Arboles

Tratamiento de “repetidos” (1)

Hemos analizado algunas situaciones en las que los elementos de nuestras estructuras tienen datos o información que se repite o necesitamos agrupar información.

Hasta el momento IGNORAMOS la información repetida!!

Datos de cuentas bancarias

Datos de un cliente en las compras de un año

Datos que se encuentran en otras estructuras anexas

Tratamiento de “repetidos” (2)

¿CÓMO LO RESOLVEMOS?

No hay una regla para ello.
¡Depende del problema que debamos solucionar!

Veamos algunos ejemplos…

Tratamiento de “repetidos” (3)

Se debe crear una estructura que totalice el saldo de diferentes cuentas bancarias a partir de los movimientos realizados.

Se necesita rápido acceso por nro. de cuenta.

Puede haber mucha información irrelevante para la solución dentro de los movimientos realizados.

Datos de cuentas bancarias

¿Sería un árbol? ¿Cómo deberíamos declararlo? ¿Qué cambiaría de lo que ya sabemos hacer?

```
insertar (arbol, mov)
  si arbol es nil
    creo nodo_nuevo y cuento con el primer valor
  sino
    si la cta. en árbol es > que en el movimiento
      insertar(hijo_izquierdo_del_árbol, mov);
    sino
      si la cta. en árbol es < que en el movimiento
        insertar(hijo_derecho_del_árbol, mov);
      sino {es igual}
        agrego el importe al total
```

Tratamiento de “repetidos” (4)

Se nos pide crear una estructura que contenga las compras de un cliente a medida que van ocurriendo.

Además debemos permitir la búsqueda eficiente por DNI del cliente.

Datos de un cliente en las compras de un año

¿Sería un árbol? ¿Cómo deberíamos organizarlo?

¿Qué cambiaría de lo que ya sabemos hacer?

```
insertar (arbol, compra)
  si arbol es nil
    creo nodo_nuevo y agrego una compra a la lista
  sino
    si el dni en árbol es > dni en compra
      insertar(hijo_izquierdo_del_árbol, dato);
    sino
      si el dni en árbol es < dni en compra
        insertar(hijo_derecho_del_árbol, dato);
      sino {es igual}
        agrego una compra a la lista del
```

Tratamiento de “repetidos” (5)

Se tiene un árbol con información bancaria. Esta estructura posee por cada DNI de cliente los datos de los movimientos realizados en su cuenta.

Se quieren generar los resúmenes de cuenta y se necesita imprimir los datos de cada cliente.

Los datos como nombre apellido y dirección, están almacenados en una lista ordenada por DNI.

Datos que se encuentran en otras estructuras

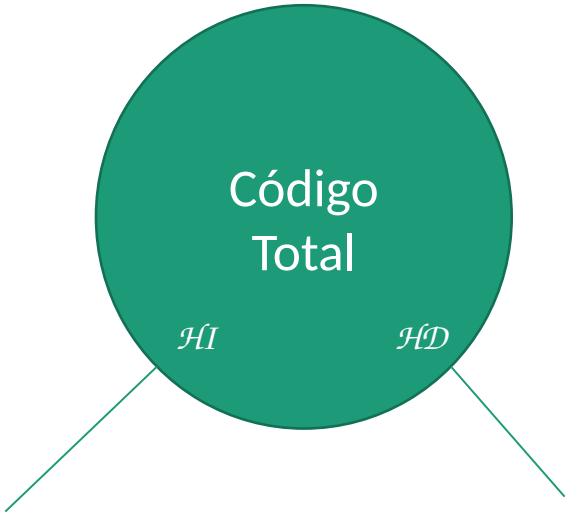
¿Cómo deberíamos recorrer el árbol?
¿Qué cambiaría de lo que ya sabemos hacer?

```
Procedure enOrden( a: árbol; L: lista
);
Var
d:cliente;
begin
  if ( a <> nil ) then begin
    enOrden (a^.HI);
    d = buscarDatos(L a^
    imprimirRes
    enOrde
  end;
end;
```

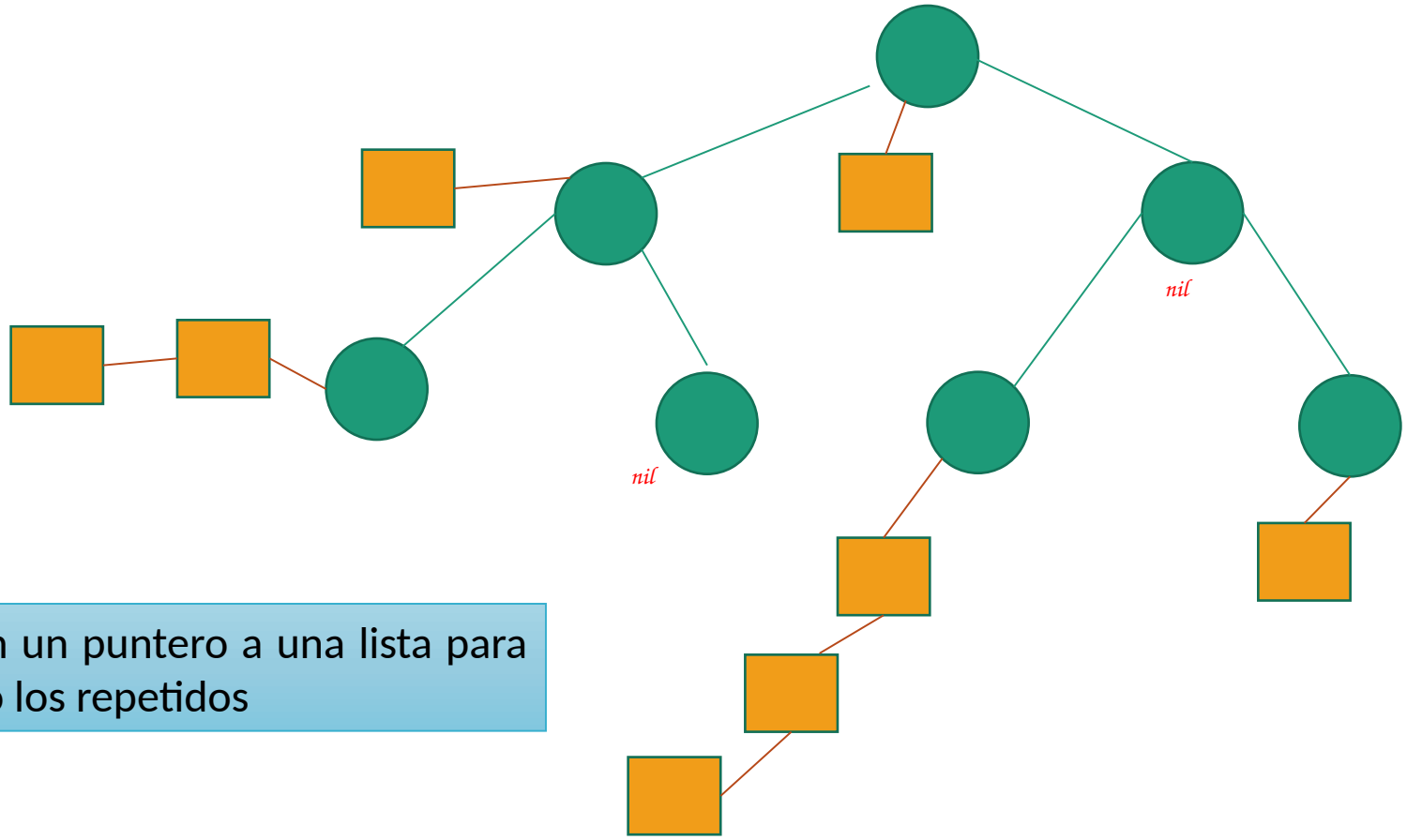
Buscar datos debe:
Buscar en la lista de clientes la info necesaria para el resumen

Síntesis

Usar un ABB y Totalizar/Contar en el nodo los repetidos

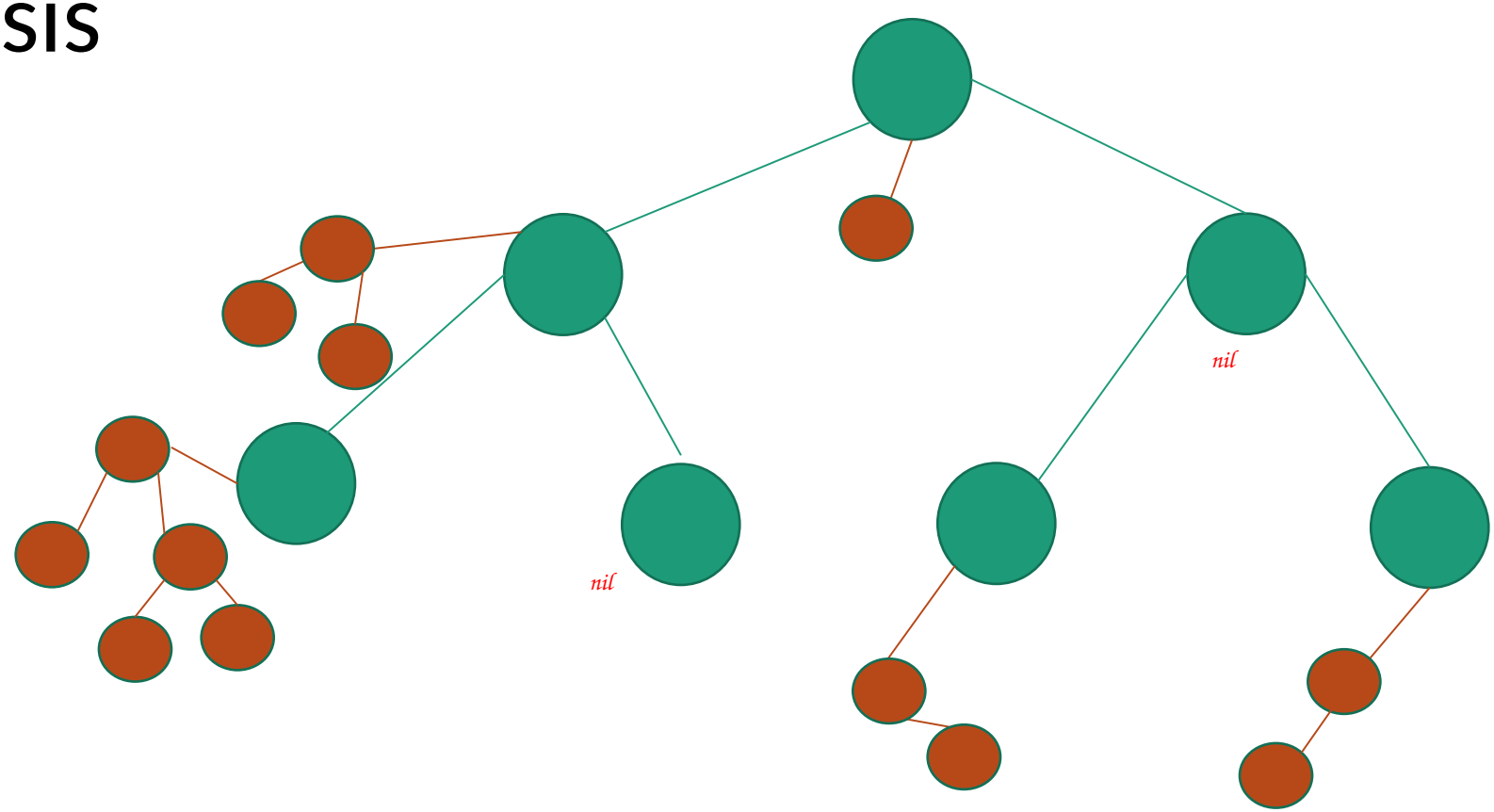


Síntesis



Usar un ABB con un puntero a una lista para contar en el nodo los repetidos

Síntesis



Usar un ABB con un puntero a un árbol para contar en el nodo los repetidos

Es para casos donde se necesite doble búsqueda eficiente

Actividades en Máquina

- **Descargar de Asignaturas** `ProgramaEncomiendas` y realizar las siguientes actividades:
- **ACTIVIDAD 9**

a) Crear una estructura eficiente para la búsqueda, que almacene para cada peso, los códigos de encomienda registrados para el mismo.

b) Imprimir a partir de la estructura generada, cada peso de encomienda con los códigos de encomienda registrados para dicho peso.