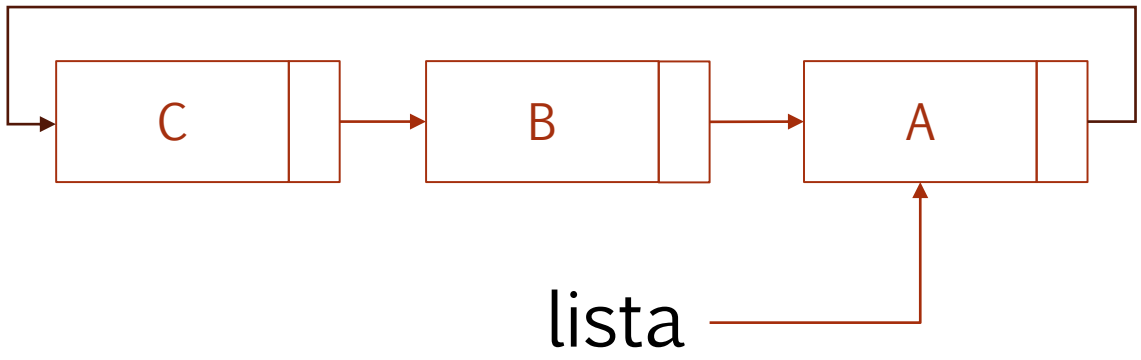
Algorítmica y Programación II

Tipos de datos recursivos
Lista ordenada, doble y multilista

Otras implementaciones

› Cuando nos referimos a la implementación de una cola o una pila dinámica, en general, pensamos en las implementaciones que definimos anteriormente.

› Si el puntero de una lista referencia al último nodo, que TDA podríamos crear?



Lista Ordenada

- En este caso debemos mantener una lista ordenada según algún criterio, por ejemplo podría ser los elementos de la mismas si fueran enteros.
- Tendremos que prestar especial atención al momento de insertar un nodo.
- Podríamos sintetizar la operación de inserción pensando dos situaciones:
  - Insertar el nodo al principio
  - Insertar el nodo más adelante (después del primero)

Lista Ordenada (ej - ascendente)

```
procedure insertar(var lista:TLista; e:TElemento);
begin
        if (lista=nil) or (lista^.info > e) then
                insertarAdelante(lista, e)
        else
                insertarEnOrden(lista, e);
end;
```

Lista Ordenada

```
procedure insertarAdelante(var lista:TLista;
        e:TElemento);
var aux:TLista;
begin
        new(aux);
        aux^.info := e;
        aux^.sig := lista;

        lista := aux;
end;
```

Lista Ordenada

```
procedure insertarEnOrden(lista:TLista; e:TElemento);
var aux:TLista;
begin
  new(aux);
  aux^.info := e;
  //busco la posición del nodo observando al siguiente
  while (lista^.sig <> nil) and (lista^.sig^.info < e)do
    lista := lista^.sig;

  aux^.sig := lista^.sig;
  lista^.sig := aux;
end;
```

Lista Ordenada

```
procedure listar(lista:TLista);
begin
      while(lista <> nil) do
            begin
            writeln(lista^.info);
            lista := lista^.sig;
            end;
end;
```

¿Cómo podríamos imprimir los elementos de forma descendente?

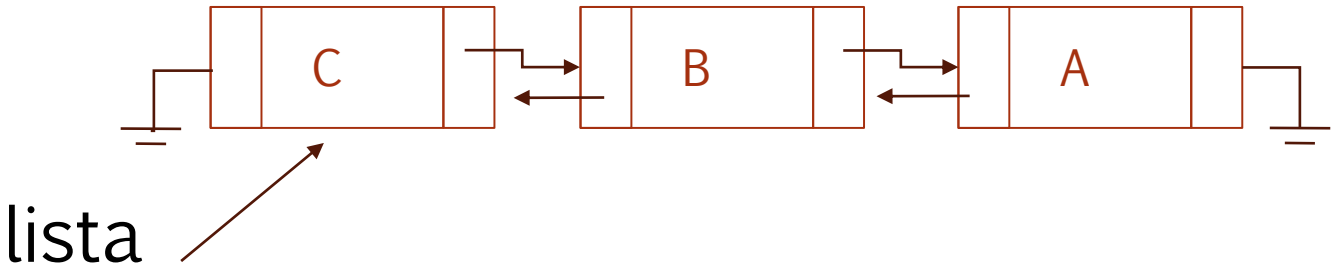
```
procedure eliminar(var lista:TLista; e:TElemento);
var aux, ant: Tlista;
begin
        aux := lista;
        ant := NIL;
        while(aux <> nil) and (e > aux^.info) do begin
                ant := aux;
                aux := aux^.sig;
        end;
        if(aux <> nil) and (aux^.info = e)then begin
                if (ant = nil)then
                        lista := aux^.sig
                else
                        ant^.sig := aux^.sig
        end;
        dispose(aux);
end;
```

Lista Ordenada -
¿qué hace, realmente inserta y enlaza?

```
procedure QueHace (var cab: tpun; e: tinfo);
var p: tpun;
begin
        if (cab = nil) or (e < cab^.info) then
                begin
                new(p);
                p^.info:= e;
                p^.sig := cab;
                cab:= P
                end
        else
                QueHace (cab^.sig, e)
end;
```

Lista Doblemente enlazada

› Además de la información propia el nodo cuenta con dos punteros, uno referencia al que lo antecede y el otro al que sucede.



Lista Doblemente enlazada

Para el caso de insertar un nodo (P) siempre deberemos actualizar 4 punteros:

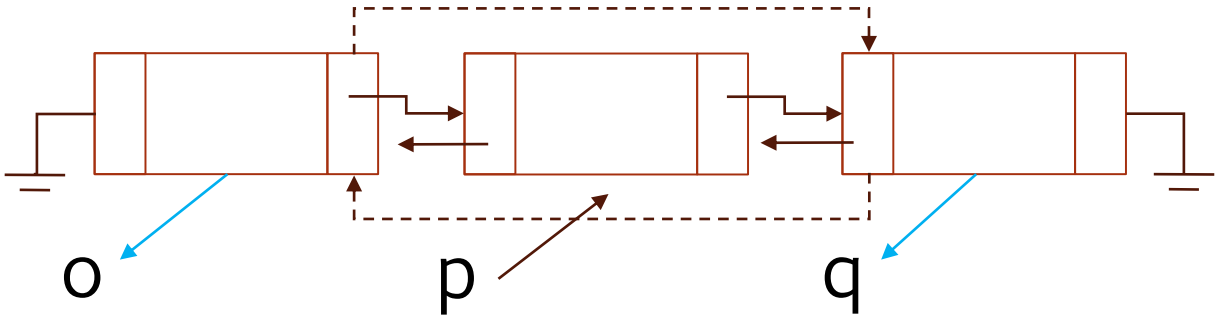
```
P^.ant := O;

P^.sig := O^.sig;

O^.sig^.ant := P;

O^.sig: =P;
```

Con los datos de O es suficiente para insertar a P



Ejercicio 1/2

Grafique y defina las estructuras de datos que considere más convenientes.

Suponga tener un archivo, con los datos de los empleados de una empresa. Entren los que se destacan:

DNI - Apellido y Nombre – Tipo de profesión

Se desea generar una estructura Dinámica, que permita ser recorrida o buscar información por distintos atributos (en este caso cualquiera de los tres anteriores).

Ejercicio 2/2

Grafique y defina las estructuras de datos, que a su criterio son las más eficientes yque permitan soportar la información para poder resolver el siguiente problema:

Un Universidad que tiene alrededor de 4000, alumnos de posgrado, les ofrece el dictado de 1500 cursos electivos. Los alumnos, en función de sus intereses podrán elegir cursar uno, varios o ninguno de ellos.

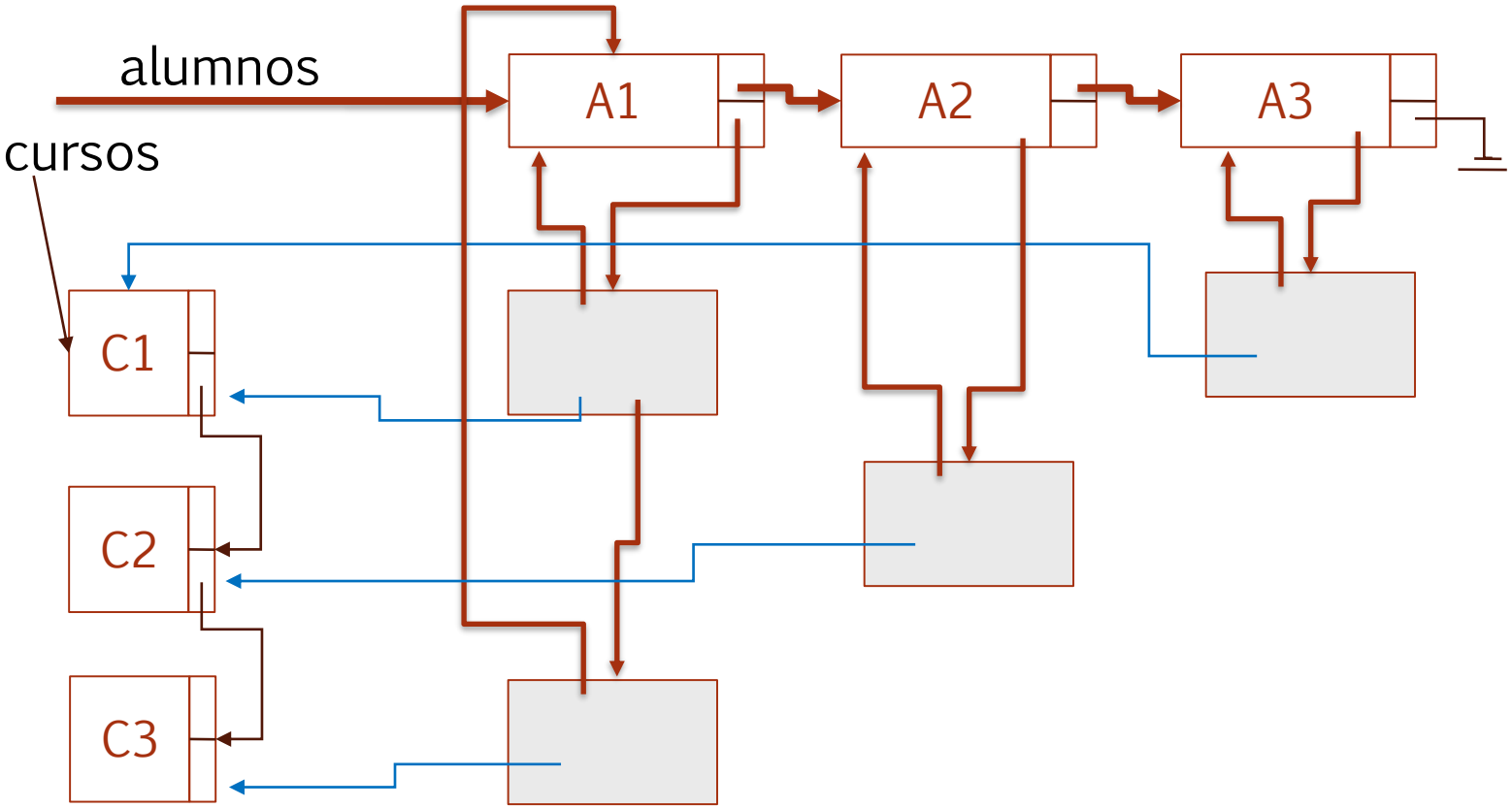
Se desea:

- Dado un alumno informar que en que cursos se ha inscripto.
- Dado un curso, que alumnos participan del mismo.

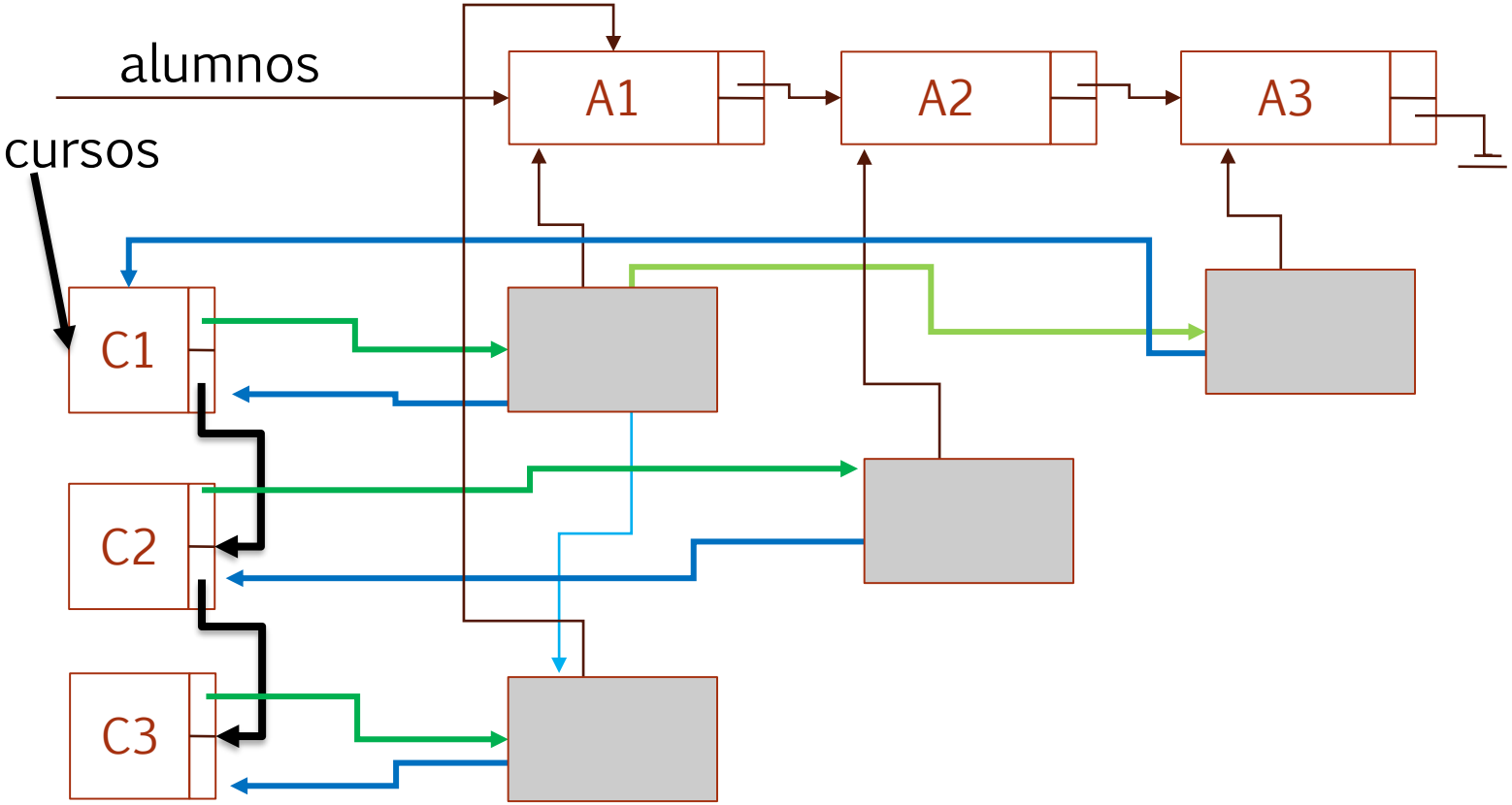
Multilistas

- Conjunto de nodos que **tienen más de un puntero y pueden estar en más de una lista simultáneamente**.
- Para cada tipo de nodo es importante distinguir los distintos campos puntero para realizar los recorridos adecuados y evitar confusiones.
- Es una estructura básica que se utiliza en Sistemas de Bases de Datos en Red.

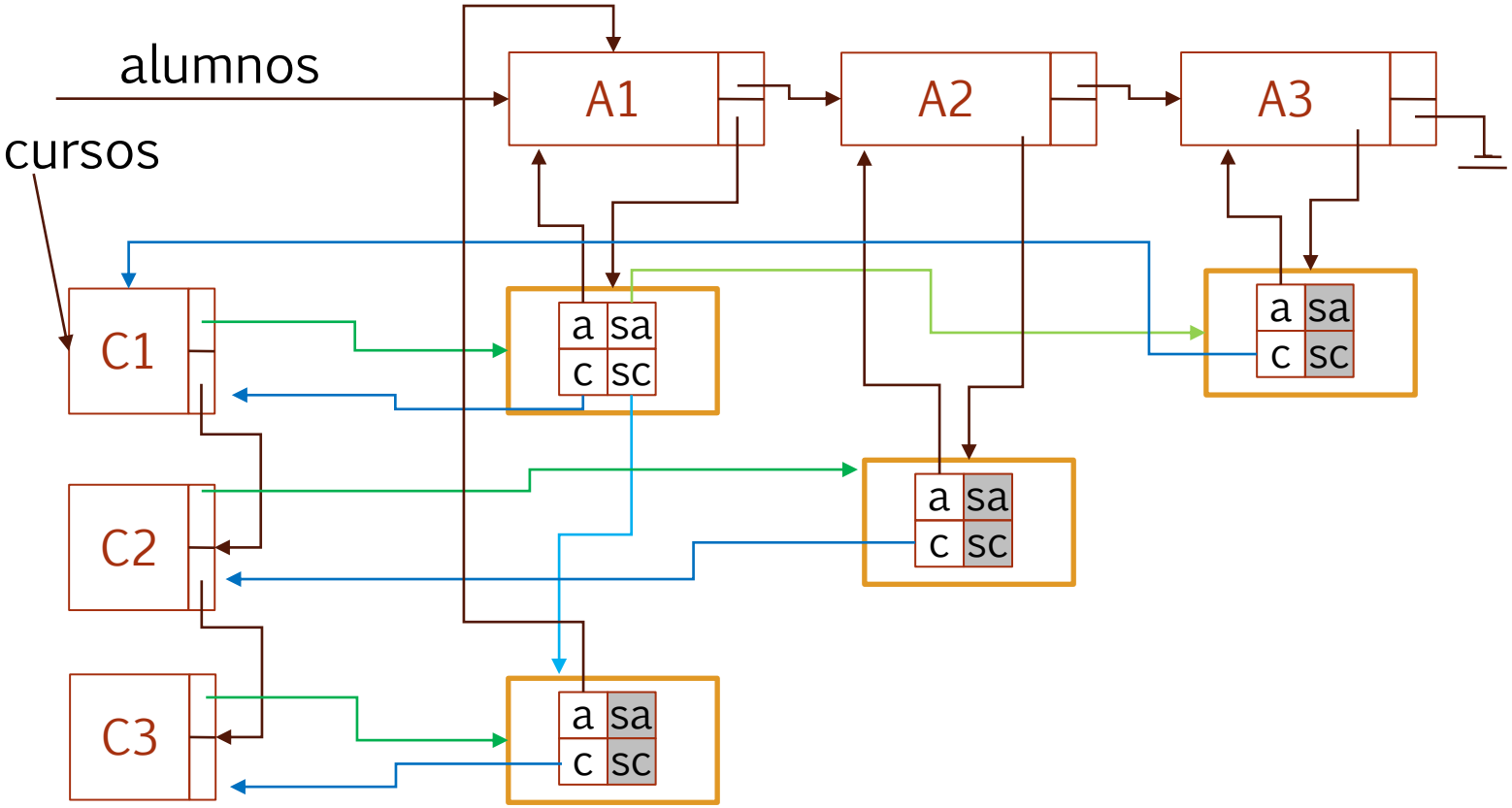
Multilista - alumnos



Multilista - cursos



Multilista



Multilistas

```
TAlumno = …; TCurso = …;

TListAlumn  = ^TPuntAlu;
TListaCurso = ^TPuntCur;
TNodoMulti  = ^TNodo;

TNodo = record
  alumno : TListAlumn;
  curso  : TListaCurso;
  alumnoSig : TNodoMulti;
  cursoSig  : TNodoMulti;
end;
```

```
TPuntCur = record
  info : TCurso;
  sig : TListaCurso;
  nodo : TNodoMulti;
end;

TPuntAlu = record
  info : TAlumno;
  sig : TListAlumn;
  nodo : TNodoMulti;
end;
```

Pasos para crear la multi lista:

1ro: Recibo el par <alu,cur>

2do: Busco o inserto alumno, me quedo con la referencia del nodo

3ro: Busco o Inserto el curso, me quedo con la referencia del nodo

4to: Creo el nodo multilista

Si ambos (alumno y curso) estaban creados, y además supongo que serán más alumnos que cursos, navego los nodos de la multilista buscando el último curso para insertar.

Si tuve que insertar solo uno, entonces busco por el otro el ultimo nodo de la multi lista para insertar

Si tuve que insertar ambos, creo directamente el nodo Multilista, y me quedo con su referencia.

Al alumno o al último Nodo del alumno (apuntan a cada curso que tiene) apunto al nuevo nodo multilista

Al curso o al último Nodo de los cursos (apuntan a cada alumno que los cursa) apunto al nuevo nodo multilista

Al nodo le asigno:

el alumno o el nodo Multilista anterior,

el curso o el nodo multilista anterior,

//si quisieramos insertar de manera ordenada sería otro tratamiento

alumnoSig NIL;

cursoSig NIL;

Conclusiones

- Al momento de elegir la estructura se debe tener en cuenta el tipo y la frecuencia de operaciones que se realizan sobre esta.
- En el caso de las lista doblemente enlazada, se podrá apreciar la cantidad de memoria adicional que requiere, aunque, si la operación de eliminación o inserción es la que mayor frecuencia tiene, entonces se puede ver que es una buena opción.

TDA - Bibliografía

- Data Structures. Nalle Dale.
- Intermediate Problem Solving and Data Structures - Helman P.- Veroff R.
- Estructuras de datos en C. A, Tenenbaum - Y, Langsam – M, Augenstein.
- https://en.wikipedia.org/wiki/List_of_data_structures