E2C / **A3P**(AL)
2026/2027

Cours C3

© Denis BUREAU, ESIEE Paris

Sommaire

1. Rappels sur les pointeurs
2. Rappels sur les tableaux
3. Notion de structure
4. Type structure
5. Pointeur de structure et notation
6. Tableaux dans une structure
7. Tableau de structures
8. Problèmes de saisie

1.1. Rappels sur les pointeurs

- Une variable pointeur contient une adresse
- Une variable pointeur se déclare en mettant une *** après** le type que l'on veut pointer
- L'adresse d'une variable se calcule en plaçant **& devant** le nom de la variable
- La variable pointée par un pointeur s'obtient en plaçant *** devant** le nom du pointeur
- Un pointeur d'entier est différent d'un pointeur de réel ou de caractère

1.2. Rappels sur les pointeurs

- `double* vPtr;`
  // `vPtR` est une variable pointeur de `double`
  // donc pourra contenir une adresse de `double`
- `double vR = 1.2;  vPtr = &vR;`
  // `vPtR` contient l'adresse de `vR`
  // ou `vPtR` pointe sur `vR`
- `*vPtr` désigne ce qui est pointé par `vPtr`
  donc la variable réelle `vR` qui vaut 1.2
  `*vPtr=0.0;` // la remet à 0 comme `vR=0.0;`

2. Rappels sur les tableaux

- Le nom d'un tableau est équivalent à l'adresse de son premier élément :
  `double vTab[10];`
  `vTab == &vTab[0]`
- Plus généralement : `vTab[i] == *(vTab+i)`
  Attention ! `vTab+1` ajoute 8 et non 1, pourquoi ?
- Pourtant, `double vTab1[10];`
  est très différent de `double *vTab2;`
  On peut écrire : `vTab2 = vTab1;`
  mais pas l'inverse ! Pourquoi ?

3. Notion de structure

- ```
  struct {
     Type1 champ1;
     Type2 champ2;
     Type3 champ3;  // int
  } vS1, vS2;
  ```

- `vS1.champ1 = valeur1;`
- `vS2.champ1 = vS1.champ1;`
- `scanf( "%d", &(vS1.champ3) );`
- `scanf( "%d", &vS1.champ3 );`

  (le . est très prioritaire ☺)
  ```

4.1. Type structure

- ```
  struct {
     Type1 champ1;
     Type2 champ2;
     Type3 champ3;
  } vS1; // dans le main
  ```
- ```
  struct {
     Type1 champ1;
     Type2 champ2;
     Type3 champ3;
  } vS2; // dans une procédure
  ```
- `vS1` et `vS2` sont de deux types incompatibles !

4.2. Type structure

- ```
  typedef struct {
      Type1 champ1;
      Type2 champ2;
      Type3 champ3;
  } MonNouveauType;
  ```

- `MonNouveauType vS1, vS2;`
- `MonNouveauType vS3 = {v1,v2,v3};`
- `vS1 = {v1,v2,v3}; //` interdit

- Dans les paramètres, retours de fonction, etc.
  ```

5.1. Pointeur de structure

- ```
  MonNouveauType vS1, vS2;
  MonNouveauType* vPS1;
  ```

- ```
  MonNouveauType* vPS1, vPS2;
  MonNouveauType *vPS1, *vPS2;
  ```
  ou, encore mieux :
  ```
  typedef MonNouveauType* TypePointeur;
  TypePointeur vPS1, vPS2;
  ```

- `vPS1 = &vS1;`
- `vPS2 = &vS2;`

5.2. Nouvelle notation

- `*vPS1.champ1` // pb de priorité

(le `.` est très prioritaire ☹)

- `(*vPS1).champ1` // ok
  `vS1.champ1`

- `vPS1->champ1`
  // équivalent, mais plus clair

QUIZZ

- Soit `vP` un pointeur de structure
  et `chp` un champ de cette structure,

- Quelle(s) notation(s) sont correctes pour désigner le champ à partir du pointeur ?

A **vP.chp**	**D** **(*vP).chp**
B ***vP.chp**	**C** **vP->chp**

6. Tableau dans structure

```
struct {
   char chaine[20];
   int code;
} vS, *vPS;
vPS = &vS;
```

```
vS.chaine[15] = '\0';
printf("%s\n",vPS->chaine)';
// ..
```

7. Tableau de structures

- `MonNouveautype vTab[10];`
- `vTab[0].champ3`
- `vTab[i].champ2`

- `vTab->champ3`
- `(vTab+i)->champ2`

- Si `champ2` est une chaîne de car. :
  `vTab[i].champ2[0]`

8.1. Problèmes de saisie

- ```
  printf( "numerateur ? " );
  scanf( "%d", &vNum );
  printf( "denominateur ? " );
  scanf( "%d", &vDen );
  printf( "(%d/%d)", vNum, vDen );
  ```
  ➔
- numerateur ? **3 Entrée**
  denominateur ? **16 Entrée**
  (3/16)
  ```

8.2. Problèmes de saisie

- ```
  printf( "1er car.? " );
  scanf( "%c", &car1 );
  printf( "2ème car.? " );
  scanf( "%c", &car2 );
  printf( "(%c/%c)", car1, car2 );
  ```
  ➔
- ```
  1er car.? A Entrée
  2ème car.? (A/
  )
  ```
- `car2` vaut `'\n'` !

8.3. Problèmes de saisie

- ```
  printf( "1er car.? " );
  scanf( "%c", &car1 );
  scanf( "%c", &inutile );
  printf( "2ème car.? " );
  scanf( "%c", &car2 );
  scanf( "%c", &inutile );
  printf( "(%c/%c)", car1, car2 );
  ```
  ➔
- 1er car.? **A Entrée**
  2ème car.? **B Entrée**
  (A/B)
  ```

8.4. Problèmes de saisie

- ```
  typedef char Chaine10[10];
  Chaine10 avant="precieux", mot, reste;
  printf( "mot ? " );
  scanf( "%s", mot );
  scanf( "%s", reste );
  printf( "(%s)", mot );
  printf(" tresor %s\n", avant);
  ```



- mot ? **Programme principal**
  (Programme) tresor precieux
- mot ? **Programmation**
  (Programmation) tresor ion
  ```

8.5. Problèmes de saisie

- ```
  typedef char Chaine10[10];
  Chaine10 avant="precieux", mot, reste;
  printf( "mot ? " );
  fgets( mot, 10, stdin );
  fgets( reste, 10, stdin );
  printf( "(%s/%s)", mot, reste );
  printf(" tresor %s\n", avant);
  ```

  ➔ 

- mot ? **La programmation**
  (La progra/mmation
) tresor precieux
  ```