

Langage C

Types composés

François Cuvelier

Laboratoire d'Analyse Géométrie et Applications
Institut Galilée
Université Paris XIII.

22 septembre 2011 à 09:23

Plan

- 1 Les tableaux
 - Les tableaux monodimensionnels
 - Les tableaux multidimensionnels
- 2 Les structures

Déclaration

Syntaxe

```
<type> <NomTableau> [<NbElements>];
```

- <type> : type prédéfini (**char**, **int**, **double**, ...)
- <NomTableau> : un identificateur nommant le tableau,
- <NbElements> : **constante** correspondant aux nombres d'éléments du tableau.

Remarque

Le nombre d'éléments d'un tableau doit impérativement être connu lors de la compilation.

Déclaration

Exemples

```
1 | #define CMAX 80
2 | const int N=15;
3 | int tA [5];           /* tA est un tableau de 5 entiers
4 | double tF [N];      /* tF est un tableau de N doubles
5 | char tC [CMAX];    /* tC tableau de CMAX-1 caracteres
```

Déclaration avec initialisation

Syntaxe

```
<type> <NomTableau> [<N>] = { <const1>, ..., <constN> };
```

ou

```
char <NomTableau> [<N>] = <ChaineCaracteres>
```

- **<const1> à <constN> : constantes de type <type>.**
- **<ChaineCaracteres> : chaîne de caractères (suite de caractères délimitée par des guillemets)**

Déclaration avec initialisation

Exemples

```
1 | int tA [5]={0 , 1 , 2 , 3 , 4};  
2 | int tB []={0 , 1 , 2 , 3 , 4};  
3 | int tC [5]={0 , 1 , 2};  
4 | char tt1 []= "abcd";  
5 | char tt2 []={ 'a' , 'b' , 'c' , 'd' };  
6 | char tt3 [80]="exemple";
```

opérateur []

L'accès à un élément d'un tableau par utilisation de l'opérateur d'indilage []

Exemple

```
1  #include <stdio.h>
2
3  int main(){
4      int tE[]={4,1,3,0,2};
5      int i,N;
6      N=sizeof(tE)/sizeof(int);
7      for (i=0;i<N;i++)
8          printf ("_element_%d_de_tE:_%d\n",i,tE[i]);
9      return 1;
10 }
```

Execution

```
element 0 de tE : 4  
element 1 de tE : 1  
element 2 de tE : 3  
element 3 de tE : 0  
element 4 de tE : 2
```

Exemple

```
1  #include <stdio.h>
2
3  main(){
4      char tH[]="exemples";
5      int i,N;
6      N=sizeof(tH)/sizeof(char);
7      for (i=0;i<N;i++)
8          printf("tH[%d]=_%c_ou_%d\n",i,tH[i],tH[i]);
9  }
```

Execution

```
tH[0] = e ou 101
tH[1] = x ou 120
tH[2] = e ou 101
tH[3] = m ou 109
tH[4] = p ou 112
tH[5] = l ou 108
tH[6] = e ou 101
tH[7] = s ou 115
tH[8] = ou 0
```

Plan

- 1 Les tableaux
 - Les tableaux monodimensionnels
 - Les tableaux multidimensionnels

- 2 Les structures

1 Les tableaux

- Les tableaux monodimensionnels
- **Les tableaux multidimensionnels**

2 Les structures

Déclaration

Syntaxe

```
<type> <NomTableau> [<N1>] [<N2>] ... [<Nk>];
```

- <type> : type prédéfini (**char**, **int**, **double**, ...)
- <NomTableau> : un identificateur nommant le tableau,
- <N1> à <Nk> : **constantes**.

Ici <NomTableau> est un tableau de <N1> tableau de <N2> tableau de ... tableau de <Nk> <type>.

Déclaration

Exemples

```
1  #define DIM1 80
2  #define DIM2 60
3
4  int tA2D [DIM1][DIM2];
5  /* tA2D est un tableau de 80 tableaux de 60 int */
6
7  typedef int tab1D [DIM2];
8  /* tab1D est le type tableau de 60 int */
9  tab1D tB2D [DIM1];
10 /* tB2D est un tableau de 80 tab1D */
```

Déclaration avec initialisation

L'initialisation d'un tableau s'effectue à l'aide d'une liste d'éléments délimités par des accolades.

Si un élément est un tableau, on applique récursivement cette notation. Lorsqu'une liste est incomplète, elle est implicitement complétée par des 0.

Exemples

```
1  int tC2D[3][4] = { {1, 2, 0, 0},
2                      {4, 2, 2, 0},
3                      {1, 1, 1, 1}
4                      };
5  int tE2D[3][4] = { {1, 2},
6                      {4, 2, 2},
7                      {1, 1, 1, 1}
8                      };
```

Exemple

```
1 #include <stdio.h>
2 #define DIM1 2
3 #define DIM2 5
4 main(){
5     int tG2D [ DIM1 ][ DIM2 ]={{4 , 1 , 3 , 0 , 2} , {7 , 8}};
6     int i , j ;
7     for ( i=0 ; i < DIM1 ; i ++ )
8         for ( j=0 ; j < DIM2 ; j ++ )
9             printf ( "tG2D [%d] [%d]=%d\n" , i , j , tG2D [ i ][ j ] ) ;
10 }
```

Execution

```
tG2D [0] [0] =4
```

```
tG2D [0] [1] =1
```

```
tG2D [0] [2] =3
```

```
tG2D [0] [3] =0
```

```
tG2D [0] [4] =2
```

```
tG2D [1] [0] =7
```

```
tG2D [1] [1] =8
```

```
tG2D [1] [2] =0
```

```
tG2D [1] [3] =0
```

```
tG2D [1] [4] =0
```

1 Les tableaux

- Les tableaux monodimensionnels
- Les tableaux multidimensionnels

2 Les structures

Déclaration d'un modèle de structure

Une structure est un ensemble d'objets de types éventuellement différents.

Syntaxe

```
struct <ModeleStructure> {  
    <type-1> <identifiant-1>;  
    ...  
    <type-k> <identifiant-k>;  
};
```

- `<ModeleStructure>` : nom du modèle de structure,
- `<identifiant-1>` à `<identifiant-k>` : identificateur des différents *membres* ou *champs* de la structure ,
- `<type-1>` à `<type-k>` : type associé à chaque identificateur.

Déclaration d'un modèle de structure

Exemple

```
1 | struct etudiant{  
2 |     char Nom[80];  
3 |     char Prenom[80];  
4 |     int Age;  
5 | };
```

Déclaration d'un objet de type structure

Syntaxe

```
struct <ModeleStructure> <objet>;
```

- <ModeleStructure> : nom d'un modèle de structure existant,
- <objet> : identifiant de l'objet de type structure.

L'opérateur "." permet l'accès aux différents champs d'une structure.

Exemple 1

```
1  #include <stdio.h>
2  #include <string.h>
3  struct etudiant{
4      char Nom[80];
5      char Prenom[80];
6      int  Age;
7  };
8
9  main(){
10     struct etudiant E,F;
11     strcpy(E.Nom,"Azerty");
12     strcpy(E.Prenom,"Qwerty");
13     E.Age=23;  F=E; strcpy(F.Prenom,"Qwertybis");
14     printf("Nom_: _%s, _Prenom_: _%s, _Age_: _%d\n",
15           E.Nom,E.Prenom,E.Age);
16     printf("Nom_: _%s, _Prenom_: _%s, _Age_: _%d\n",
17           F.Nom,F.Prenom,F.Age);
18 }
```

Execution

```
Nom : Azerty, Prenom : Qwerty, Age : 23
```

```
Nom : Azerty, Prenom : Qwertybis, Age : 23
```

Exemple 1 avec typedef

```
1  #include <stdio.h>
2  #include <string.h>
3  struct etudiant{
4      char Nom[80];
5      char Prenom[80];
6      int  Age;
7  };
8
9  typedef struct etudiant Etudiant;
10
11 main(){
12     Etudiant E={"Azerty","Qwerty",23},F;
13     F=E; strcpy (F.Prenom,"Qwertybis");
14     printf ("Nom_: _%s, _Prenom_: _%s, _Age_: _%d\n",
15            E.Nom,E.Prenom,E.Age);
16     printf ("Nom_: _%s, _Prenom_: _%s, _Age_: _%d\n",
17            F.Nom,F.Prenom,F.Age);
18 }
```