Chapitre 9 : LES POINTEURS

⇨ **PROBLEMATIQUE** :

A la déclaration d'une variable, un emplacement lui est réservé dans la mémoire (RAM).

Jusqu'ici, cette réservation d'espace mémoire se fait avant l'exécution d'un programme. L'espace réservé ne peut être libéré pendant l'exécution du programme (d'où l'appellation de VARIABLES STATIQUES) :

- ***gaspillage de mémoire pour les variables inutilisées.* (1)**
- ***Impossible d'avoir d'espace supplémentaire en plein exécution.* (2)**

La mémoire centrale est formée d'un ensemble de cellules qui sont identifiées par des adresses (adresse mémoire). Ainsi, à la déclaration, chaque identificateur de variable sera rattaché avec une adresse (l'adresse de son espace de réservation).

***Jusqu'ici, pour accéder à une valeur stockée en mémoire centrale, on passe directement par le nom de la variable.* (3)**

(1) & (2) : Comment faire pour avoir des ***VARIABLES DYNAMIQUES*** ?
(3) : Comment faire pour accéder à une valeur stockée en mémoire en passant son ***ADRESSE*** ?

⇨ **SOLUTION** :

Utilisation des **POINTEURS**.

I. DEFINITIONS

Par opposition aux variables statiques, les pointeurs permettent une meilleure gestion de la mémoire centrale : on peut demander au système d'exploitation ***d'attribuer*** un espace supplémentaire ou de ***libérer*** un espace en plein exécution du programme. L'allocation d'espace mémoire d'un pointeur ne se fait pas lors de la déclaration mais plutôt lors de l'exécution du programme.

Un pointeur est une variable qui contient l'adresse mémoire d'une autre variable. La variable dont l'adresse est stockée dans le pointeur est appelé variable pointée : « le pointeur ***pointe*** sur la variable pointée ».

Les adresses sont généralement des valeurs « **hexadécimale** » constantes mais pour un pointeur, sa valeur peut changer : la variable pointée ne se déplace pas en mémoire mais c'est le pointeur qui pointe sur une autre variable.

II. DECLARATION DE POINTEUR

layegaye001@gmail.com

Pour savoir ce qui est pointé par un pointeur, les pointeurs disposent d'un type. Lors de la déclaration d'un pointeur, on ne sait pas sur quel objet (variable) il pointe mais on sait le type d'objet (variable) pointé. Par convention le nom d'un pointeur commence par la lettre « p ».

Un pointeur lors de sa déclaration ne possède pas d'adresse mémoire : *sa référence est à **NIL***. Il doit être initialisé avant d'être utilisé dans un programme.

A. DECLARATION D'UN POINTEUR DE TYPE ELEMENTAIRE

⇨ **Syntaxe 1 :**

VAR nomPointeur : ↑Type

⇨ **Syntaxe 2 :**

TYPE nomType : ↑Type

VAR nomPointeur : nomType

⇨ **Exemples :**

Syntaxe 1	Syntaxe 2	Remarques
VAR **P** : ↑entier	TYPE Pentier : ↑entier VAR **P** : Pentier	*Le contenu de P doit être une adresse mémoire d'une cellule où l'on doit placer un entier.*
VAR **P** : ↑réel	TYPE Preel : ↑réel VAR **P** : Preel	*Le contenu de P doit être une adresse mémoire d'une cellule où l'on doit placer un réel.*

B. DECLARATION DE POINTEUR D'UN TYPE ENREGISTREMENT

⇨ **Syntaxe 1 :**

Type nomEnregistrement = Structure
Debut
Champ(s) : type(s)
Fin

Type nomPointeur = ↑nomEnregistrement
VAR nomVariablePointeur : nomPointeur

OU BIEN

Var nomVariablePointeur : ↑nomEnregistrement

⇨ **Syntaxe 2 :**

Type nomPointeur = ↑Structure
Debut
Champs1 : type1

layegaye001@gmail.com

ChampsN : type
Fin
VAR nomVariablePointeur : nomPointeur

⇨ **Exemples :**

Syntaxe 1	Syntaxe 2
Type Personne = structure *Début* *Nom, Prenom : chaine* *Age : entier* *Fin* *VAR PP : ↑ Personne*	*Type Personne = ↑structure* *Début* *Nom, Prenom : chaine* *Age : entier* *Fin* *VAR PP: Personne*

III. INITIALISATION D'UN POINTEUR

Avant l'utilisation d'un pointeur, il faudra l'initialiser. Pour demander au système d'exploitation d'allouer de l'espace à un pointeur, il existe deux moyens :

⇨ Utilisation de module de gestion des allocations dynamique au système d'exploitation (MGAD),
⇨ Utilisation d'un autre pointeur déjà initialisé.

a. Initialisation par le MGAD :

```
VAR    nomVariablePointeur : ↑entier
ALLOUER (nomVariablePointeur)
SI (nomVariablePointeur  = NIL) alors
   Afficher (« Pas d'espace mémoire libre »)
SINON
  Afficher (« Allocation réussie »)
FINSI
```

b. Initialisation par un autre pointeur déjà initialisé :

```
VAR nomVariablePointeur1, nomVariablePointeur2 : ↑entier
ALLOUER (nomVariablePointeur1)
SI (nomVariablePointeur1  = NIL) alors
   Afficher (« Pas d'espace mémoire libre »)
SINON
  nomVariablePointeur2 ⟵ nomVariabblePoiteur1  //
initialisation
FINSI
```

REMARQUE :

*Il faut toujours initialiser un pointeur. En effet, un pointeur non initialisé (**pointeur pendant**) est un pointeur qui se pointe n'importe où. Si on tente d'affecter une valeur*

layegaye001@gmail.com

*dans l'emplacement mémoire de son adresse, on risque d'écraser une zone mémoire quelconque d'où le risque de faire planter le programme. Pour faire pointer un pointeur nulle part, on l'initialise à NIL (**Not Identified Link**).*

IV. MANIPULATION D'UN POINTEUR

Manipuler un pointeur revient à lui donner une adresse, à lui reprendre l'adresse, à extraire (**Déréférencer un pointeur**) la valeur de la variable pointée ou à mettre une valeur dans l'emplacement de son adresse (dans la variable pointée). Dans tous les cas, avant de manipuler un pointeur, il faudra que l'allocation soit réussie.

Si **P** est un pointeur quelconque déjà initialisé, alors :

- Le contenu de P est une adresse mémoire.
- La notation **P** représente l'adresse.
- La notation ↑**P** représente la valeur stockée dans l'adresse **P**.

Exemple :

Ecrire un programme qui permet de déclarer deux pointeurs d'entiers. Le programme permet de saisir une valeur pour le premier pointeur puis affecte au second pointeur trois fois la valeur du premier. Le programme affiche l'adresse mémoire de chaque pointeur puis leur valeur.
1. Représenter le résultat sous forme de schéma avec 8 comme valeur saisie par l'utilisateur.
2. Si après les deux affichages on remet le deuxième pointeur à NIL, que se passe-t-il ? Représenter le schéma.
3. Représenter le résultat après LIBERER (premier pointeur).
4. Représenter le résultat des instructions suivantes par un schéma :

- *var P1, P2 : ↑entier*
- *Allouer(P1)*
- *P1↑← 8*
- *P2 ←P1*

5. Puis après l'instruction : libérer (P2).

REMARQUE :

- *L'espace mémoire alloué à un pointeur peut être libéré à tout moment en utilisant la fonction* ***LIBERER (nomVariablePointeur).***
- *Après initialisation d'un pointeur, il reçoit NIL alors le pointeur vers l'adresse est rompu (l'adresse stockée dans le pointeur est supprimée) mais la valeur dans la mémoire centrale n'est pas perdue mais reste inaccessible.*
- *Pour supprimer la valeur et l'adresse, il faut utiliser la fonction LIBERER.*
- *Un pointeur ne peut recevoir une valeur mais uniquement une adresse (en autre pointeur par exemple).*

layegaye001@gmail.com

- *Une valeur ne peut pas recevoir une adresse mais uniquement une valeur (valeur de la variable pointée par exemple).*

layegaye001@gmail.com