CREACION ARBOL

```
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;
```

RECORRIDO ARBOLES

```
procedure EnOrden (a: arbol);
begin
    if (a <> nil) then begin
        EnOrden (a^.HI);
        write (a^.dato); // o otra accion
        EnOrden (a^.HD);
    end;
end;

procedure preOrden (a: arbol);
begin
    if (a <> nil) then begin
        write(a^.dato); o otra accion
        preOrden (a^.HI);
        preOrden (a^.HD);
    end;
end;

procedure postOrden (a: arbol);
begin
    if (a <> nil) then begin
        postOrden (a^.HI);
        postOrden (a^.HD);
        write(a^.dato);
    end;
end;
```

BUSCAR ARBOLES

devuelve V o F si lo encuentro ono:

```
function buscar (a: arbol; x: integer): boolean;
begin
    if (a = nil) then buscar := false
    else (a^.dato = x) then buscar := true
    else if (x > a^.dato) then
        buscar := buscar (a^.HD, x)
    else buscar := buscar (a^.HI, x);
end;
```

devuelve el nodo donde esta el valor buscado:

```
Function buscarNodo (a: arbol; x: integer): arbol;
begin
    if (a = nil) then buscarNodo := nil
    else (a^.dato = x) then buscarNodo := a
    else if (x > a^.dato) then
        buscarNodo := buscarNodo (a^.HD, x)
    else buscarNodo := buscarNodo (a^.HI, x);
end;
```

MINIMO Y MAXIMO ARBOLES

que retorne el valor minimo: cambiar por HD si se busca maximo

```
Function minimo (a: arbol): integer;
begin
    if (a <> nil) then begin
        if (a^.HI = nil) then
            minimo := a^.dato;
        else
            minimo:=minimo(a^.HI);
    end
    else minimo := 9999;
end;
```

devuelve el nodo que contiene el minimo: cambiar por HD si busca max

```
function minimoNodo (a: arbol): arbol;
begin
    if (a = nil) then minimoNodo := nil;
    else if (a^.HI = nil) then
        minimoNodo:=a;
    else
        MinimoNodo:= minimoNodo (a^.Hi);
end;
```

BUSQUEDA ACOTADA

Funcion que retorna cantidad entre 2 codigos: (sin incluir)

```
function Busqueda (a:arbol; inF, sup: integer): integer;
begin
   iF (a<>nil) then begin
      if (a^.d.cod >= sup) then
         Busqueda:= Busqueda(a^.HI, inF, sup)
      else
         if(a^.d.cod > inF) and (a^.d.cod < sup) then
            Busqueda:= 1 + Busqueda(a^.HI, inF, sup) + Busqueda (a^.HD, inF, sup)
       else if (a^.d.cod <= inF) then
            Busqueda:= Busqueda (a^.HD, inF, sup);
   end
   else Busqueda
end;
```

Para incluir realizar combios en rosa