Name_ Lastname_ Student number

Net Computing
academic year 2010-11

Rules

- This is a closed books exam.
- The operation of any electronic device is prohibited (e.g, no calculator, phone or PDA).
- Answer the questions being *precise, complete,* and *formal.*
- Write as *clearly* as possible, both in terms of handwriting and wording.
- You can answer the questions both in Dutch or in English, you choice.

Questions

1. Illustrate the general structure of a Remote Method Invocation architecture. Use Java RMI as an example for the explanation.
2. Illustrate the components of the name `http://portal.groningen.nl/`. Illustrate the differences between the iterative and the recursive name resolution and evaluate the two options.
3. Illustrate the phases of a process joining a Jini network and looking for a color printing service in the `student` domain.
4. Explain what are Jetty continuations as a form of async Web request processing. In particular, illustrate their use in the context of Web programming and provide typical scenarios.
5. Explain the Java code on the next page. If there are errors, please indicate them and explain how to correct them.

```
import java.io.*;
import java.net.*;
public class A{ 1
   ServerSocket s;
   Socket c = null;
   ObjectOutputStream o;
   ObjectInputStream i;
String m;
   Provider(){}
   void run()
  2{
      try{ 3
         s = new ServerSocket(2004, 10);
         c = s.accept();
         System.out.println(c.getInetAddress().getHostName());
         o = new ObjectOutputStream(c.getOutputStream());
         o.flush();
         i = new ObjectInputStream(c.getInputStream());
         sendMessage("Bleah");
         do{ 4
            try{ 5
               m = (String)i.readObject();
               System.out.println(m);
               if (m.equals("XXX"))
                  sendMessage("no no");
            }5
            catch(IOException ioException){
               ioException.printStackTrace();
          5}while(!m.equals("XXX"));
         }3
         catch(IOException ioException){3
            ioException.printStackTrace();
      }3
    2}
     void sendMessage(String msg)
     {2
        try{3
           o.writeObject(msg);
           o.flush();
        }3
        catch(IOException ioException){
           ioException.printStackTrace();
        }3
  }2
  public static void main(String args[])
  {2
     A ss = new A();
     while(true){ 3
        ss.run();
     } 3
  }2
}1
```