EJEMPLO 6.1

```
Type
  Nodo = record
    Elemento: tipo_dato_elemento;
    Hijo_Izquierdo: integer;
    Hijo_Derecho:integer;
End;
Archivo: file of Nodo;

Procedure Insertar (var A: Archivo, elem:
                    tipo_dato_elemento)
Var
  Raiz, nodo_nuevo: Nodo;
  Pos_nuevo_nodo:   integer;
  Encontre_Padre:   boolean;

Begin
Reset(A);
With nodo_nuevo do
  Elemento   :=elem;
  Hijo_izquierdo    := -1;
  Hijo_Derecho      := -1;
End;

If Eof(A) Then {significa que es un árbol vacío y el
elemento es insertado como raíz}
  Write(A,nodo_nuevo);
Else
  Read(A, Raiz);
  Pos_nuevo_nodo := filesize(A);
  Seek(A, pos_nuevo_nodo); {posicionarse al final del
  archivo}
  Write(A, nodo_nuevo); {escribir el nuevo nodo al final}
  Encontre_Padre := false;

  {buscar al padre para agregar la referencia al nuevo
  nodo}
  While not (Encontre_Padre) do
  begin
      If (Raiz.elemento > nodo_nuevo.elemento) Then
        If (Raiz.hijo_izquierdo <> -1) Then
           Seek(A, Raiz.hijo_izquierdo);
           Read(A, Raiz);
        Else
           Raiz.hijo_izquierdo := Pos_nuevo_nodo;
           Encontre Padre:= true;

      Else
        If (Raiz.hijo_derecho <> -1) Then
           Seek(A, Raiz.hijo_derecho);
           Read(A, Raiz);
        Else
           Raiz.hijo_derecho := Pos_nuevo_nodo;
           Encontre_Padre := true;
        end;
      end;
  end;
  {raíz es el padre y ya lo leí, debo volver a
  posicionarme}
  Seek(A, Filepos(A)-1);

  {guardo al padre con la nueva referencia}
  Write (A, raiz);
end.
```

EJEMPLO 6.2

```
Type
  Nodo = record
      Elemento: tipo_dato_elemento;
      Hijo_Izquierdo: integer;
      Hijo_Derecho:integer;
End;
Archivo : file of Nodo;

Function Buscar (var A:archivo, elem:tipo_dato_elemento):
integer
{retorna el NRR del nodo; si el árbol es vacío, retorna
−1}

Var
  Raiz: nodo;
  Encontre:boolean;
  Pos:integer;

Begin

  Reset(A);
  Encontre := False;
  Pos      := −1;

  If not eof(A) Then
    begin
        Read(A, Raiz);

        While not (encontre) and not eof(A) do
              If(raiz.elemento > elem) then
                Pos := raiz.hijo_izquierdo;
                Seek(pos);
                Read(A, Raiz);
              Else
                If (Raiz.elemento < elem) then
                    Pos:= raiz.hijo_derecho;
                    Seek(pos);
                    Read(A, Raiz);
                Else
                    Encontre := true;

    End;

    {pos tiene por defecto −1, sino el NRR del nodo donde
    está el elemento buscado}
    Buscar := pos;
  End
```