Institut National des sciences Appliquées et des Technologies

Algorithmique et Structures de Données 1
Niveau MPI

Année universitaire
2019-2020

Dr. Aymen Sellaouti
Dr. Majdi Jribi

Chapitre 7
Les enregistrements

Plan

- Partie 1: Les enregistrements en algorithmique
- Partie 2: Traduction en langage C

Plan

- Partie 1: Les enregistrements en algorithmique
- Partie 2: Traduction en langage C

Définition

Contrairement aux tableaux qui sont des structures de données dont tous les éléments sont de même type, les enregistrements sont des structures de données dont les éléments peuvent être de types différents.

Les éléments qui composent un enregistrement sont appelés champs.

Les enregistrements sont aussi appelés structures, en analogie avec le langage C.

Syntaxe

Syntaxes :

(notation inspirée du Pascal)

Type

nom_type **= enregistrement**

nom_champ1: *type_champ1*

...

nom_champn: *type_champn*

finenreg

(notation inspirée du C)

ou

Structure *nom_type*

nom_champ1: *type_champ1*

...

nom_champN: *type_champN*

FinStruct

Exemple:

Type

Tpersonne=**enregistrement**

nom: chaine[20]

prenom: chaine[20]

age: entier

finenreg

ou

Type

Structure Tpersonne

nom: chaine[20]

prenom: chaine[20]

age: entier

finStruct

Déclaration

Type

Structure tpersonne
nom: chaine[20]
âge: entier
finStruct

Syntaxe

Var

nom_var : *nom_type*

Exemple:

Var

pers1, pers2, pers3 : tpersonne

Accès aux champs

Accès aux champs d'un enregistrement

nom_enregistrement . nom_champ
représente la valeur mémorisée dans le champ de l'enregistrement

Par exemple, pour accéder à l'âge de la variable pers2, on utilise l'expression:

pers2.âge

Imbrication d'enregistrements

Un type structure peut être utilise comme type pour des champs d'un autre type de structure.

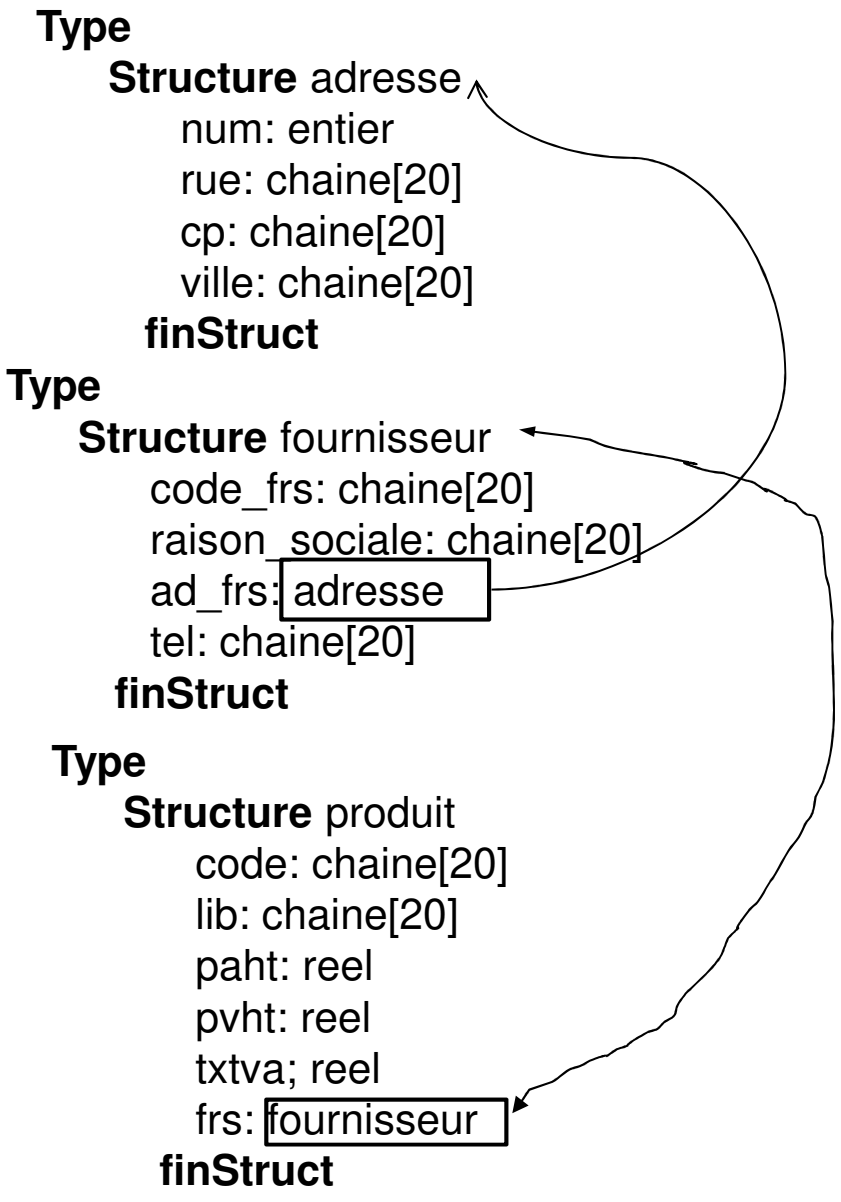
```
Type
    Structure date
        jour: entier
        mois: chaine[20]
        année: entier
     finStruct

   Type
        Structure personne
            nom: chaine[20]
            ddn: date
         finStruct
```

Pour accéder à l'année de naissance d'une personne, il faut utiliser deux fois l'opérateur '.'

pers1.ddn.année

Exemple

Un produit est livré par un seul fournisseur. Un fournisseur est caractérisé par son code, sa raison sociale, son adresse et son numéro de téléphone.



Exemple

Var
p: produit

Voilà l'instruction qui permet d'afficher le numéro de téléphone du fournisseur du produit p.frs.tel

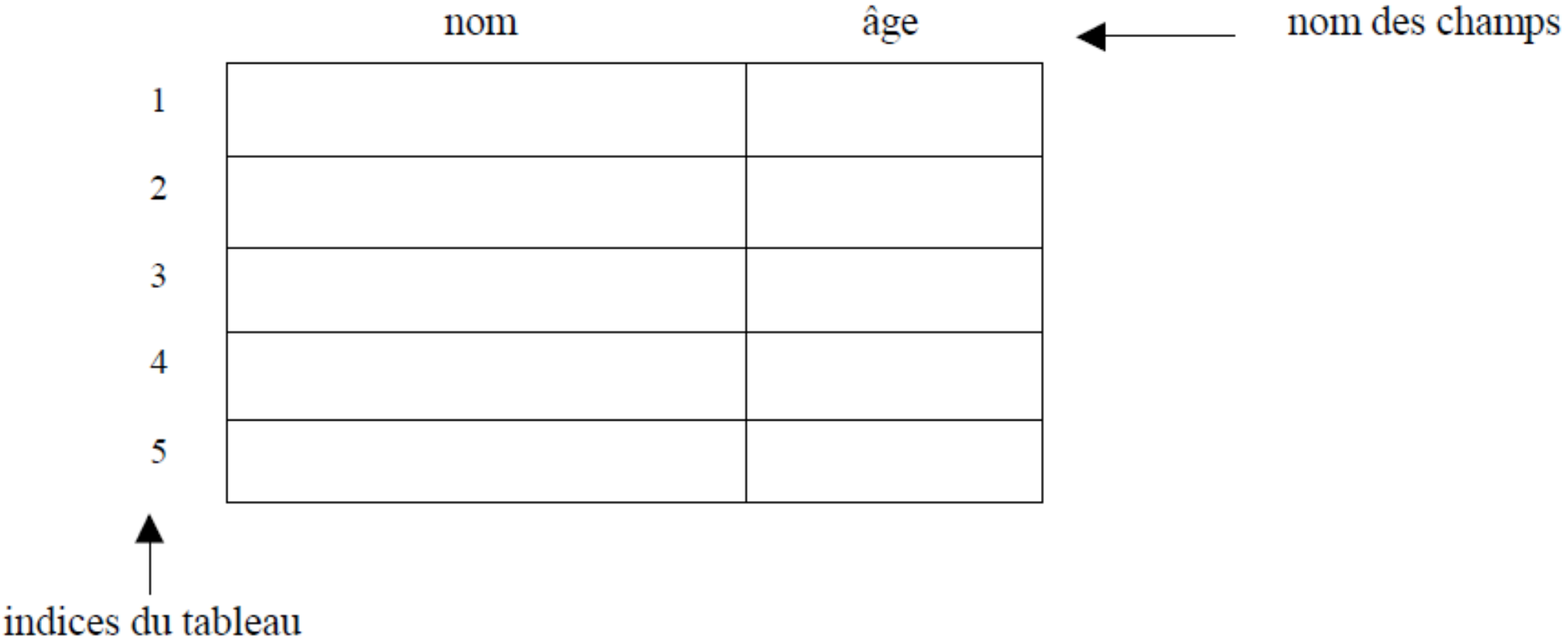
Ecrire("téléphone du fournisseur de ", p.lib, " : ", p.frs.tel **)**

Les tableaux d'enregistrements


```
Const
NP = 20                    // nombre de personnes du groupe

  Type
      Structure personnes
          nom: chaine[20]
          âge: entier
       finStruct

Var
groupe:  tableau[1..NP] de personnes
```

Les tableaux d'enregistrements

groupe[2] représente la deuxième personne du groupe
groupe[2].nom représente le nom de la deuxième personne du groupe



Plan

- Partie 1: Les enregistrements en algorithmique
- Partie 2: Traduction en langage C

Syntaxe et déclaration

Il y a deux définitions possibles

```
struct Nom_structure
       {
          <type1> var1;
          <type2> var2;
          <typeN> varN;
       };

// Définition d'une variable :
struct Nom_structure Variable;
```

```
typedef struct
        {
          <type1> var1;
          <type2> var2;
          <typeN> varN;
        } Nom_structure;

// Définition d'une variable :
Nom_structure Variable;
```

Exemple

Un livre est caractérisé par les données suivantes :
un titre
un code
un auteur
un éditeur
un prix
On désire définir un type ouvrage tel que chaque élément de ce type soit caractérisé par ces données.

En C :

```
struct ouvrage
{
   char titre [20] ;
   int code ;
   char auteur [30] ;
   char editeur [20] ;
   float prix ;
}
```

Déclaration et accès

A partir de cette définition, on pourra définir des variables de type ouvrage, la déclaration se fait ainsi :

struct ouvrage l ;
la variable l est composée de :

char [20]	Int	char [30]	char [20]	float

Pour accéder à un champ d'une variable de type ouvrage :
nomvar.nomchamp . Ainsi,
l.titre est une variable de type chaîne de caractères.
l.code est une variable de type entier.

Exercices

Exercice 1 :

Ecrire une fonction qui permet d'afficher un livre.

Exercices

Exercice 1 :

Ecrire une fonction qui permet d'afficher un livre.

```
void affich_livre (struct ouvrage l)
{
   puts (l.titre) ;
   printf("%d",l.code) ;
   puts (l.auteur) ;
   puts (l.editeur) ;
   printf ("%f",l.prix) ;
}
```

Exercices

Exercice 2 :

Ecrire une fonction qui permet de saisir un livre au clavier. Il y a 2 solutions :

Exercices

Exercice 2 :

Ecrire une fonction qui permet de saisir un livre au clavier. Il y a 2 solutions :

On retourne l'ouvrage à la fin de la fonction

```
struct ouvrage lecture1 ()
{
   struct ouvrage l ;
   gets (l.titre) ;
   scanf ("%d",&l.code) ;
   gets (l.auteur) ;
   gets (l.editeur) ;
   scanf ("%f",&l.prix) ;
   return l ;
}
```

On passe l'ouvrage à la fonction par adresse

```
void lecture2 (struct ouvrage *l)
{
   gets (l→titre) ;
   scanf ("%d",&(l→code)) ;
   gets (l→auteur) ;
   gets (l→editeur) ;
   scanf ("%f",&(l→prix)) ;
}
```

➔l'opérateur → permet d'accéder aux champs d'une variable contenant l'adresse d'un enregistrement.

Exercices


```
void main()
{
   struct ouvrage l1, l2 ;
   l1 = lecture1 () ;
   lecture2 (&l2) ;
   afich_livre(l1) ;
   afich_livre(l2) ;
}
```

On peut déclarer un tableau de type ouvrage :

struct ouvrage livre [10] ;

livre est un tableau de 10 éléments de type struct ouvrage.

Pour accéder à un champ de l'élément d'indice i : livre [i].champ

Exercices

Exercice 3 :

Ecrire une fonction qui permet de saisir un tableau de livres au clavier.

Exercices

Exercice 3 :

Ecrire une fonction qui permet de saisir un tableau de livres au clavier.

```
void lecture_tab (int *n, struct ouvrage livre[ ])
{
   int i ;
   do
   {
         printf (« Donner le nombre d'ouvrages ») ;
         scanf ("%d",n) ;
   }while ((*n <= 0) || (*n > 10)) ;
   for (i = 0 ; i < *n ; i ++)
         lecture2 (livre+i) ;
}
```

Exercices


```
 typedef struct
{
   int jour ;
  char mois [12] ;
   int annee ;
}date ;
```

```
 typedef struct
{
   char titre [20] ;
   int code ;
   char auteur [30] ;
   char editeur [20] ;
   float prix ;
   date date_edition ;
}ouvrage ;
```

Un ouvrage est :

titre	code	auteur	editeur	prix	Date edition		
					jour	mois	Année

Exercices

Exercice 4 :

Ecrire une fonction qui permet de saisir un ouvrage au clavier selon la structure précédente.

```
void lecture3 (ouvrage *l)
{
   gets (l→titre) ;
   scanf ("%d",&l→code) ;
   gets (l→auteur) ;
   gets (l→editeur) ;
   scanf ("%f",&l→prix) ;
   scanf("%d",&l→date_edition.jour) ;
   gets(l→date_edition.mois) ;
   scanf("%d",&l→date_edition.annee) ;
}
```

Exercices

Exercice 5 :

Ecrire une fonction qui permet d'afficher un livre selon la structure précédente.

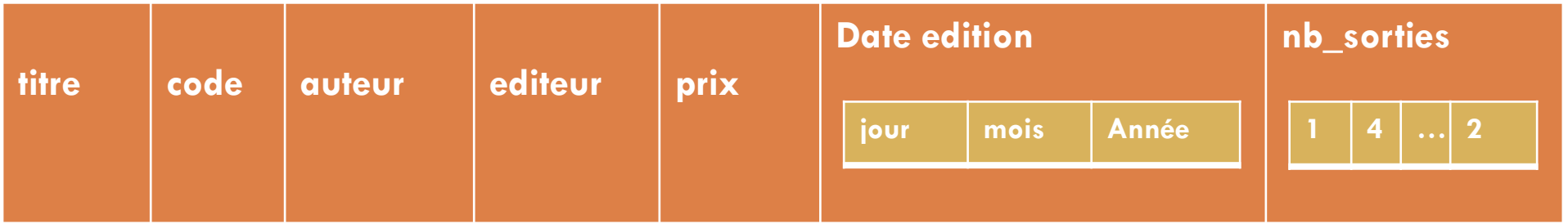
```
void Affich_livre2 (ouvrage l)
{
   puts (l.titre) ;
   printf ("%d",l.code) ;
   puts (l.auteur) ;
   puts (l.editeur) ;
   printf ("%f",l.prix) ;
   printf ("%d",l.date_edition.jour) ;
   puts (l.date_edition.mois);
   printf ("%d",l.date_edition.annee) ;
}
```

Exercices

Si on veut ajouter un tableau indiquant le nombre de sorties du livre pour chaque jour du mois :

```
typedef struct
{
   char titre [20] ;
   int code ;
   char auteur [30] ;
   char editeur [20] ;
   float prix ;
   date date_edition ;
   int nb_sorties[31] ;
}ouvrage ;
```

Un ouvrage est :



Exercices

Exercice 6 :

Ecrire une fonction qui permet d'afficher un livre selon la structure précédente.

```
void Affich_livre3 (ouvrage l)
{
   puts (l.titre) ;
   printf ("%d",l.code) ;
   puts (l.auteur) ;
   puts (l.editeur) ;
   printf ("%f",l.prix) ;
   printf ("%d",l.date_edition.jour) ;
   puts (l.date_edition.mois);
   printf ("%d",l.date_edition.annee) ;
   for (i = 0; i <31; i ++)
          printf ("%d",l.nb_sorties[i]);
}
```