**CC10A-Computación I – EXAMEN – viernes 26 de noviembre de 2004**
**Tpo: 2 hrs 45 minutos – CON apuntes – SIN Consultas – Contestar en hojas separadas**

1. Una relación es un conjunto de pares ordenados, por ejemplo, R={(1,2),(2,3),(1,4),(3,3),...}. Al respecto, se dispone de la clase Relacion con los siguientes métodos:

| **encabezamiento** | **objetivo** |
|---|---|
| Relacion() | constructor que inicializa como vacío un objeto de la clase Relacion |
| boolean agregarPar(int x,int y) | agrega par (x,y) a la relación. Devuelve false si el par ya existía. |
| int[] ordenadas(int x) | Entrega un arreglo con todos los y tales que existe un par(x,y) |
| int[] abcisas(int y) | Entrega un arreglo con todos los x tales que existe un par(x,y) |

**a) (2 ptos)** Escriba el método **boolean esFuncion(Relacion R,int a, int b)** que entregue true si R representa una función en el intervalo [a,b], es decir, si para cada valor entero x entre a y b, existe un sólo par (x,y).

```
static public boolean esFuncion(Relacion R,int a,int b){
for(int i=a; i<=b; ++i){      //0.5
  int[]y=R.ordenadas(i);      //0.5
  if(y.length!=1)             //0.5
    return false;             //0.25
}
return true;                  //0.25
}
```

**b) (2 ptos)** Escriba el método **ordenadas** suponiendo que la representación de un objeto es: boolean[][]par=new boolean[100][100]; //par[i][j]==true si existe par (i,j) en la relación

```
public int[] ordenadas(int x){
//contar ordenadas: 1.0
  int n=0;                    //0.1
  for(int i=0; i<100; ++i)    //0.3
    if( par[x][i] )           //0.4
      ++n;                    //0.2
//entregar ordenadas en arreglo: 1.0
  int[]y = new int[n];        //0.1
  n=0;                        //0.1
  for(int i=0; i<100; ++i)    //0.2
    if( par[x][i] )           //0.2
      y[n++]=i;               //0.3
  return y;                   //0.1
}
```

**c) (2 ptos)** Escriba el método **agregarPar** suponiendo que la representación de un objeto es a través de una lista enlazada de nodos de la clase: class Nodo{int x,y; Nodo sgte;}

```
public boolean agregarPar(int x,int y){
//buscar en lista enlazada: 1.0
for(Nodo r=primero; r!=null; r=r.sgte)    //0.5
  if( r.x==x && r.y==y )                  //0.4
    return false;                         //0.1
//agregar a lista enlazada: 1.0
Nodo r = new Nodo();    //0.2
r.sgte=primero;         //0.2
r.x=x; r.y=y;           //0.2
primero=r;              //0.2
return true;            //0.2
}
```

**2.** Para apoyar la próxima Teletón, en que la gente deposita dinero en la cuenta 24500-03 en diferentes sucursales bancarias ubicadas en distintas ciudades de las regiones del país, se ha diseñado una base datos compuesta por las siguientes tablas:

| Tabla | Columna | Tipo | Ejemplo |
|---|---|---|---|
| **Cuentas** | número | 10 caracteres | 24500-03 |
| | titular | 20 caracteres | Teletón |
| | saldoTotal | entero (10 digitos) | 0 |
| **Sucursales** | código | 5 caracteres | 12345 |
| | ciudad | 20 caracteres | Santiago |
| **Depósitos** | codigoSucursal | 5 caracteres | 12345 |
| | numeroCuenta | 10 caracteres | 24500-03 |
| | dinero | entero (10 digitos) | 80000 |

Escriba un programa que pueda ser consultado en cualquier momento durante la Teletón para saber la cantidad de dinero recolectada en una ciudad y en el país, de acuerdo a la siguiente interfaz:

| | | | |
|---|---|---|---|
| Label | Nombre ciudad? | x...x | TextField |
| Label o TextField | Total ciudad=$Nº | Total país=$Nº (x%de la meta) | Label o TextField |

```
//definición de frame: 1.5 ptos
class Pregunta2 extends Frame implements ActionListener{          //0.1
static public void main(String[]args){new Pregunta2().show();}    //0.2
private Label preg=newLabel("Nombre ciudad?"); //0.1
private TextField ciudad = new TextField();    //0.1
private Label totalCiudad = new Label();       //0.1
private Label totalPais = new Label();         //0.1
public Pregunta2(){                            //0.1
  setLayout( new GridLayout(2,2));             //0.1
  add(preg); add(ciudad);                      //0.2
  add(totalCiudad); add(totalPais);            //0.2
  ciudad.addActionListener(this);              //0.2
}
//obtener sucursales de una ciudad: 1.5 ptos
public void actionPerformed(ActionEvent x){                       //0.1
Connection con=DriverManager.getConnection("url","login","pswd");//0.1
Statement st=con.createStatement();                               //0.1
String nombre = ciudad.getText();                                 //0.1
ResultSet r1=st.executeQuery(                                     //0.2
"select codigo from Sucursales where ciudad='"+nombre+"'");       //0.5
int total = 0;
while(r1.next()){                                                 //0.2
  String codigo=r1.getString("codigo");                           //0.2
  //obtener dinero de sucursales de ciudad: 1.5 ptos
  ResultSet r2=st.executeQuery(                                          //0.2
  "select dinero from Depositos where numeroCuenta=´24500-03´" +         //0.1
  "and codigoSucursal='"+codigo+"'");                                    //0.3
  while(r2.next()){                                                      //0.2
    int dinero=Integer.parseInt( r2.getString("dinero"));                //0.3
    total += dinero;                                                     //0.2
  }
}
totalCiudad.setText("Total ciudad=$" + total);                           //0.2
//mostrar total nacional: 1.5 ptos
ResultSet r3=st.executeQuery(                                     //0.2
"select saldoTotal from Cuentas where numero='24500-03");         //0.3
r3.next();                                                        //0.1
int saldoTotal=Integer.parseInt( r3.getString("saldoTotal"));     //0.5
totalPais.setText(                                                //0.1
"Total pais=$" + saldoTotal +                                     //0.1
"("+100.0*saldoTotal/2000000000+"% de la meta)");                 //0.2
}
}
```

**Solución alternativa: con clases ad-hoc**

**//definición de frame: 1.5 ptos**

```
class Pregunta2 extends Frame implements ActionListener{          //0.1
static public void main(String[]args){new Pregunta2().show();}    //0.2
private Label preg=newLabel("Nombre ciudad?"); //0.1
private TextField ciudad = new TextField();    //0.1
private Label totalCiudad = new Label();       //0.1
private Label totalPais = new Label();         //0.1
public Pregunta2(){                            //0.1
  setLayout( new GridLayout(2,2));             //0.1
  add(preg); add(ciudad);                      //0.2
  add(totalCiudad); add(totalPais);            //0.2
  ciudad.addActionListener(this);              //0.2
}
```

**//obtener sucursales de una ciudad: 1.5 ptos**

```
public void actionPerformed(ActionEvent x){                       //0.1
String nombre = ciudad.getText();                                 //0.1
int total = 0;                                                    //0.1
RandomAccessFile Sucursales=new RandomAcessFile("Sucursales","R");//0.3
Sucursal sucursal=new Sucursal();                                 //0.3
while(sucursal.leer(Sucursales)){                                 //0.3
  if( !nombre.equals(sucursal.codigo) ) continue;                 //0.3
```

**//obtener dinero de sucursales de ciudad: 1.5 ptos**

```
  RandomAccessFile Depositos=new RandomAcessFile("Depositos","R");//0.2
  Deposito deposito=new Deposito();                               //0.1
  while(deposito.leer(Depositos)){                                //0.2
    if( deposito.numeroCuenta.equals("24500-0)==false             //0.2
    || deposito.codigoSucursal.equals(codigo)==false)continue;    //0.3
    total += deposito.dinero;                                     //0.3
  }
}
totalCiudad.setText("Total ciudad=$" + total);                    //0.2
```

**//mostrar total nacional: 1.5 ptos**

```
RandomAccessFile Cuentas=new RandomAcessFile("Cuentas","R");      //0.2
Cuenta cuenta=new Cuenta();                                       //0.1
while( cuenta.leer(Cuentas)){                                     //0.2
  if(cuenta.numero.equals("24500-03")){                           //0.3
    int saldoTotal=cuenta.saldoTotal;                             //0.2
    totalPais.setText(                                            //0.1
    "Total pais=$" + saldoTotal +                                 //0.1
    "("+100.0*saldoTotal/2000000000+"% de la meta)");             //0.2
    break;                                                        //0.1
  }
}
}
```

**3.**Es muy habitual llamar a la empresa telefónica para preguntar por el número de una persona. Para facilitar esta labor escriba un programa servidor que, a través del port 9999, reciba una línea de texto con el nombre de una persona y entregue otra línea de texto con el número de teléfono (o el mensaje "no existe").

**Notas**.

- El programa debe atender concurrentemente a múltiples clientes
- El programa dispone del archivo de texto ("nombres.txt") que en cada línea contiene el nombre y el teléfono separados por "**:**" (por ejemplo, "Juan Pérez**:**1234567").
- El archivo está ordenado por nombre y cada nombre aparece una sola vez

```
//programa principal: 1.5
static public void main(String[]args)throws Exception{
ServerSocket ss=new ServerSocket(9999);  //0.5
while(true){                             //0.2
  Socket s= ss.accept();                 //0.3
  T t=new T(s);                          //0.3
  t.start();                             //0.2
}
//definicion de thread: 0.5
class T extends Thread{       //0.1
private Socket s;             //0.1
public T(Socket x){ s=x; }    //0.2
public void run(){            //0.1
  //obtener nombre de persona: 1.0
  BufferedReader br = new BufferedReader(      //0.2
    new InputStreamReader(s.getInputStream()));//0.2
  String nombre = br.readLine();               //0.5
  br.close();                                  //0.1

  //leer archivo hasta encontrar nombre: 1.5
  int i=0, c=0;                                //0.1
  BufferedReader a=new BufferedReader(new FileReader("nombres.txt"));//0.1
  String linea;                                //0.1
  while((linea=a.readLine()) != null){         //0.2
    i=linea.indexOf(":");                      //0.3
    c = nombre.compareTo(linea.substring(0,i));//0.5
    if( c<=0 ) break;                          //0.2
  }

  //enviar respuesta a programa cliente: 1.5
  PrintWriter pw= new PrintWriter(       //0.1
    s.getOutputStream(),true);           //0.1
  if( linea==null || c<0 )               //0.5
    pw.println("no existe");             //0.3
  else
    pw.println(linea.substring(i+1));    //0.5
  pw.close();
  s.close();
}
}
```