Les listes chainées

- On pourrait travailler uniquement avec des indices et des tableaux

- Un indice représentant un objet particulier

Inconvénient : c'est compliqué

- quand on veut supprimer un objet (que devient son indice ?)

- quand on veut insérer un objet (que devient son indice ?)

- quand on veut ajouter un objet (les tableaux doivent être agrandis)

Les listes chainées

- Un pointeur est un type de données dont la valeur fait référence (référencie) directement (pointe vers) à une autre valeur;

- Un pointeur référencie une valeur située quelque part d'autre en mémoire habituellement en utilisant son adresse;

- Un pointeur est une variable qui contient une adresse

Mémoire;

- Un pointeur permet de réaliser des indirections : désigner des objets, sans être ces objets.

Les listes chainées

- Un pointeur est un type de données dont la valeur **pointe vers une autre valeur.**

- Obtenir la valeur vers laquelle un pointeur pointe est appelé **déréférencer le pointeur.**

- Un pointeur qui ne pointe vers aucune valeur aura la valeur **nil**

Les listes chainées

- Une liste chaînée désigne une structure de données représentant une collection ordonnée et de taille arbitraire d'éléments.
- L'accès aux éléments d'une liste se fait de manière Séquentielle;
- chaque élément permet l'accès au suivant (contrairement au cas du tableau dans lequel l'accès se fait de manière absolue, par adressage direct de chaque cellule dudit tableau).
- Un élément contient un accès vers une donnée

Les listes chainées

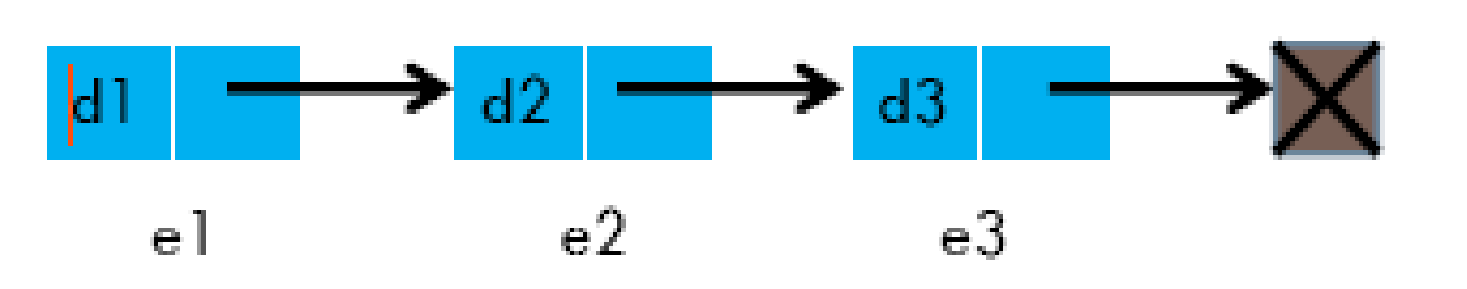
- Le principe de la liste chaînée est que chaque élément possède, en plus de la donnée, des pointeurs vers les éléments qui lui sont logiquement adjacents dans la liste;
- premier(L) désigne le premier élément de la liste;
- nil désigne l'absence d'élément;

Liste simplement chaînée :

- donnée(elt) désigne la donnée associée à l'élément elt;
- suivant(elt) désigne l'élément suivant elt

Les listes chainées

Représentation : ⟶ correspondant au suivant.



premier(L)= e1

suivant(e1) = e2

suivant(e2) = e3

suivant(e3) = nil

Les listes chainées

Opérations:

Trois opérations principales :

- Parcours de la liste
- Ajout d'un élément
- Suppression d'un élément

A partir de là d'autres opérations vont être obtenues : recherche d'une donnée, remplacement, concaténation de liste, fusion de listes, etc.

Les listes chainées

Liste et Tableau:

Le principal avantage des listes sur les tableaux :

- L'ordre des éléments de la liste peut être différent de leur ordre en mémoire.
- Les listes chaînées vont permettre l'ajout ou la suppression d'un élément en n'importe quel endroit de la liste en temps constant.

En revanche, certaines opérations peuvent devenir coûteuses comme la recherche d'un élément contenant une certaine donnée.

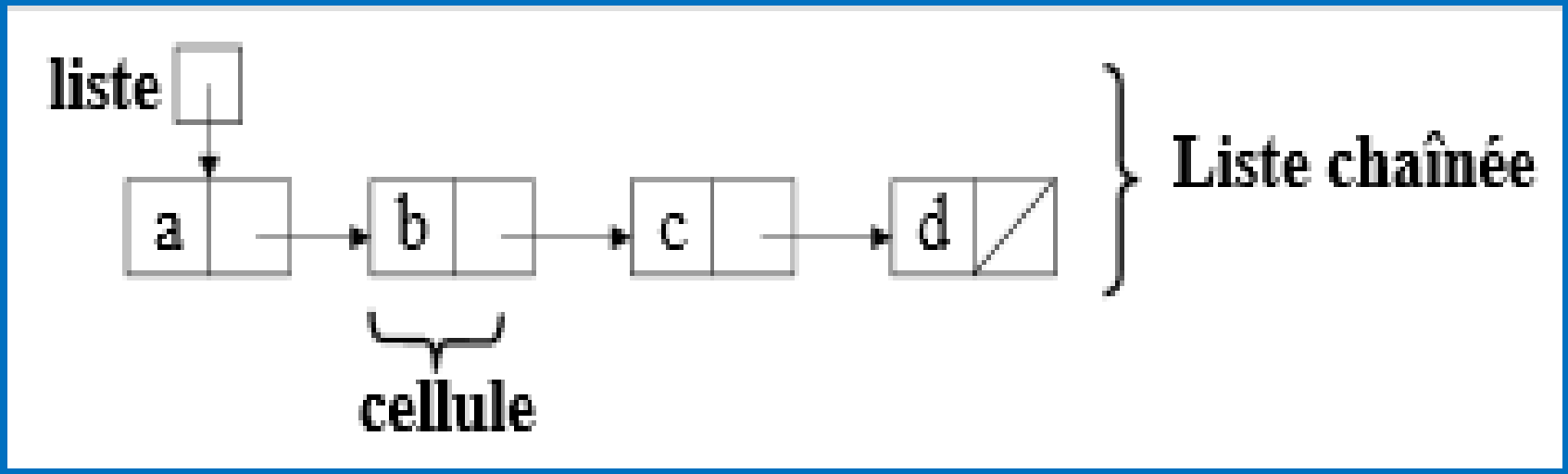
Pas de recherche dichotomique dans une liste : on ne peut pas atteindre le ième élément sans parcourir

Les listes chainées

- une liste chainée est composée d'un ensemble de cellules.

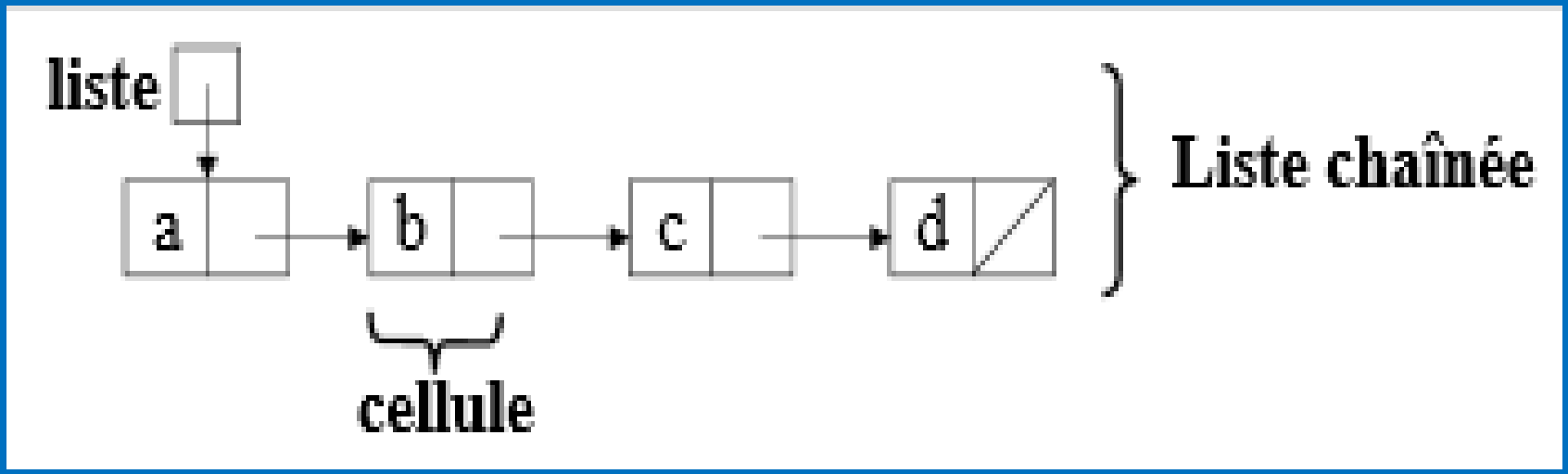
Une cellule est composée de:

- d'un élément de la suite
- d'un lien vers la cellule suivante (pointeur).



Les listes chainées

Liste contient l'adresse de la première cellule, qui à son tour contient l'adresse de la cellule suivante.



Les listes chainées

Représentation:

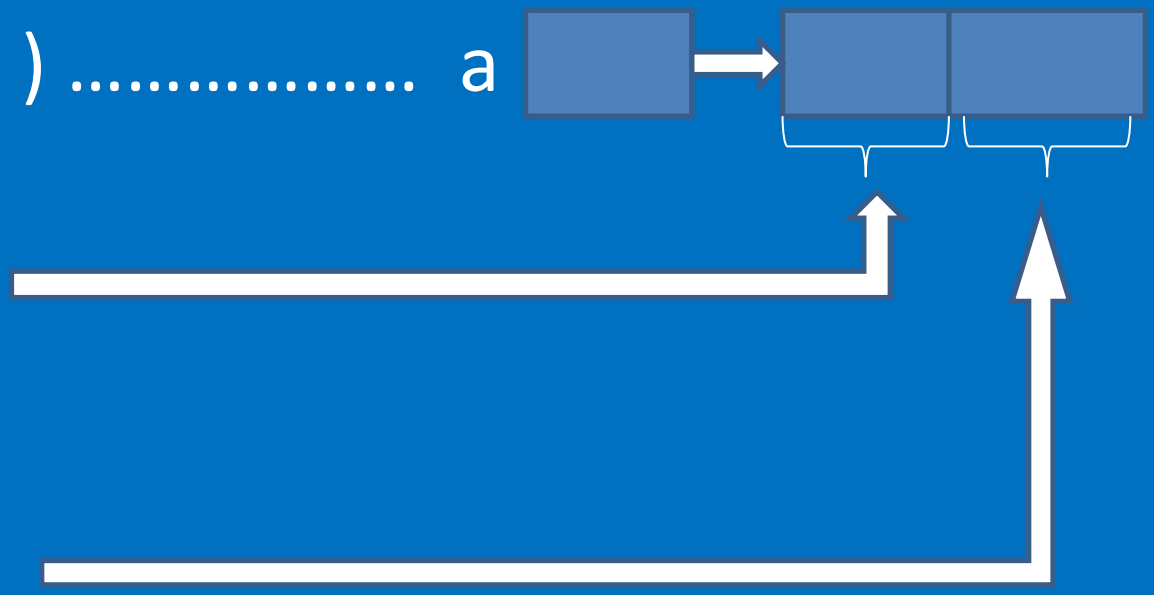
```
Type liste =^cellule;
        cellule = enregistrement
                info :  typeelement;
                suivant : liste;
Fin;
Var a : liste;
```

Les listes chainées

Manipulation:

Variable a : liste; …….. a ?

Allouer (a) …….. a



a^. Info

a^.suivant

Les listes chainées

Création d'une liste qui représente (x,y) :

Var a, p : liste;

création d'une liste vide :

a:= nul ;

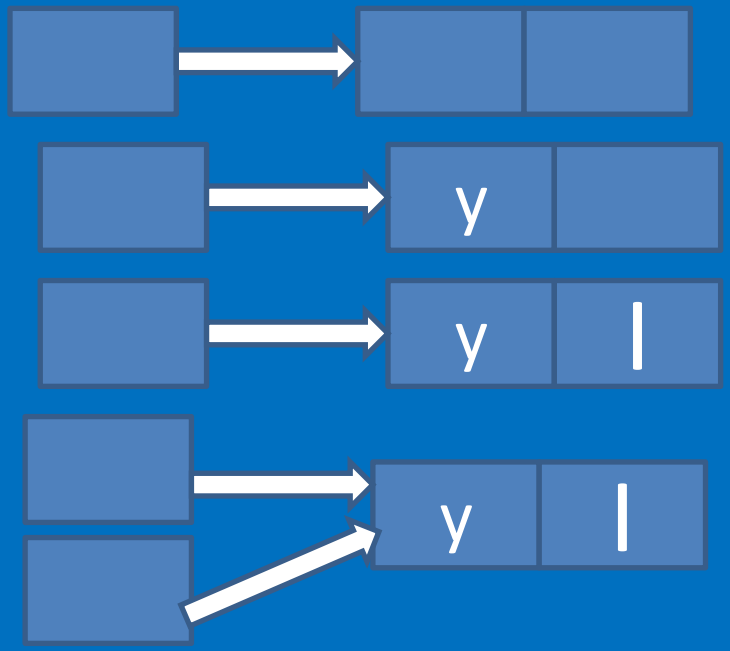
Création de la liste contenant (y):

Allouer (p);

P^.info := 'y';

P^.suivant := a

a:= p;



Les listes chainées

Création de la liste a contenant (x,y):

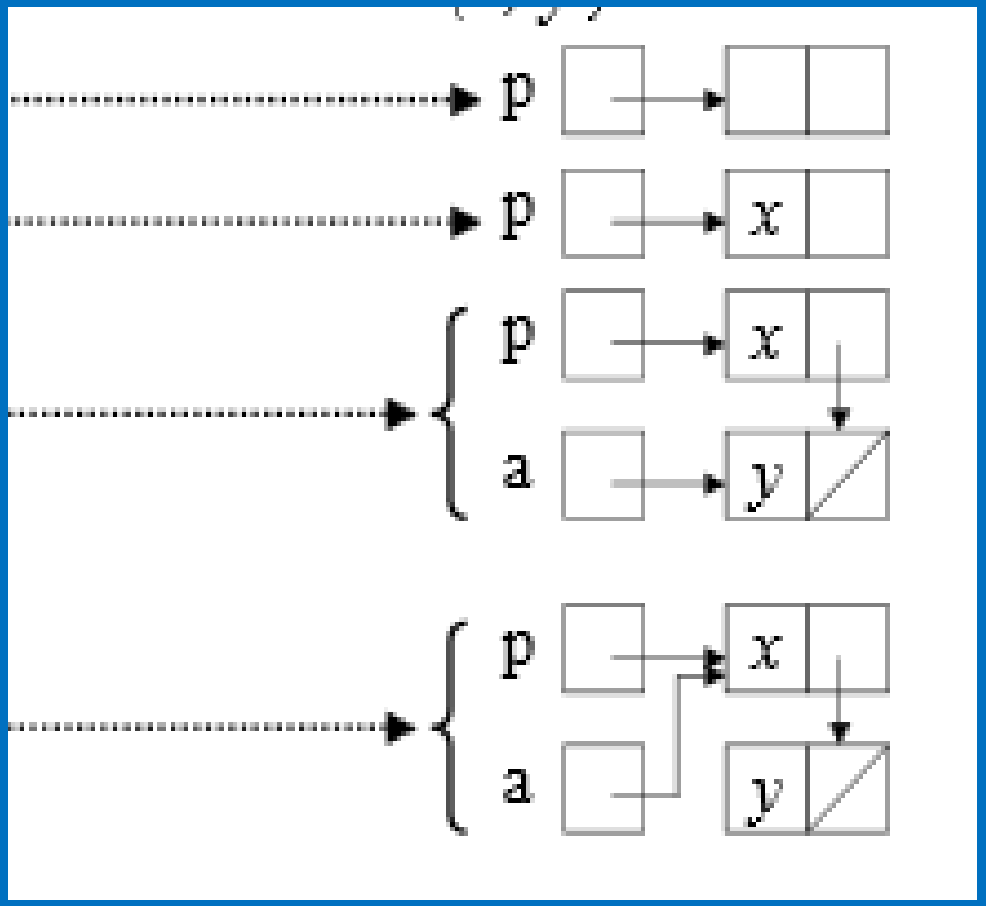
Allouer(p)

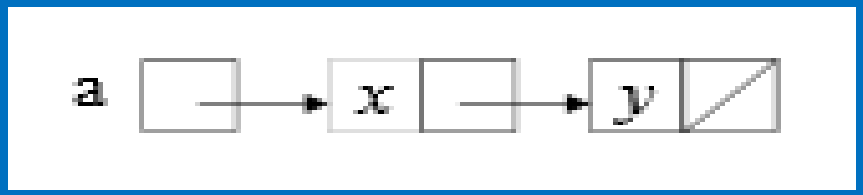
p^.info := x ;

p^.suivant := a ;

a:= p

On obtient :





Les listes chainées

Initialiser une liste à vide:

```
Procedure initListe (var : liste)
Debut
        a := nul;
Fin
```

Tester si une liste est vide:

```
Fonction listevide (a : liste) : booleen
Debut
        listevide := (a = nul);
Fin;
```

Les listes chainées

Insertion en tète de liste:

```
Procedure insertete (elem : typeElement, var a : liste)
Var p : liste;
Debut
        allouer (p);
        p^.info := elem;
        p^.suivant := a ;
        a := p ;
Fin;
```

Les listes chainées

Opérations sur les listes chainées