQLException e) { System.err.println(e.getMessage());}
60    }
61
62    public static void create(PCStatement o) {
63      write(new ArrayList());
64      System.out.println((new XStream()).toXML(o) + "\nCreato:\n" + read());
65      send(o);
66      if (o.senderPort == receivePort)
67        receive(); //08
68    }
69
70    public static void add(PCStatement o) {
71      ArrayList al = read();
72      int dimensioneDesiderata = 0;
73      while(al.size() > dimensioneDesiderata) { //01
74        send(o); //02
75        o = receive(); //03
76        dimensioneDesiderata++; //04
77      } //05
78      al.add(o.item);
79      write(al);
80      System.out.println((new XStream()).toXML(o) + "\nInserito:\n" + read());
81    }
82
83    public static void remove(PCStatement o) {
84      ArrayList al = read();
85      if (!al.remove(o.item)) { //06
86        send(o); //07
87        if (o.senderPort == receivePort)
88          receive(); //08
```

```
    }
    else {
      write(al); //09
      System.out.println((new XStream()).toXML(o) + "\nEstratto:\n" + read());
    }
  }

  public static void main(String[] args) {
    System.out.println("- sono " + args[0]);
    configure(Integer.parseInt(args[0]), Integer.parseInt(args[1]));
    PCStatement o;
    if (args.length > 2)  //10
      o = new PCStatement (Integer.parseInt(args[0]), args[2], Double.parseDouble(args[3]));
    else  //11
      o =  receive();
    switch(o.statement) {
      case "create": create(o); break;
      case "add": add(o); break;
      case "remove": remove(o); break;
    }
  }
}

/* Note
01 finchè non trova la dimensione desiderata
02 invia al successivo
03 attende nel caso non si trovi la dimensione desiderata
04 incrementa la dimensione desiderata
05 se e' stata trovata la dimensione desiderata l'operazione non sara' propagata
06 se il valore non viene estratto
07 invia al prossimo se giro non completo
08 se richiedente rimani in attesa dell'ultimo
09 se viene estratto archivia e stampa
10 e' il richiedente
11 non e' il richiedente
*/
```