Taller de Programación

AGENDA

Estructura de datos arbol de búsqueda

Operaciones – CREACION – CARGA DE DATOS

ÁRBOLES BINARIOS DE BÚSQUEDA- CREACION

```
Programa arboles;
Type
  arbol = ^nodo;
  nodo = record
           dato: integer;
           HI: arbol;
           HD: arbol;
         end;

Var
 a:arbol;
 num:integer;
Begin
 ….
End.
```

```
Begin
  a:= nil; //indico que el árbol está vacío

   read (num); //leo un valor
   while (num <> 50) do

     begin
       agregar (a,num); //agrego el valor al arbol
       read (num);
     end;
End..
```

Suponga que se leen los siguientes valores y se quieren ir agregando en a (9, 18, 22, 19, 7,50). Cómo quedarán guardados?

ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

Suponga que se leen los siguientes valores y se quieren ir agregando en un ABB (9, 18, 22, 19,7, 50)

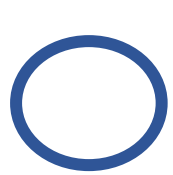
```
Programa arboles;
Type
  arbol = ^nodo;
  nodo = record
           dato: integer;
           HI: arbol;
           HD: arbol;
         end;
Var
 a:arbol;  num:integer;
Begin
  a:= nil;
 read (num);
  while (num <> 50) do
   begin
    agregar (a,num);
    read (num);
   end;
End.
```

Cómo quedarán guardados los valores en el ABB?

a = nil

Se **lee el valor 9 (num)** y se invoca al procedimiento agregar (a,num)

Como a = nil, el primer valor leído será la raíz del arbol.
Para agregarlo al ser una estructura dinámica debe reservarse memoria.



Luego se asigna en el campo dato el valor de num, y como por ahora este nodo no tiene hijos, en los campos HI e HD debe asignarse nil

El procedimiento agregar termina y vuelve al programa principal en donde **a** ahora apunta a un nodo con valor 9 y sus hijos en nil.

ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

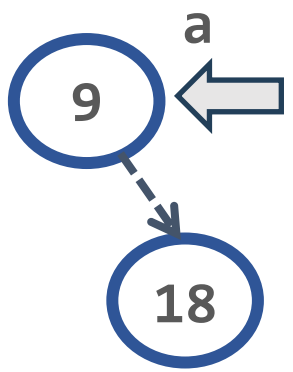
Suponga que se leen los siguientes valores y se quieren ir agregando en un ABB (9, 18, 22, 19, 7, 50)

```
Programa arboles;
Type
  arbol = ^nodo;
  nodo = record
          dato: integer;
          HI: arbol;
          HD: arbol;
         end;
Var
 a:arbol;  num:integer;
Begin
  a:= nil;
 read (num);
  while (num <> 50) do
   begin
    agregar (a,num);
    read (num);
   end;
End.
```

a = 9

Se **lee el valor 18 (num)** y se invoca al procedimiento agregar (a,num)

Como el árbol NO es vacío, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja.



Comparo num (18) con lo que está apuntado a (9), como 18 > 9 se determina que hay que agregarlo a la derecha de 9.

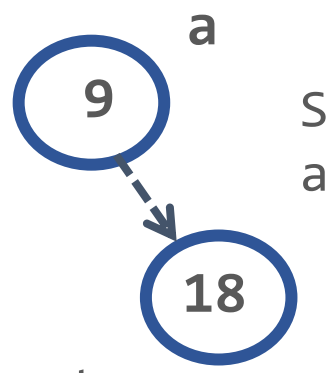
Como HD de 9 es =nil, ya se encontró el lugar (hoja), reservo memoria dinámica y asigno los valores correspondientes

El procedimiento agregar termina y vuelve al programa principal en donde a ahora apunta **a** un nodo con valor 9 y su HI= nil y **HD = 18**.

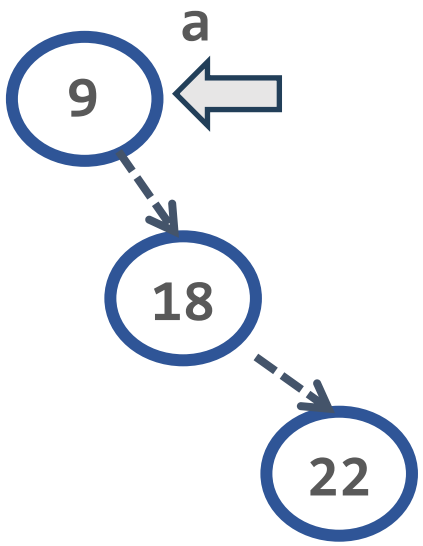
ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

Suponga que se leen los siguientes valores y se quieren ir agregando en un ABB (9, 18, 22, 19, 7, 50)

a = 9



Se **lee el valor 22 (num)** y se invoca al procedimiento agregar (a,num)



Como el árbol a <> nil, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja.

Comparo num (22) con lo que está apuntado a (9), 22> 9 se determina que hay que agregarlo a la derecha de 9. Como HD de 9 <> nil, sigo recorriendo hacia la derecha. Luego se compara y 22 > 18 y como HD 18 = nil se encontró el lugar donde agregar el 22.

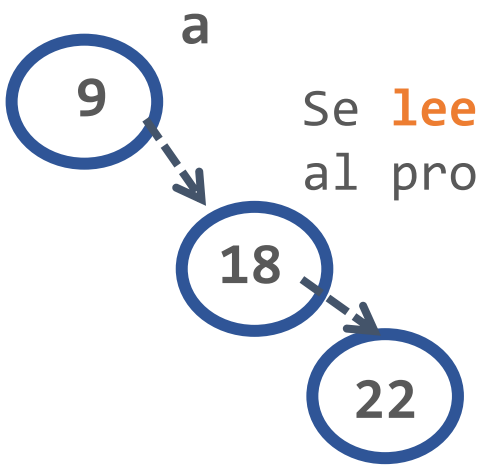
Reservo memoria dinámica y asigno los valores correspondientes

El procedimiento agregar termina y vuelve al programa principal en donde a ahora apunta **a** un nodo con valor 9 y su HI= nil y HD = 18 y 18 con su **HD= 22**.

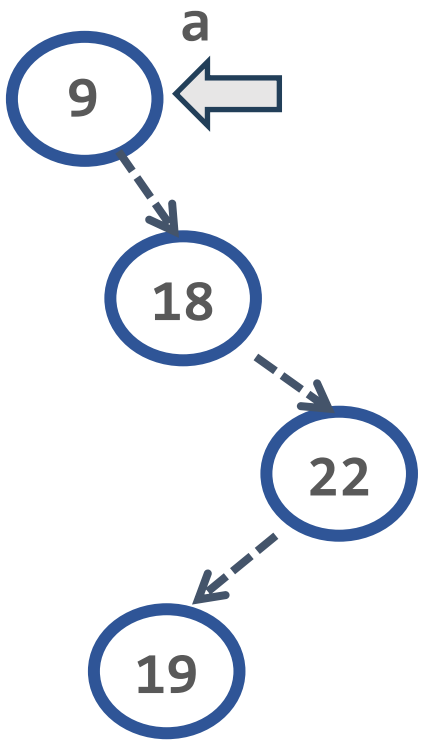
ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

Suponga que se leen los siguientes valores y se quieren ir agregando en un ABB (9, 18, 22, 19, 7, 50)

a = 9



Se **lee el valor 19 (num)** y se invoca al procedimiento agregar (a,num)



Como a<>nil, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja.

Comparo num (19) con lo que está apunta a (9) y como 18 > 9, hay que agregarlo a la derecha 9. Luego 18 <= 19 entonces hay que agregarlo a la derecha. Luego 19 <= 22, hay que agregarlo a la izquierda de 22. Como es nil, se encontró el lugar.

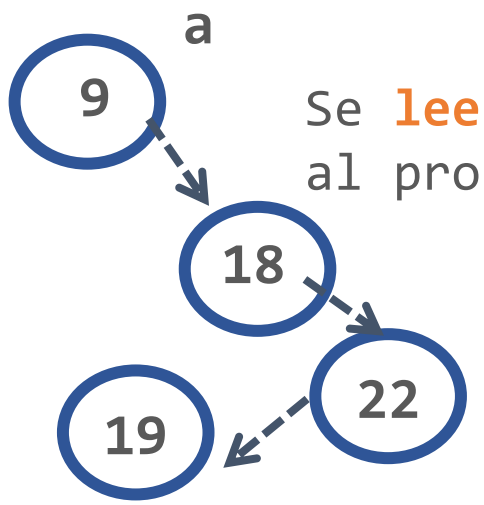
Reservo memoria dinámica y asigno los valores correspondientes

El procedimiento agregar termina y vuelve al programa principal en donde a = 9, su HD =18, a su vez su HD=22 y el **HI de 22 = 19**.

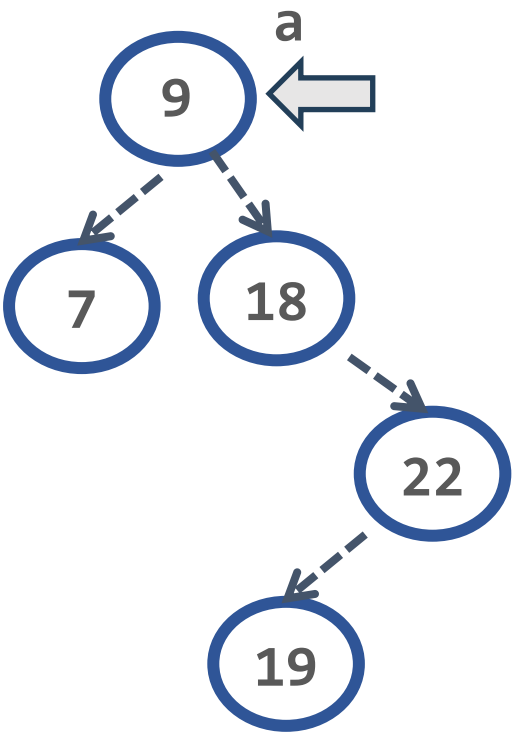
ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

Suponga que se leen los siguientes valores y se quieren ir agregando en un ABB (9, 18, 22, 19, 7, 50)

a = 9



Se **lee el valor 7 (num)** y se invoca al procedimiento agregar (a,num)



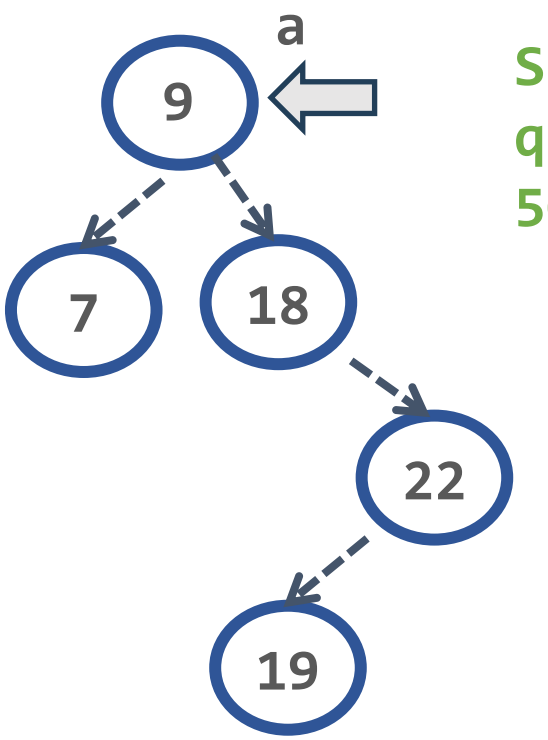
Como a<>nil, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja.
Comparo num (7) con lo que está apunta a (9) y como es 7<=9 se determina que hay que agregarlo a la izquierda de 9, que como 9 no tiene HI se encontró el lugar.

Reservo memoria dinámica y asigno los valores correspondientes

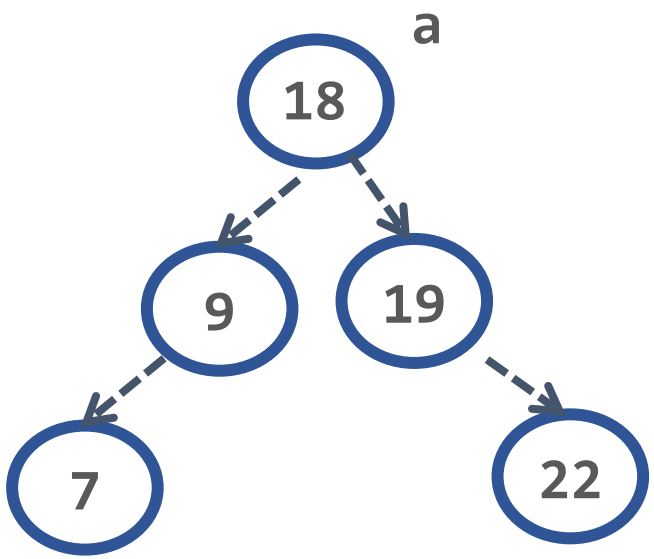
El procedimiento agregar termina y vuelve al programa principal en donde a = 9, su HD =18, y su **HI=7.**

ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

Suponga que se leen los siguientes valores y se quieren ir agregando en un ABB (9, 18, 22, 19, 7, 50)

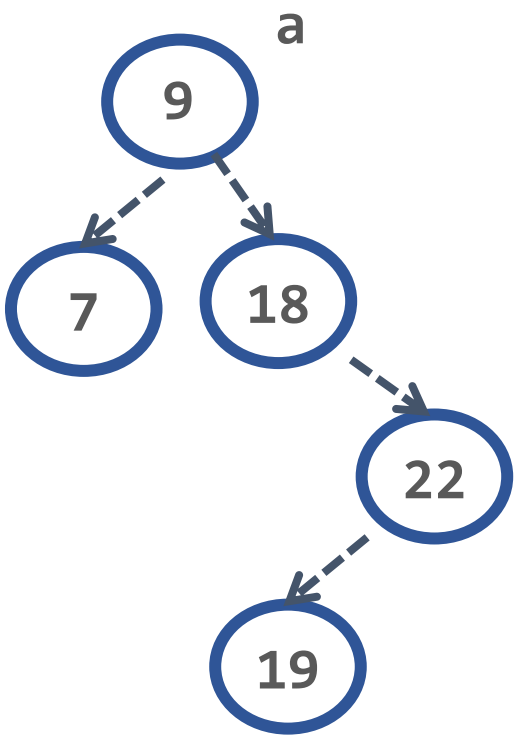


Si se leyeran los mismos valores pero en otro orden quedaría formado el mismo arbol? (18, 9, 7, 19, 22, 50)

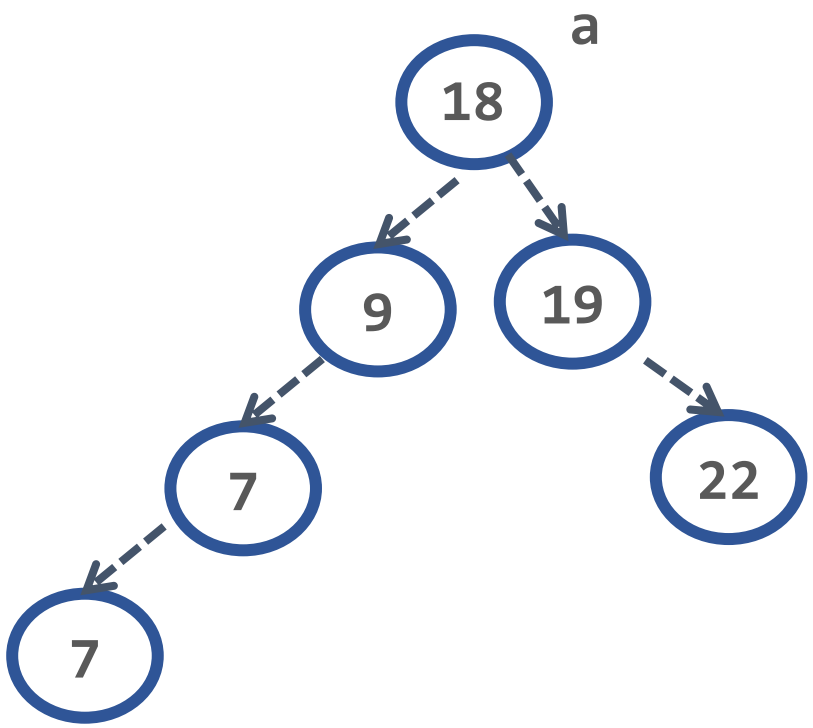


ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

Suponga que se leen los siguientes valores y se quieren ir agregando en un ABB (9, 18, 22, 19, 7, 50)



Qué ocurre si se leen valores repetidos?
(18, 9, 7, 7,19, 22, 50)



Cómo lo implementamos?

Cuál sería el caso base?

ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

```
Programa arboles;

Type
  arbol = ^nodo;

  nodo = record
   dato: integer;
   HI: arbol;
   HD: arbol;
  end;
Var
 abb:arbol; x:integer;
Begin
 abb:=nil;
 read (x);
 while (x<>50)do
  begin
   AGREGAR(abb,x);
   read(x);
  end;
End.
```

```
Procedure agregar (var a:árbol; num:integer);
Begin
  if (a = nil) then
   begin
      new(A);
      a^.dato:= num; a^.HI:= nil; a^.HD:= nil;
   end
   else
    if (num <= A^.dato) then agregar(a^.HI,num)
    else agregar (a^.HD,num)
End;
```

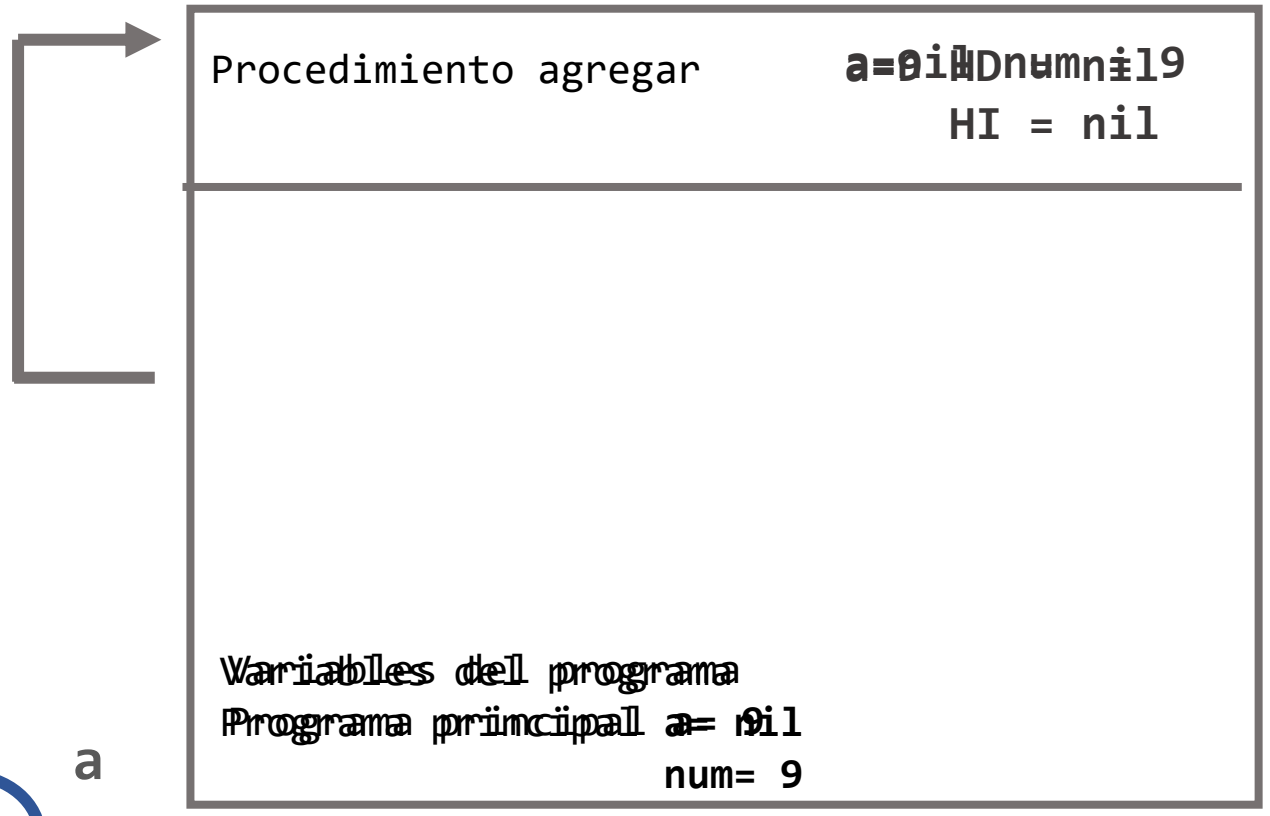
Cómo funciona?

ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

(9, 18, 22, 7)

```
Procedure agregar (var a:arbol; num:integer);
Begin
  if (a = nil) then
   begin
      new(a);
      a^.dato:= num; a^.HI:= nil; a^.HD:= nil;
   end
   else
    if (num <= a^.dato) then agregar(a^.HI,num)
    else agregar(a^.HD,num)
End;
```

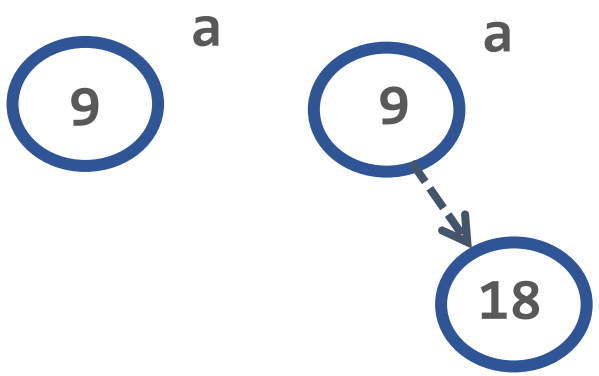
a

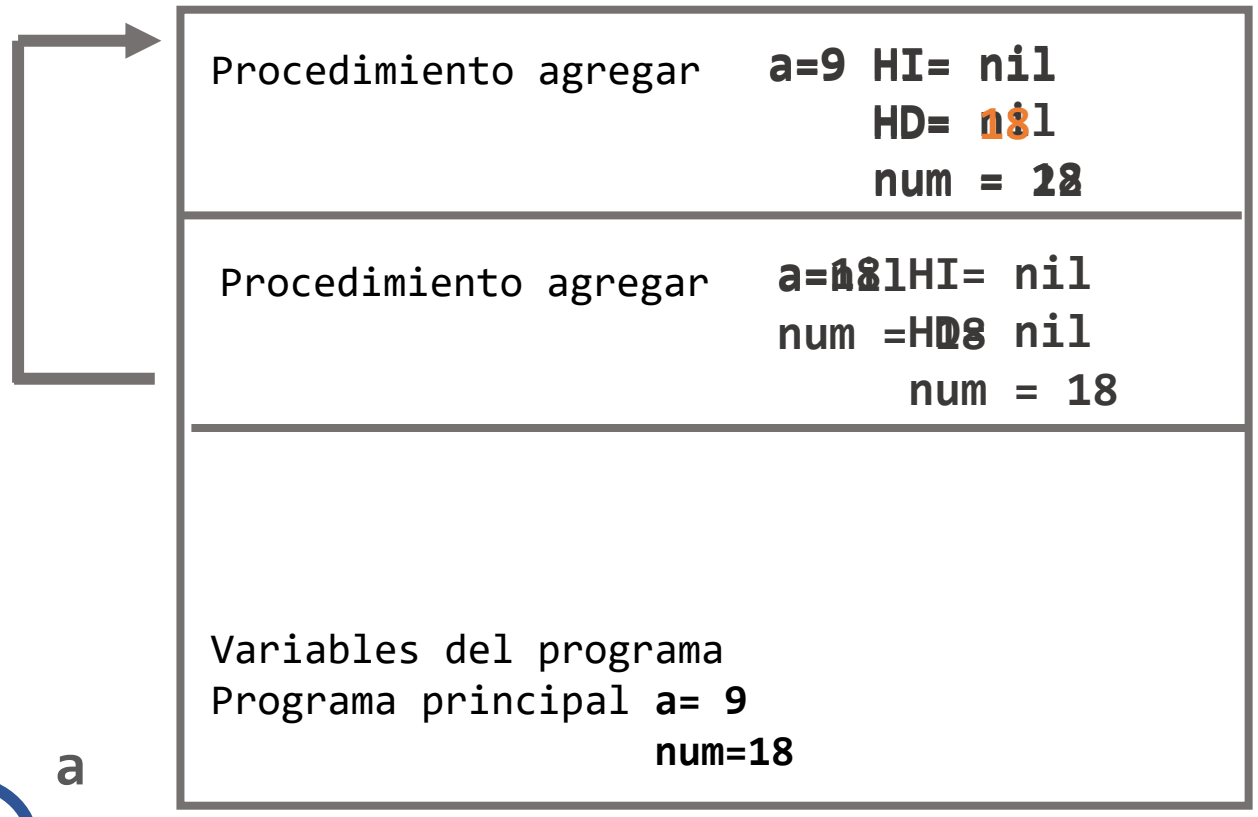


ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

(9, 18, 22, 7)

```
Procedure agregar (var a:arbol; num:integer);
Begin
  if (a = nil) then
   begin
      new(a);
      a^.dato:= num; a^.HI:= nil; a^.HD:= nil;
   end
   else
    if (num <= a^.dato) then agregar(a^.HI,num)
    else agregar(a^.HD,num)
End;
```

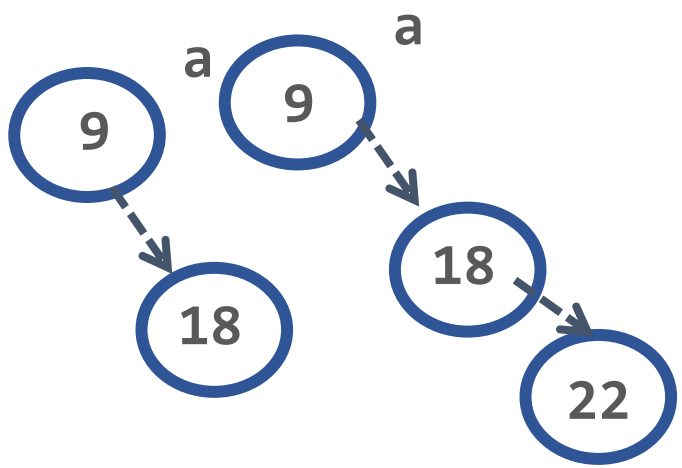




ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

(9, 18, 22, 7)

```
Procedure agregar (var a:arbol; num:integer);
Begin
  if (a = nil) then
   begin
      new(a);
      a^.dato:= num; a^.HI:= nil; a^.HD:= nil;
   end
   else
    if (num <= a^.dato) then agregar(a^.HI,num)
    else agregar(a^.HD,num)
End;
```



Procedimiento agregar	a=9 HI= nil HD= 18 num = 22
Procedimiento agregar	a=18 HI= nil HD= nil num = 22
Procedimiento agregar	a=22 HI= nil HD= nil num = 22
Variables del programa Programa principal	a= 9 num=22

ÁRBOLES BINARIOS DE BÚSQUEDA- CARGA

(9, 18, 22, 7)

```
Procedure agregar (var a:arbol; num:integer);
Begin
  if (a = nil) then
   begin
      new(a);
      a^.dato:= num; a^.HI:= nil; a^.HD:= nil;
   end
   else
    if (num <= a^.dato) then agregar(a^.HI,num)
    else agregar(a^.HD,num)
End;
```