Chapitre 2 : Les enregistrements

Hassen NAKBI ::: hassen.nakbi@gmail.com

1^{ere} année Licence

12 février 2024

Introduction

Structure de donnée hétérogène

Un tableau est une structure qui permet de mémoriser un ensemble d'informations du même type dans un espace mémoire contigu. Cependant, si l'on souhaite stocker des données de types différents au sein d'une même structure, par exemple les informations relatives à divers produits, une nouvelle structure est nécessaire pour représenter ces données hétérogènes et les regrouper.

2 Enregistrement

- 2.1 Définition
- 2.2 Propriétés
- 2.3 Définition algorithmique et programmation

Enregistrement

2.1 Définition

Un enregistrement est une structure de données en algorithmique qui permet de regrouper plusieurs variables de types différents sous un même nom. Il constitue un moyen de définir un nouveau type de données personnalisé.

2 Enregistrement

- 2.1 Définition
- 2.2 Propriétés
- 2.3 Définition algorithmique et programmation

Enregistrement

2.2 Propriétés

- **Nouveau type** : Il permet de définir un nouveau type de données personnalisé.
- **Regroupement de Variables** : Un enregistrement permet de regrouper des variables de types différents sous un seul nom.
- **Accès aux membres** : Les variables regroupées dans un enregistrement sont appelées membres ou champs.
- **Type des membres** : Le type de donnée d'un membre peut être simple ou structuré.
- **Enregistrement imbriqué** : Un champ membre peut être une structure d'enregistrement imbriquée.

2 Enregistrement

- 2.1 Définition
- 2.2 Propriétés
- 2.3 Définition algorithmique et programmation

Définition algorithmique

Algorithme principal Enregistrement

```
TYPE NomType = Structure
        nomchamp1 : type1
        nomchamp2 : type2
        nomchampN : typeN
Fin structure

VAR V1 : NomType
DÉBUT
V1.nomchamp1 ⟵ valeur1
V1.nomchamp2 ⟵ valeur2
V1.nomchampN ⟵ valeurN
FIN
```

Définition en Langage C

Définition d'une structure en C

```
typedef struct {
        type1 nomchamp1 ;
        type2 nomchamp2 ;
        typeN nomchampN ;
} nomType ;

nomType V1 ;
int main( ) {
V1.nomchamp1 = valeur1 ;
V1.nomchamp2 = valeur2 ;
V1.nomchampN = valeurN ;
return 0 ;}
```

Exemple en algorithmique

On voudrait implémenter un nouveau type d'enregistrement nommé 'produit' contenant les champs 'nom', 'quantité' et 'prix', pouvant être initialisés avec des valeurs arbitraires.

```
TYPE produit = structure
        nom : chaîne
        quant : entier
        prix : réel
Fin structure
VAR P1 :produit
DÉBUT
copie(P1.nom,"Ordinateur")
P1.quant ⟵ 5
P1.prix ⟵ 5700,567
FIN
```

Enregistrement en Langage C

Exemple

```
typedef struct {
        char nom[20] ;
        short quant ;
        float prix ;
} produit ;

produit P1 ;
int main( ) {
strcpy(P1.nom,"Ordinateur") ;
P1.quant = 5 ;
P1.prix = 5700,567 ;
return 0 ;}
```

Tableau d'enregistrements

3.1 Définition

Un tableau d'enregistrements permet de stocker et de manipuler un ensemble d'enregistrements de même type. Chaque élément du tableau est un enregistrement contenant plusieurs champs, regroupant ainsi des données liées dans une seule structure cohérente.

Algorithme principal TableauEnregistrement

```
TYPE produit = structure
        nom : chaîne
        quant : entier
        prix : réel
Fin structure
TYPE TAB = tableau[1..N]des produits
VAR TP : TAB
DÉBUT
Pour i de 1 à N pas ⟵1 faire
Lire(TP[i].nom)
Lire(TP[i].quant)
Lire(TP[i].prix)
FinPour
FIN
```

Tableau d'enregistrements

Tableau d'enregistrements en langage C

```
typedef struct {
        char nom[20];
        short quant;
        float prix;
} produit;
produit TP[N];
int main( ) {
short i;
for(i=0;i<N;i++) {
scanf("%s",TP[i].nom);
scanf("%hi",&TP[i].quant);
scanf("%f",&TP[i].prix);}
return 0;}
```

4 Pointeur sur les enregistrements

- 4.1 Définition
- 4.2 Accès aux champs
- 4.3 Définition algorithmique et programmation
- 4.4 Paramètre d'une procédure ou fonction

Pointeur sur les enregistrements

4.1 Définition

Un pointeur d'enregistrement pointe vers l'adresse de la première variable d'un enregistrement structuré. Cela permet d'accéder aux différents champs de l'enregistrement.
Les pointeurs d'enregistrement sont souvent utilisés dans des structures de données complexes telles que les listes chaînées, les arbres, ou les graphes.

Accès aux champs d'un pointeur

Accès aux champs d'un pointeur

L'accès aux champs d'un pointeur d'enregistrement se réalise par adresse en utilisant la notation fléchée, qui permet d'associer le nom d'un pointeur avec son nom champ membre.

pointeur$^\wedge$.nomchamp $\Leftrightarrow$ pointeur$\rightarrow$nomchamp

Algorithme principal pointeur

```
TYPE produit = structure
        nom : chaîne
        quant : entier
        prix : réel
Fin structure
VAR PE :^produit
DÉBUT
PE ⟵ allouer(3*produit)
Pour i de 0 à 2 pas ⟵1 faire
Lire((PE+i)⟶nom)
Lire((PE+i)⟶quant)
Lire((PE+i)⟶prix)
FinPour
FIN
```

Programmation en langage C

```
extbftypedef struct {
        char nom[20] ;
        short quant ;
        float prix ;
} produit ;
produit *PE = NULL ;
int main( ) {
short i ;
PE = malloc(3*sizeof(produit)) ;
for(i=0 ;i<N ;i++) {
scanf("%s",(PE+i)⟶nom) ;
scanf("%hi",&(PE+i)⟶quant) ;
scanf("%f",&(PE+i)⟶prix) ;}
return 0 ;}
```

Paramètre d'une procédure ou fonction

Paramètre d'une procédure ou fonction

Le passage de paramètre d'enregistrement par adresse ou par valeur représente deux approches distinctes pour transmettre des données à une fonction ou une procédure.

- Le passage par valeur représente une copie de l'enregistrement, alors toute modification apportée à l'intérieur de la fonction n'affectera pas l'enregistrement du programme principal.
- Le passage par adresse représente l'enregistrement d'origine, alors toute modification apportée à l'intérieur d'une fonction ou procédure affectera directement l'enregistrement du programme principal.

Paramètre d'une procédure ou fonction

```
TYPE produit = structure
        nom : chaîne
        quant : entier
        prix : réel
Fin structure
VAR PE :produit
Procédure remplir(P :^produit)
Début
Écrire("Nom : ") Lire(P⟶nom)
Écrire("Quantité : ") Lire(P⟶quant)
Écrire("Prix : ") Lire(P⟶prix)
Fin
```

Paramètre d'une procédure ou fonction

```
TYPE produit = structure
        nom : chaîne
        quant : entier
        prix : réel
Fin structure
VAR PE :produit
Procédure affichage(P :produit)
Début
Écrire("Nom : ",P.nom)
Écrire("Quantité : ",P.quant)
Écrire("Prix : ",P.prix)
Fin
```

Paramètre d'une procédure ou fonction

```
TYPE produit = structure
        nom : chaîne
        quant : entier
        prix : réel
Fin structure
VAR PE :^produit
Procédure réserver(P :^^produit, N :entier)
Début
P^ ⟵ allouer(N*produit)
Fin
```

Paramètre d'une procédure ou fonction

```
Procédure remplir(P :^produit, N :entier)
VAR i :entier
Début
Pour i de 0 à N-1 pas ←1 faire
Écrire("Nom : ") Lire((P+i)→nom)
Écrire("Quantité : ") Lire((P+i)→quant)
Écrire("Prix : ") Lire((P+i)→prix)
FinPour
Fin
```

Paramètre d'une procédure ou fonction

```
Procédure affichage(P :^produit, N :entier)
VAR i :entier
Début
Pour i de 0 à N-1 pas ←1 faire
Écrire("Nom : ",(P+i)⟶nom)
Écrire("Quantité : ",(P+i)⟶quant)
Écrire("Prix : ",(P+i)⟶prix)
FinPour
Fin
```