

## Introducción a la Programación II – Ejemplo eliminar un nodo de un árbol

```
Program E6P6;  
{carga un arbol y despues elimina un numero ingresado por el usuario si es que esta}
```

```
Type puntarbol=^Tarbol;  
Tarbol= record  
nro:integer;  
menores:puntarbol;  
mayores:puntarbol;  
end;
```

```
Procedure CargarValor(var Arbol:puntarbol; valor:integer);  
{carga el valor que recibe en el arbol}
```

```
begin  
if (arbol=NIL) then  
begin  
new(Arbol);  
Arbol^.nro:=valor;  
Arbol^.menores:=NIL;  
Arbol^.mayores:=NIL;  
end  
else  
if (Arbol^.nro<valor) then  
CargarValor(Arbol^.mayores,valor);  
if (Arbol^.nro>valor) then  
CargarValor(Arbol^.menores,valor);  
end;
```

```
Procedure IngresarValores(var Arbolito:puntarbol);  
{pide al usuario que ingrese los valores que va a ir ingresando en el arbol}
```

```
var numero:integer;  
begin  
Arbolito:=NIL;  
numero:=-1;  
writeln('Ingrese Numeros Positivos Para Que Sean Cargados En El Arbol,  
Si No Quiere Seguir Cargando, Ingrese Un Valor Negativo');  
read(numero);  
while (numero>=0) do  
begin  
CargarValor(Arbolito,numero);  
read(numero);  
end;  
end;
```

```
Procedure Eliminar(var Arbol:puntarbol);  
{recibe el puntero que apunta al valor que quiero eliminar}  
var AEliminar:puntarbol;  
begin  
  AEliminar:=Arbol;  
  Arbol^.menores:=NIL;  
  Arbol^.mayores:=NIL;  
  arbol:=NIL;  
  dispose(AEliminar);  
end;
```

  

```
Procedure buscarElMayor(var listamenores:puntarbol);  
{devuelve el puntero al mayor de los menores}  
var auxiliar:puntarbol;  
begin  
  if (listamenores^.mayores<>NIL) then  
    begin  
      while (listamenores^.mayores^.mayores<>NIL) do  
        listamenores:=listamenores^.mayores;  
        auxiliar:=listamenores^.mayores;  
        listamenores^.mayores:=auxiliar^.menores;  
        listamenores:=auxiliar;  
        auxiliar^.mayores:=listamenores^.menores;  
      end;  
    end;
```

  

```
Procedure intercambiar(var arbol:puntarbol);  
{intercambia el puntero a eliminar con el mayor de los menores}  
var nodoeliminar,nodointercambiar:puntarbol;  
begin  
  nodoeliminar:=arbol;  
  if (arbol^.menores=NIL) then  
    arbol:=arbol^.mayores  
  else  
    begin  
      nodointercambiar:=arbol^.menores;  
      buscarelmayor(nodointercambiar);  
      if (nodointercambiar<>arbol^.menores) then  
        nodointercambiar^.menores:=arbol^.menores;  
        nodointercambiar^.mayores:=arbol^.mayores;  
        arbol:=nodointercambiar;  
      end;  
  Eliminar(nodoeliminar);  
end;
```

```
Procedure buscarposicionaeliminar(var arbol:puntarbol; valor:integer);  
{busca la posicion del que quiero eliminar}  
begin  
  if (arbol^.nro=valor) then  
    begin  
      if (arbol^.mayores=NIL) and (arbol^.menores=NIL) then  
        Eliminar(arbol)  
      else  
        intercambiar(arbol);  
      end  
    else  
      if (arbol^.nro<valor) then  
        buscarposicionaeliminar(arbol^.mayores,valor)  
      else  
        if (arbol^.nro>valor) then  
          buscarposicionaeliminar(arbol^.menores,valor);  
        end;  
    end;
```

```
Procedure DarDeBajaNodo(var Arbol:puntarbol);  
{se encarga de eliminar del arbol un numero que ingresa el usuario si es  
que se encuentra}  
var nroAeliminar:integer;  
begin  
  nroAeliminar:=-1;  
  writeln('Ingrese Un Numero a Eliminar (que este en el arbol, sino  
explota todo)');  
  readln(nroAeliminar);  
  buscarposicionaeliminar(arbol,nroAeliminar);  
end;
```

```
Procedure ListarAscendentemente(Arbol:puntarbol);  
{Usa recursion para recorrer el arbol y mostrar el contenido de menor a  
mayor}  
Begin  
  If (Arbol^.menores<>NIL) Then  
    ListarAscendentemente(Arbol^.menores);  
    WriteLn(Arbol^.nro);  
  If (Arbol^.mayores<>NIL) Then  
    ListarAscendentemente(Arbol^.mayores);  
end;
```

```
var ArbolPrincipal:puntarbol;  
begin  
  IngresarValores(ArbolPrincipal);  
  writeln('Arbol Completo');  
  ListarAscendentemente(ArbolPrincipal);  
  DarDeBajaNodo(ArbolPrincipal);  
  writeln('Arbol Despues De La Eliminacion');  
  if ArbolPrincipal<>NIL then  
    ListarAscendentemente(ArbolPrincipal)  
  else  
    writeln('El Arbol No Tiene Ningun Elemento');  
end.
```