→ Sockets em Java

1 – O programa abaixo permite-lhe obter o Nome da máquina onde está a trabalhar. Implemente-o e explore a classe InetAddress.

```
import java.net.*;

public class GetName {
  public static void main (String args[]) throws Exception{
    InetAddress host = null;
    host = InetAddress.getLocalHost();
    System.out.println(host.getHostName());

  }
}
```

2 – O programa abaixo permite-lhe obter o endereço IP da máquina onde está a trabalhar. Implemente-o e estude o seu comportamento.

```
import java.net.*;

public class GetIP {
  public static void main (String args[]) throws Exception{
    InetAddress host = null;
    host = InetAddress.getLocalHost();
    byte ip [] = host.getAddress();
    for ( int i= 0 ; i < ip.length; i++){
      if (i>0) System.out.print(".");
      System.out.print(ip[i] & 0xff);
    }
    System.out.println();
  }
}
```

3 – Pretende-se que o programa abaixo, dado um determinado IP, nos diga qual o nome da máquina correspondente. Complete-o.

```
import java.net.*;
import java.io.*;
```

```
public class IPtoName {
 public static void main (String args[] ) throws IOException{
          String s=" ";
          char c;
       System.out.print("IP address? ");
          while ( (c=(char)System.in.read())!= 10)
                    s+=c;
          s=s.trim();

       InetAddress address =null;
       try {

        < exercício >

      }
       catch (UnknownHostException e){
         System.out.println("IP malformed ");
         e.getMessage();
       }
      }
}
```

4 – Construa um programa que dado o nome de uma máquina, nos diga qual o IP correspondente.

5 – Estude e implemente a classe abaixo.

```
import java.net.*;
import java.io.*;

public class Cliente {
  public Cliente(){
  try {
    Socket sc = new Socket("127.0.0.1", 2222);
    InputStream is = sc.getInputStream();
    BufferedReader br = new BufferedReader (new InputStreamReader(is));
    PrintWriter pr =new PrintWriter ( sc.getOutputStream() , true);

    System.out.println(br.readLine());
    pr.println(" Olá, eu sou o cliente");
```

```
    System.out.println(br.readLine());
    pr.println("Cliente: Continuo por aqui");

    pr.close();
    is.close();
    sc.close();
   }
   catch (IOException e){
    System.out.println( e.getMessage());
   }
   }
   public static void main (String args[]){
     Cliente c=new Cliente();
   }
}
```

6 – Implemente uma classe, Servidor, que comunique com o Cliente do exercício anterior.

7 – Depois de executar o cliente e o servidor anteriores na mesma máquina, teste o seu processo Cliente com o processo Servidor de um dos seus colegas e vice-versa.

8 – Modifique o Cliente e o Servidor dos exercícios 5 e 6 de forma a ambos efectuarem primeiro as operações de escrita no socket e depois as operações de leitura. O que acontece?

9 – Modifique o Cliente e o Servidor dos exercícios 5 e 6 de forma a ambos efectuarem primeiro as operações de leitura do socket e depois as operações de escrita. O que acontece?

10 – Pretende-se construir uma aplicação cliente/servidor cuja comunicação é feita através de Sockets. O processo Servidor recebe um valor do tipo char que identifica o tipo de cliente que está a comunicar com ele.
Clientes do tipo ‘A’, depois de enviarem ao servidor o seu tipo, enviam um valor inteiro e recebem como resposta o quadrado desse valor.
Clientes do tipo ‘B’, depois de enviarem ao servidor o seu tipo, enviam um valor do tipo double e recebem como resposta a raiz quadrada desse valor.

- Construa as classes Servidor, Cliente_A e Cliente_B.

Notas:

Para gerar os valores a enviar pelos clientes use o método "double Math.random()", para calcular a raiz quadrada use o método "double Math.sqrt(double)".
Para criar uma input stream:
DataInputStream is = new DataInputStream(<input stream do socket>);
Para ler da input stream use os métodos:
DataOutputStream os = new DataOutputStream(<output stream do socket>);
Para criar uma output stream:
DataOutputStream os = new DataOutputStream(<output stream do socket>);
Para escrever na output stream use os métodos:
"void writeInt(int);", "void writeChar(char);"e "void writeDouble(double);"

11 - Pretende-se construir um servidor que a cada ligação de um cliente através de um Socket responda com uma saudação aleatória. O servidor deverá possuir um array de Strings com diferentes saudações, por exemplo {"Bom dia", "Bem disposto?", "Oi", "Salvé", ... }.
Sempre que um processo cliente envia uma mensagem ao servidor (independentemente do conteúdo dessa mensagem) o servidor responderá com uma das saudações do array escolhida aleatoriamente. O servidor deve possuir um ciclo infinito onde aceita a ligação com o cliente, recebe a mensagem daquele e envia a sua saudação aleatória.

- Construa as classes cliente e servidor.

Notas:
Para criar uma input stream:
BufferedReader is = new BufferedReader (new InputStreamReader(<input stream do socket>));
Para ler da input stream use o método: String readLine()
Para criar uma output stream:
PrintWriter os = new PrintWriter (<output stream do socket>, true);
Para escrever na output stream use o método: void println(String);

12 – Construa uma aplicação cliente / servidor em que o servidor receba do cliente dois arrays de inteiros, calcule a sua soma e devolva o resultado ao cliente.