Enunciado (enunciado FÁCIL):

Se tiene un árbol binario en donde se guardan los nombres de los empleados de una empresa en orden alfabético. Cada nodo 'Empleado' tiene un puntero extra a su jefe inmediato. Ningún empleado tiene más de un jefe, y pueden existir empleados que no tengan ningún superior. Se tiene además de un Archivo, donde cada registro contiene el nombre de un empleado y el nombre de su jefe.

Declarar las estructuras y realizar los siguientes algoritmos:

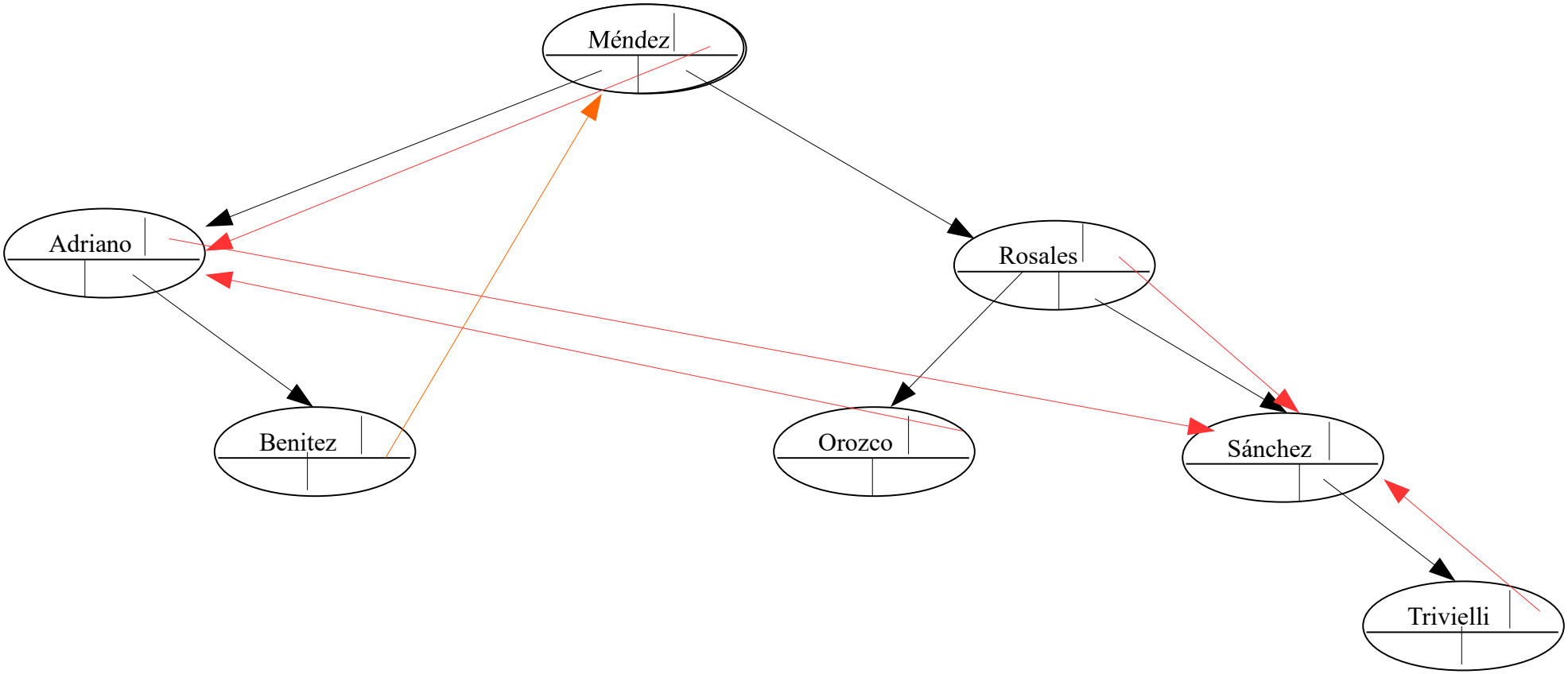
Realice un procedimiento/función que agregue a la estructura dada, los punteros a los jefes indicados por el archivo.

Realice un procedimiento/función que dado el nombre de un empleado imprima su jefe y sus subordinados.

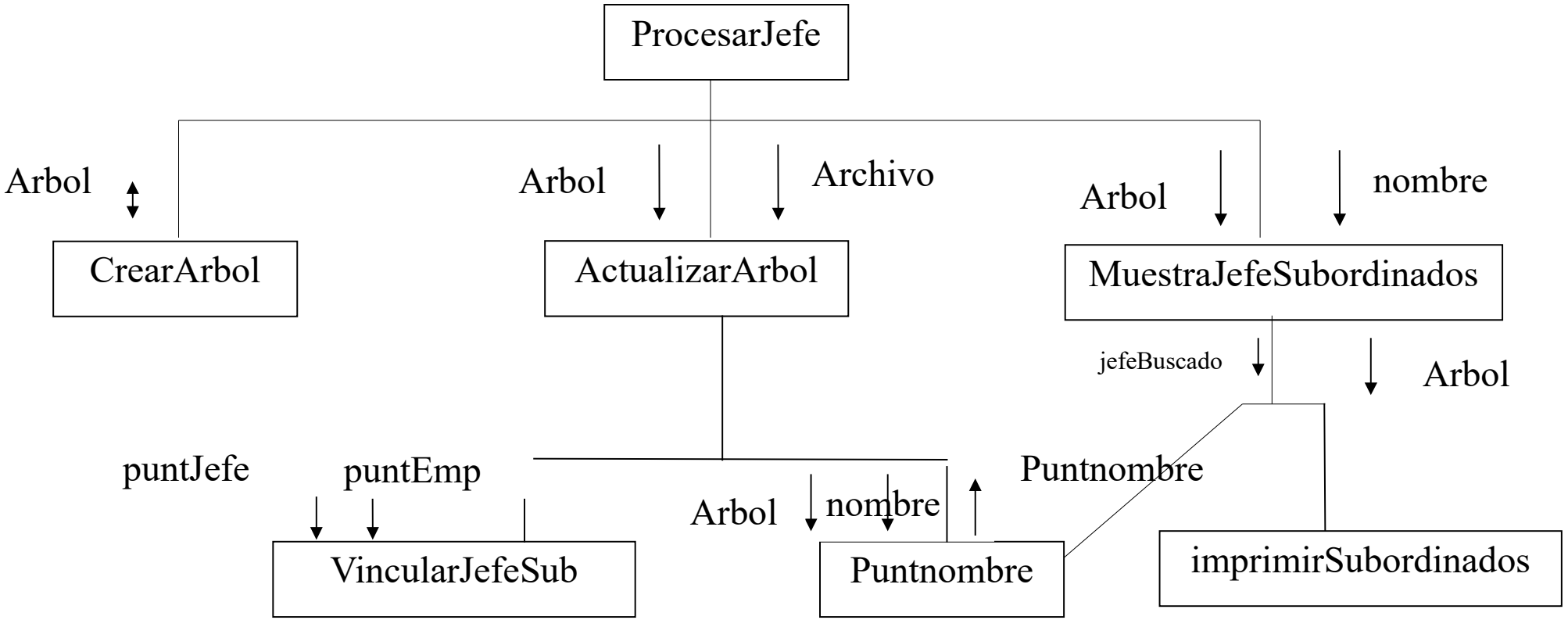
- **Nota:**

  El árbol existe y puede tener algunos jefes cargados, si es así los datos del árbol tienen mayor prioridad. El archivo existe y tiene los datos cargados. No debe utilizar estructuras auxiliares. La inexistencia de algunos de los nombres del archivo en el árbol debe ser informada mediante un mensaje y no dar de alta nodos en el árbol.

- EJEMPLO



En este ejemplo, Sánchez no tiene Jefe, y sus subordinados son Rosales, Trivelli y Adriano. A su vez Méndez es jefe de Benitez, y Adriano es jefe de Orozco y Méndez


ProcesarJefe
Arbol
Arbol
Archivo
Arbol
nombre
CrearArbol
ActualizarArbol
MuestraJefeSubordinados
jefeBuscado
Arbol
puntJefe
puntEmp
Arbol
nombre
Puntnombre
VincularJefeSub
Puntnombre
imprimirSubordinados

```
 Program EjemploArbol;
type
      prtArbol = ^ TArbol;
      TArbol = record
          nombre: string;
          jefe:   prtArbol;
          izq, der: prtArbol;
      end;
TArchivo = record
          nombre: string;
          jefe: string;
          end;
ArchivoEmpleado= File of TArchivo;
```

```
.....
var
            arbol:  prtArbol;
            Archivo:   File of TArchivo;
            nombre: string;
 begin
 assign (Archivo,  '/ip2/carleo-datosJefes')
 {arbol:= nil;
  writeln( 'creo datos del arbol');
  cearArbol(arbol);}
  readln(nombre);
  ActualizarArbol( arbol, Archivo) ;
  MuestraJefeSubordinados(arbol, nombre);
  end.
```

```
procedure ActualizarArbol( var arbol:prtArbol ; var Archivo:ArchivoEmpleado);
    var
            datos:  Tarchivo;
              puntEmp, puntJefe: prtArbol;
   begin
        Reset (Archivo);
        While (not eof (Archivo)) do
           begin
           Read (Archivo, datos );
           puntEmp:= punteroANombre(arbol, datos.nombre);
                 If puntEmp <> nil
                       then begin
                                puntJefe:= punteroANombre(arbol, datos.jefe);
                                if (puntJefe<> nil )
                                        then VincularJefeSub(puntEmp,puntJefe)
                                       else Writeln ('No existe El jefe en el Arbol');
                     else Writeln ('No existe Empleado el Arbol');
           end;
        close (Archivo);
        end;
```

```
function PunteroAnombre(arbol: prtArbol; nombre:string):  prtArbol;
    {dado un nombre de empleado lo busca en el Arbol y devuelve el puntero si existe, c
contrario, devuelve nil}
begin
        if (arbol = nil) then
            PunteroAnombre:= nil
        else
          if (arbol^.nombre   =  nombre) then
              PunteroAnombre:= arbol
          else
              if    (arbol^.nombre   >  nombre ) then
                  PunteroAnombre:=PunteroAnombre(arbol^.izq, nombre)
              else
                  PunteroAnombre:=PunteroAnombre(arbol^.der, nombre);
end;

procedure VincularJefeSub (puntEmp: prtArbol; puntJefe:prtArbol);
   { Recibe los dos punteros a vincular. Informa si ya tiene jefe}
        begin
            if puntEmp^.jefe = nil  then
                puntEmp^.jefe:= puntJefe
            else
                writeln('Ya tiene jefe asignado.');
        end;
```

```
Procedure MuestraJefeSubordinados(arbol: prtArbol ; nombreEmpleado:
string );
 {busca en el árbol el nombre del empleado, si esta imprime el jefe si lo tiene y
luego invoca a un módulo para imprimir los subordinados}
 var
            ptrEmpleado: prtArbol;
  begin
     ptrEmpleado:=  PunteroAnombre(arbol, nombreEmpleado);
     if ptrEmpleado <> nil
       then begin
            if   (ptrEmpleado^.jefe <> nil ) then
                writeln('el empleado ', nombreempleado, 'tiene como jefe a ' ,
                                                        ptrEmpleado^.jefe^.nombre)
            else
                 writeln ('no tiene jefe');
            imprimeSubordinado(arbol, nombreempleado);
        end
        else
            writeln ('empleado no existe')
 end;
```

```
Procedure imprimeSubordinado(arbol:prtArbol ; jefeBuscado:string);
{dado el nombre del empleado  ´jefe´,  busca de manera inorder en  el arbo
que empleados  son subordinados de él}

 begin
    if arbol <>   then  begin
          imprimeSubordinado(arbol^.izq, jefeBuscado);
          if (arbol^.jefe <> nil)  and (arbol^.jefe^.nombre = efeBuscado)  then
                          writeln ( 'el subordinado es  ' , arbol^.nombre);
          imprimeSubordinado(arbol^.der,jefeBuscado);
     end;
 end;
```