PROGRAMACIÓN I

TEORÍA – CECILIA SANZ

Temas

- ✓ Listas
- ✓ Características y Operaciones
- ✓ Ejemplos

¿Seguimos pensando?

LISTAS - OPERACIONES

Crear lista agregando los elementos al inicio.

Crear lista agregando los elementos al final

Insertar un nuevo elemento en una lista ordenada – Enlace al código en Pascal

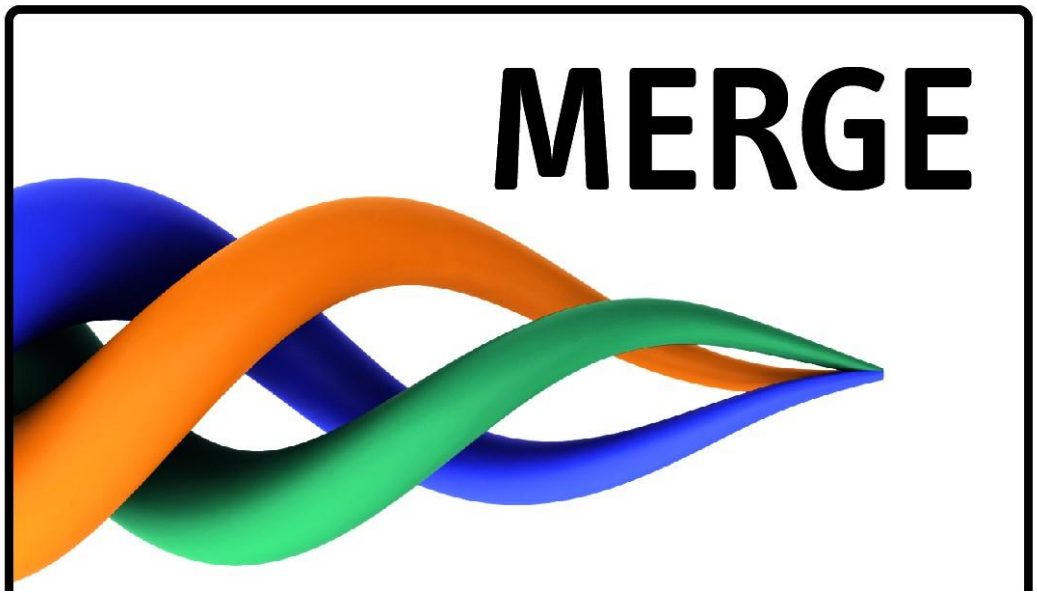
Recorrer una lista

Acceder al k-ésimo elemento de la lista

Eliminar un elemento de la lista

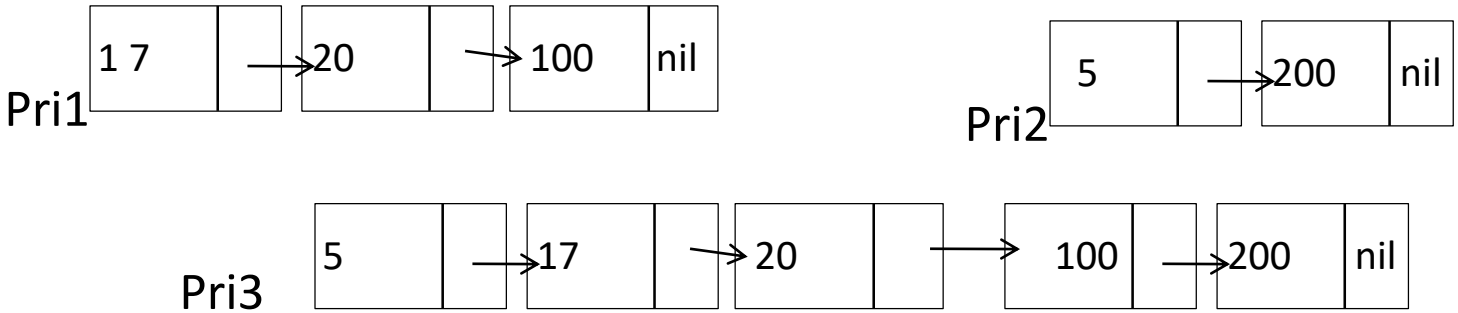
Combinar dos listas ordenadas formando una sola ordenada (Merge de Listas)

Merge de Listas

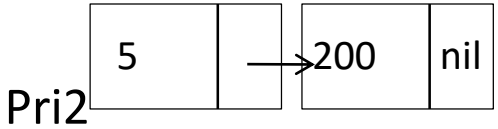


LISTAS – MERGE

Suponga que se tienen dos listas creadas de manera ordenada. Se pide implementar una algoritmo que combine ambas listas y genere una tercer lista ordenada.
Ambas listas están ordenadas por el mismo criterio. La lista nueva también es generada ordenada por el mismo criterio.



LISTAS – MERGE



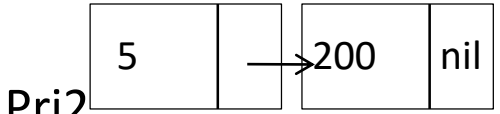
Obtengo el mínimo → 5 → Agrego el mínimo a la lista pri3 → Avanzo en la lista que tuvo el mínimo (pri2)

Obtengo el mínimo → 17 → Agrego el mínimo a la lista pri3 → Avanzo en la lista que tuvo el mínimo (pri1)

Obtengo el mínimo → 20 → Agrego el mínimo a la lista pri3 → Avanzo en la lista que tuvo el mínimo (pri1)

Obtengo el mínimo → 100 → Agrego el mínimo a la lista pri3 → Avanzo en la lista que tuvo el mínimo (pri1=nil)

LISTAS – MERGE



Obtengo el mínimo

➡ La lista que se terminó no se puede comparar ya que se estaría comparando con nil, entonces devuelve lo que está apuntando pri2.

Obtengo el mínimo

➡ Como ambas listas se terminaron el algoritmo termina

LISTAS – MERGE

CÓDIGO EN PASCAL

```
Program UsoMezcla;
Const fin=9999;
Type Lista= ^Nodo;
        Nodo= Record
                    Datos:integer;
                    Sig: Lista;
                End;

Var L1, L2, L3 : Lista;
```

```
…
Begin
 crearLista(L1);
 crearLista(L2);
 l3:= nil;
merge (l1,l2,l3);
End.
```

LISTAS – MERGE

```
Procedure merge (l1,l2: lista; var l3:lista);
Var min: integer;
Begin
  minimo(l1,l2,min);
  while (min <> fin) do
   begin
    AgregarAtras(l3,min); {no lo implementamos, es un agregar al final ya visto}
    minimo (l1,l2,min);
   end;
End.
```

LISTAS – MERGE

```
Procedure minimo( var l1:lista; var l2: lista; var min:integer);
Var
Begin
  if (l1 =  nil) and (l2=nil) then min:=fin
  else {alguno no es nil}
    if (l1<> nil) and (l2<> nil) then
      if (l1^.datos <= l2^.datos) then begin min:= l1^.datos; l1:= l1^.sig; end
      else begin min:= l2^.datos; l2:= l2^.sig; end
    else if (l1 <> nil) and (l2 = nil) then begin min:= l1^.datos; l1:= l1^.sig; end
         else begin min:= l2^.datos; l2:= l2^.sig end;
end;
```

Caso 1: las dos listas no tienen elementos

Caso 2: las dos listas tienen elementos

Caso 3: solo l1 tiene elementos

Caso 4: solo l2 tiene elementos

IDEAS FINALES

LISTAS – Reflexiones

Analicemos qué factores entran en juego al elegir cualquiera de las estructuras vistas hasta ahora:

✓Espacio

✓Tiempo

recuperar datos

almacenar datos

✓Parámetros

LISTAS – Reflexiones

Espacio: se refiere a la cantidad de memoria utilizada por la estructura de datos.

Suponiendo que tienen la misma cantidad de datos

- los arreglos ocupan menos memoria que las listas (por los enlaces)

LISTAS – Reflexiones

Tiempo: se refiere al tiempo que toma recuperar un dato de la estructura.

- acceso secuencial utilizado por las listas, para acceder al 4to. elemento debo pasar por los tres anteriores.

- acceso directo utilizado en el arreglo, para acceder al 4to. Elemento debo: a un desplazamiento base sumarle la cantidad de elementos que se tiene que mover * tamaño del elemento

  Dirección Elemento Actual = Dirección Base + (Offset * Indice)

LISTAS – Reflexiones

Tiempo: se refiere al tiempo que toma recuperar un dato de la estructura.

- acceso secuencial: requiere un tiempo que no es constante, depende el número del elemento al que se quiere acceder.

- acceso directo: requiere un tiempo fijo.

LISTAS – Reflexiones

Datos: se refiere a cómo se almacenan los datos.

- Listas: siempre hay lugar, distintos casos para agregar al principio, final u ordenado.
- Arreglos: no siempre hay lugar, distintos casos para agregar al principio, final u ordenado. Siempre debo validar si hay lugar.

LISTAS – Reflexiones

Parámetros: por valor o por referencia

- Valor: de la lista se copia solo el puntero inicial. En un arreglo se realiza una copia de todo.
- Referencia: de la lista se copia el valor inicial para poder ser modificado, del arreglo se copia la dirección en donde se encuentra almacenado.