Tableaux et chaînes de caractères

Dr Khadim DRAME
kdrame@univ-zig.sn

Département d'Informatique
UFR des Sciences et Technologies
Université Assane Seck de Ziguinchor

30 juin 2021

Plan

Introduction

- Les types présentés depuis le début du cours sont **simples** : manipulation de valeurs simples.
- Beaucoup de problèmes nécessitent de manipuler des données plus **complexes** (structurées) :
  - traiter simultanément 20 valeurs (par exemple, des notes d'étudiants pour calculer la moyenne de la classe) ;
  - manipuler les températures moyennes mensuelles d'une année.
- Un **type structuré** est un type défini à partir d'autres types.

Introduction

Utilité des tableaux

- Exemple

  Calculer la moyenne d'une classe de 20 étudiants et les classer.

  Solution avec les moyens dont nous disposons :

  - déclarer 20 variables N1, N2, ..., N20 ;
  - calculer la moyenne m $\leftarrow$ (N1+N2+...+N20)/20.

- Très lourd !

  $\Rightarrow$ Solution : utiliser un **tableau**, qui permet de

  - rassembler toutes les valeurs dans une seule variable ;
  - manipuler simplement ces valeurs en utilisant une boucle.

Plan

Le type tableau

Définition

- Le **type tableau** est un type structuré communément utilisé en programmation.
- Un **tableau** permet de représenter un ensemble de valeurs de **même type** de données.
- Chaque élément (valeur) du tableau est accessible via un **indice**.
- L'**indice** est de type ordinal (généralement entier) qui permet de repérer chaque élément du tableau.

Le type tableau

Déclaration d'un tableau à une dimension

- **Syntaxe**

  var

  <nom_tab> : **array** [indice_min**..**indice_max] **of** <type> ;

  où

  - <nom_tab> est le **nom** du tableau ;
  - <indice_min> est l'**indice minimal** du tableau ;
  - <indice_max> est l'**indice maximal** du tableau ;
  - <type> est le **type des éléments** du tableau.

- Exemples

  var

  ages : array[1..50] of integer ;

  notes : array[1..10] of real ;

Le type tableau

Déclaration d'un tableau à une dimension

- Le nom du tableau doit respecter les règles d'écriture des identificateurs.
- Les indices du tableau doivent être de type ordinal.
- Les éléments d'un tableau peuvent être de tout type.
- La **taille** d'un tableau (nombre d'éléments qu'il peut contenir) est fixée à priori.
- La taille n'est **pas modifiable** au cours de l'exécution du programme.
- Attention : **il ne faut pas confondre l'indice d'un élément d'un tableau avec son contenu**.

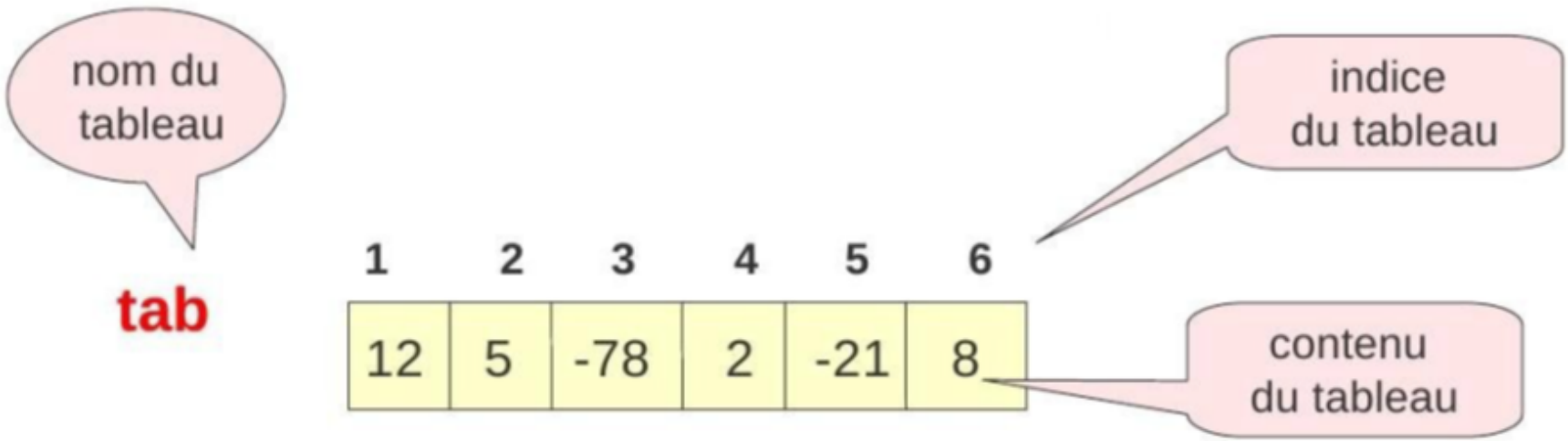
  Exemple : notes[2] :=16 ;

Le type tableau

Exemple de déclaration de tableau

- **Exemple**

  var tab : array[1..6] of integer ;



tab[1] vaut 12.

tab[4] vaut 2.

La taille du tableau **tab** est de 6.

Le type tableau

Déclaration de type tableau

- Il est possible et même préférable de définir un **type tableau** et de l'utiliser pour déclarer des variables de ce type.

type

tabEntiers = array[1..10] of integer ;

tabReels = array[1..5] of real ;

var

t1,t2 : tabEntiers ;

t3,t4 : tabReels ;

Le type tableau

Déclaration d'un tableau à 2 dimensions

- C'est un tableau ayant plusieurs **lignes** et plusieurs **colonnes**.
- **Syntaxe**

  var <nom_tab> : **array** [iminL**..**imaxL] **of array** [iminC**..**imaxC] **of** <type> ; ou

  var <nom_tab> : **array** [iminL**..**imaxL**,**iminC**..**imaxC] **of** <type> ;

  - <nom_tab> est le **nom** du tableau ;
  - <iminL> et <imaxL> sont les **indices** des lignes du tableau ;
  - <iminC> et <imaxC> sont les **indices** des colonnes ;
  - <type> est le **type des éléments** du tableau.
- Exemple

  var matrice : array[1..2] of array[1..7] of integer ;
- NB : On peut généraliser à des tableaux à N dimensions.

Le type tableau

Accès aux éléments du tableau

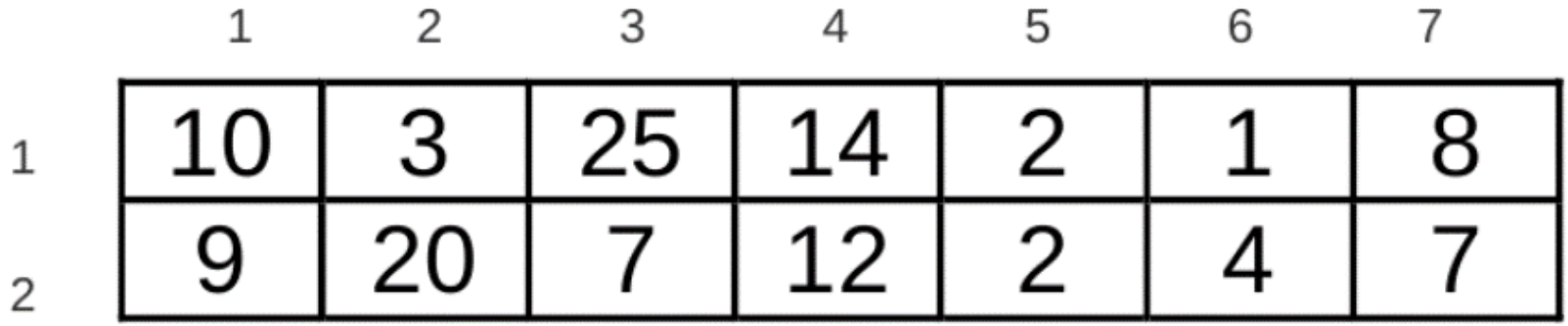
- Chaque élément d'un tableau à une dimension est accessible via son indice : **nom_tab**[**indice**].
- Chaque élément d'un tableau à 2 dimensions est accessible via ses indices : **nom_tab**[**ind1**][**ind2**] ou **nom_tab**[**ind1**,**ind2**].
- On peut accéder à un élément du tableau pour l'initialiser, le modifier ou l'afficher.
- Le **parcours** d'un tableau se fait en utilisant une boucle, **for** généralement.

Le type tableau

Accès aux éléments du tableau

- **Exemple**

  var M : array[1..2,1..7] of integer ;



	1	2	3	4	5	6	7
1	10	3	25	14	2	1	8
2	9	20	7	12	2	4	7

tableau à 2 lignes et 7 colonnes

M[1][3] vaut 25.

M[2,6] vaut 4.

M[1][7] vaut 8.

Parcours de tableau

Parcours d'un tableau à une dimension

```
1 program initialisation_tableau1;
2 var
3   i: integer;
4   tab : array[1..10] of integer;
5 begin
6   for i:=1 to 10 do
7     tab[i]:=i*i;
8 end.
```

```
1 program affichage_tableau1;
2 var
3   i: integer;
4   tab : array[1..10] of integer;
5 begin
6   for i:=1 to 10 do
7     write(tab[i],' ');
8 end.
```

Parcours de tableau

Parcours d'un tableau à 2 dimensions

```
1  program affichage_tableau2d;
2  var
3    i, j: integer;
4    M : array[1..2,1..7] of integer;
5  begin
6    for i:=1 to 2 do
7      begin
8        for j:=1 to 7 do
9          write(M[i][j],' '); {ou M[i,j]}
10       writeln();
11     end;
12 end.
```

Parcours de tableau

Exercices d'application

- Exercice 1

  Écrire un programme qui permet à l'utilisateur de saisir les valeurs d'un tableau de 10 entiers.

Parcours de tableau

Exercices d'application

- Exercice 1

  Écrire un programme qui permet à l'utilisateur de saisir les valeurs d'un tableau de 10 entiers.

- Correction

```
1  program saisie_tableau;
2  var
3    i : integer;
4    t : array[1..10] of integer;
5  begin
6    for i:=1 to 10 do
7      begin
8        write('Donner t[',i,']: ');
9        readln(t[i]);
10     end;
11 end.
```

Parcours de tableau

Exercices d'application

- Exercice 2
  Écrire un programme qui parcourt un tableau de 10 réels et détermine son maximum.

Parcours de tableau

Exercices d'application

- Exercice 2
  Écrire un programme qui parcourt un tableau de 10 réels et détermine son maximum.
- Correction

```
1  program max_tableau;
2  var
3    i : integer;
4    max : real;
5    t : array[1..10] of real;
6  begin
7    max:= t[1];
8    for i:=2 to 10 do
9      if(t[i]>max) then
10       max:= t[i];
11   write('Le maximum du tableau est ', max);
12 end.
```

Parcours de tableau

Exercices d'application

- Exercice 3

  Écrire un programme qui vérifie si un élément est dans un tableau de 10 réels.

Parcours de tableau

Exercices d'application

- Correction de l'exercice 3

```
1  program cherche_element;
2  var
3    i : integer; elt : real; trouve : boolean;
4    t : array[1..10] of real;
5  begin
6    i:=1; trouve:= false; elt:=12;
7    while (i<=10) and (trouve=false) do
8      begin
9        if(elt=t[i]) then trouve:=true;
10       i:=i+1;
11     end;
12   if(trouve=true) then
13    write('L''élément ',elt,' est dans le tableau')
14   else
15    write('L''élément n''est pas dans le tableau');
16 end.
```

Plan

Chaînes de caractères

- Le type **chaîne de caractères** est un type structuré communément utilisé en programmation pour représenter une suite de caractères.
- Une chaine de caractères peut être considérée comme un **tableau de caractères**.
- Des opérations particulières sont applicables à ce type : **concaténation**, **extraction de sous-chaînes**, **longueur**.
- On peut aussi **comparer** des chaînes avec les **opérateurs relationnels** (ordre alphabétique).

Chaînes de caractères

Déclaration de variables de type chaîne de caractères

- En Pascal, une chaîne de caractères est désignée par le type **String**.
- Syntaxes de déclaration :

  var <nom_variable> : String ;{longueur max. par défaut 255}

  var <nom_variable> : String[10] ;{longueur maximale}
- Exemples

```
1 var
2   s1, s2 : String;
3   s3, s4 : String[20];
```

Chaînes de caractères

Affectation de valeurs de type chaîne de caractères

- Lorsque qu'une valeur est affectée à une variable de type String, elle doit être mise entre 2 côtes (apostrophes).
- Exemples

```
1 s1:='Bonjour';
2 s2:='Merci, au revoir';
```

- Si la valeur à affecter contient une apostrophe, celle-ci doit être doublée.
- Exemple

```
1 s3:='Bienvenue à l''UASZ';
2 s4:='Je m''appelle';
```

- NB : On peut affecter un caractère à une variable de type chaîne de caractères, mais pas l'inverse.

Opérations sur les chaînes de caractères

Comparaison de chaînes de caractères

- On peut **comparer** des chaînes en utilisant opérateurs relationnels : =, <, >, <=, >=, <>.

Exemples

'bonjour'<'monsieur' vaut true

'bon'<'bien' vaut false

'Monsieur'='monsieur' vaut false

'Monsieur'<>'Madame' vaut true

Opérations sur les chaînes de caractères

Concaténation de chaînes de caractères

- La **concaténation** de deux chaînes de caractères permet de juxtaposer les deux chaînes pour former une nouvelle chaîne.
- En Pascal, cette opération peut être réalisée avec la fonction **concat** ou l'opérateur +.
- Exemples

```
1 s:=concat('Bien', 'venue'); {s vaut 'Bienvenue'}
2 s1:=concat('Bonjour',' Monsieur');
3 s2:='Bienvenue' + ' Madame la directrice';
4 s3:=s1 + s3;
```

Opérations sur les chaînes de caractères

Longueur d'une chaîne de caractères

- La **longueur** d'une chaîne de caractères est le nombre de caractères la composant.
- La chaine de longueur nulle correspond à la **chaîne vide**.
- En Pascal, c'est la fonction **length** qui détermine la longueur d'une chaîne de caractères.
- Exemples

```
1 length('Bonjour'); {renvoie la valeur 7}
2 length(''); {renvoie la valeur 0}
3 length('Bonjour Monsieur'); {renvoie la valeur 16}
```

Opérations sur les chaînes de caractères

Indice de caractères d'une chaîne

- Les caractères d'une chaîne sont indexés à partir de 1.
- Le premier caractère a pour indice 1, le second a pour indice 2, etc.
- On peut accéder à n'importe quel caractère d'une chaîne.
- Exemples

```
1 s:='Bonjour';
2 s[3]; {s[3]='n'}
3 s[6]; {s[6]='u'}
```

- Une chaine de caractères peur être modifiée via ses indices.
- Exemples

```
1 s:='Bonjoux';
2 s[7]:='r'; {s devient 'Bonjour'}
```

Opérations sur les chaînes de caractères

Recherche et extraction de sous chaînes

- Recherche d'une sous-chaîne dans une chaîne :

  pos(ss,s) qui renvoie la position de la la première occurrence de **ss** dans **s** et 0 si elle n'existe pas.

```
1 s:='Bonjour Monsieur';
2 pos('Monsieur', s); {renvoie 9}
3 pos('Madame', s); {renvoie 0}
```

- Extraction d'une sous-chaîne dans une chaîne :

  copy(s,i,n) qui renvoie une sous-chaîne de **s** commençant par la position **i** et ayant **n** caractères.

```
1 s:='Bonjour Monsieur';
2 copy(s,9,3); {renvoie 'Mon'}
3 copy(s,17,3); {renvoie la chaîne vide}
```