Algorithmique Partie 2

Formation

Développeur Web et Web Mobile

Module : 03

Développer la partie front-end d'une application web

Séquence : 03

Développer une interface utilisateur web dynamique

Séance : 01

Ecrire un algorithmique

Les tableaux (d2 à d10)

Les agrégats (d11 à d13)

Les structures de données (d14 à d19)

Les tableaux

⇨ introduction

- Problème :

  On veut faire le même traitement sur un grand nombre de variables du même type.

- Remarques :
  - Les noms des variables ne sont pas dynamiques (le traitement de l'algo ne peut pas modifier le nom d'une variable, il est donné à la déclaration).
  - Il serait bien qu'une structure de données portant un nom unique puisse conserver une collection de valeurs du même type.

- Exemple :

Saisir une suite de notes, puis afficher la moyenne.

⇨ Nécessité de conserver les notes en mémoire

```
note1,note2, … : réel
moyenne : réel
constante NBR_NOTE ← 10
saisir(note1)
saisir(note2)
…

moyenne ← ( note1 + note2 + …. ) / NBR_NOTE
afficher(moyenne)
```

Problèmes :

- Il y a autant de variables que de notes à saisir
- Le nombre de notes est fixé
- On saisit toujours le même nombre de notes

Les tableaux

⇨ Exemples d'applications

- Collection de valeurs
  type entières, réelles, booléennes
- Collection de noms
  type chaine
- Collection de caractères
  type caractère
- Collection d'employés
  type Employé : nom, prénom, matricule
- Collection d'ouvrages :
  type Ouvrage : titre, auteur, éditeur, …

Les tableaux

⇨ Traitements

- Ce que nous voulons faire :
  - créer des tableaux
  - remplir le tableau avec des valeurs
  - récupérer, consulter des valeurs d'un tableau
  - rechercher si une valeur particulière existe
  - modifier certaines valeurs
  - trier les valeurs suivant un certain critère
  - ...

- Un tableau est une variable contenant une collection de valeurs du même type

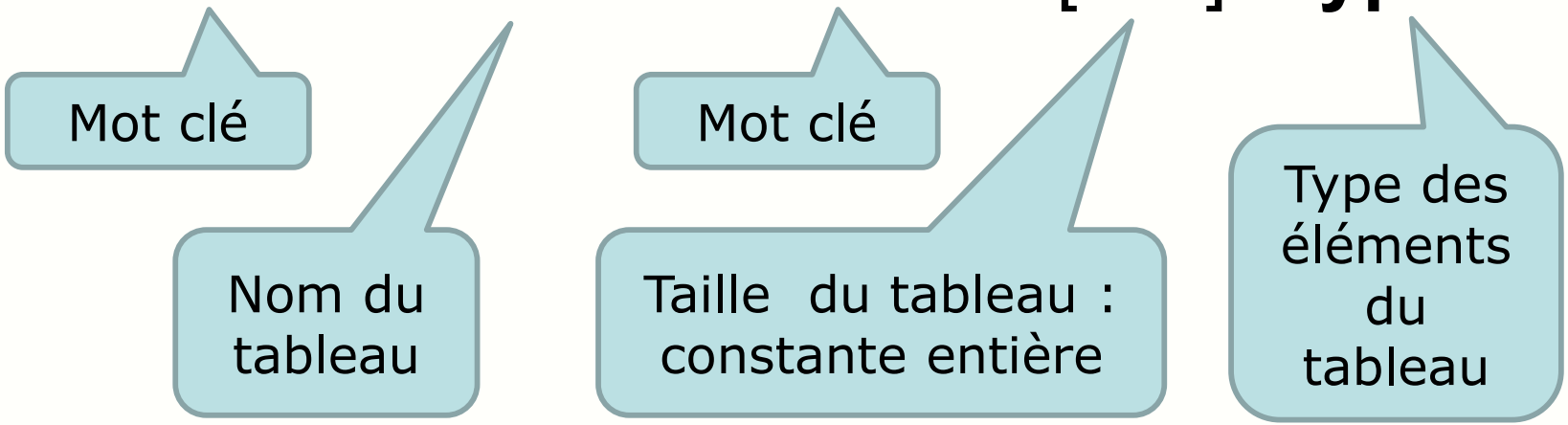
notes

0	1	2	3	4	5	6	7
13	08	12	19	14	10	16	11

- *Remarque :*
  - *appeler cette variable **tabNotes** plutôt que **notes***
  - *il faut pouvoir accéder aux éléments individuellement : notion **d'indice***
  - *en général l'indice d'un tableau démarre à 0*
  - *le nombre d'éléments est fixé par déclaration*
  - *valeurs du même type*

- La déclaration d'un tableau indique son nom, sa taille et le type de ses éléments :

variable tabNom : **tableau** [dim] : **type**



Exemple :

variable tabNotes : tableau[30] : réel

Attention : vous pouvez trouver dans la littérature la dimension sous forme de 2 valeurs; l'indice min et l'indice Max : ex tabl[1,10] à ne pas confondre avec un tableau à 2 dimension (voir ci après)

⇨ Utilisation de l'indice

- Accès en lecture :

  afficher(tabl[4])

  L'indice est donné par une constante : le contenu du tableau à l'indice 4 est affiché à l' écran

- Accès en écriture :

  saisir(tabl[5])

  la valeur entrée par l'utilisateur est enregistrée dans le tableau à l'indice 5

  tabl[3] ← 18

  ou

  idx ← 3

  tabl[idx] ← 18

  L'indice est donné par une variable (ex. idx) préalablement déclarée et initialisée :
  la valeur 18 est placée dans le tableau à l'indice 3

⇨ Initialisation

- Un tableau peut être initialisé (donner les valeurs de ses éléments) en une seule instruction :

```
variable tabNotes : tableau[30] : réel

tabNotes = {12.5; 14.0; 5.5; 18.0}
```

- Ceci est équivalent aux 4 instructions :

```
tabNotes[0] ← 12.5

tabNotes[1] ← 14.0

tabNotes[2] ← 5.5

tabNotes[3] ← 18.0
```

Les tableaux

⇨ Plusieurs dimensions

- Un tableau peut avoir plusieurs dimensions.
- Il n'y a pas de limite.
- A chaque dimension doit correspondre un indice.

Exemple : tableau à 2 dimension

- variable tabNotes : tableau[2][6] : réel

	0	1	2	3	4	5
0	13	08	12	19	14	10
1	19	7	13	10	16	14

- affiche(tabNotes[1][2])affiche la valeur 13

- Déclaration :

variable tabNom : tableau [dim1][dim2] : type

Les agrégats

Introduction

- Il est souvent intéressant de pouvoir regrouper sous une même appellation des données de nature différentes.

  on parle alors d'**agrégat, d'enregistrement ou de structure**

- Un agrégat possède un identifiant définissant un nouveau type de donnée.
- Chaque champ ou donnée qui le constitue à lui aussi son propre identifiant et type.
- L'accès à un champ se fait par:

  nomAgrega**.**nomDuChamp

Déclaration

- Déclaration :

```
type <identificateur> = agregat
    champ1: type 1
    champ2 : type 2
    …
    champN: type N
fin_agregat
```

- Utilisation :

  variable nomVar : identificateur

- Le type des champs peut être un agrégat

Exemple

- Exemple :

  type ***Personne =*** *agregat*

  nom : chaine

  prenom : chaine

  age : entier

  fin_agregat

- Utilisation
  - déclaration d'une variable :

    variable pers1 : Personne

    variable carnetAdr : tableau[100] : Personne
  - accès aux champs :

    pers1.nom = "Dupond"

    si (carnetAdr[1].age > 18) alors …

Les structure de données

structures dynamiques

- Il est souvent intéressant de gérer de façon dynamique les données en mémoire.
- Exemple, lors de la lecture d'informations dans un fichiers. Leur nombre est inconnu. On ne sait pas à l'avance combien de valeurs on a. Déclaration d'un tableau ?
- Plusieurs structures existent :
  - Les piles
  - Les files
  - Les listes chaînées
  - Les arbres

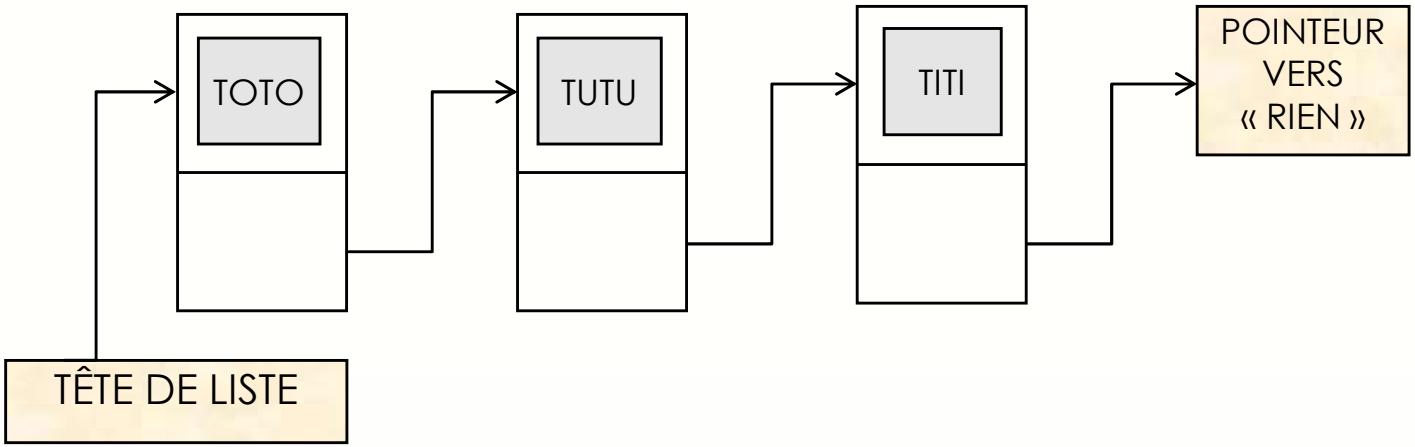
Les structure de données

structures dynamiques : les piles (files)

- Une pile est décrite par un ensemble d'éléments de même type.
- Il existe **3 opérations** pour gérer une pile :
  - ***Deposer*** un élément
  - ***Enlever*** un élément
  - ***Vider*** tous les éléments
- Fonctionne suivant 2 modes :
  - **FIFO** (**F**irst **I**n **F**irst **O**ut)
  - **LIFO** (**L**ast **I**n **F**irst **O**ut) appelée FILE

structures dynamiques : les chaînes

- Les listes chaînées peuvent **se représenter physiquement** sous forme d'un double vecteur : le premier contenant **les éléments**, le second les pointeurs sur **les éléments** :



Les structure de données

structures dynamiques : les chaînes

- Opérations possibles sur une liste chaînée :
- L'accès à un élément particulier de la liste : celui-ci n'est **pas réalisé** par l'intermédiaire d'un indice mais **par rapport à un autre élément** de la liste grâce aux fonctions : « **Premier** » qui ramène le premier élément de la liste et « **Suivant** » qui permet d'avancer dans la liste et de ramener élément.
- **L'insertion** d'un élément dans la liste.
- La **suppression** d'un élément de la liste.
- Le test déterminant si la **liste est vide**.

Les structure de données

structures dynamiques : les arbres

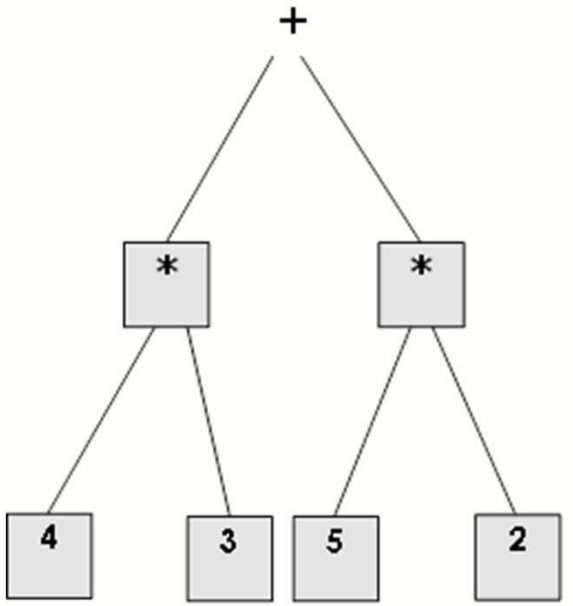
- C'est un ensemble d'éléments organisés de façon **hiérarchique**.
- Permettent de représenter un très grand nombre de situations et de phénomènes.
- A l'image d'un arbre généalogique, on appelle
  - racine : l'entrée de l'arbre
  - nœuds : l'embranchement vers d'autres nœuds
- On parcours l'arbre en suivant cette hiérarchie

Les structure de données

structures dynamiques : arbre binaire

- Un arbre binaire est un arbre dont chaque nœud ne possède que 2 branches.
- Utilisé dans les algorithme

  de tri
- 3 parcours possibles :
  - **<u>préordre</u>** : + * 4 3 * 5 2
  - **<u>postordre</u>** : 4 3 * 5 2 * +
  - **<u>inordre</u>** : 4 * 3 + 5 * 2



Fin de la deuxième partie