TP : LES POINTEURS

Généralités sur les pointeurs

- Les opérateurs de base

Lors du travail avec des pointeurs, nous avons besoin

- d'un opérateur 'adresse de': & pour obtenir l'adresse d'une variable.

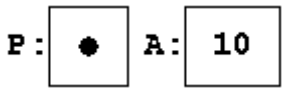
- d'un opérateur 'contenu de': * pour accéder au contenu d'une adresse.

- d'une syntaxe de déclaration pour pouvoir déclarer un pointeur.

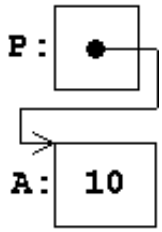
L'opérateur 'adresse de' : **`&<NomVariable> fournit l'adresse de la variable <NomVariable>`**.

- Représentation schématique

Soit P un pointeur non initialisé et A une variable (du même type) contenant la valeur 10 :



Alors l'instruction **`P = &A`** affecte l'adresse de la variable A à la variable P. Nous pouvons illustrer le fait que 'P pointe sur A' par une flèche:



L'opérateur 'contenu' : **`*<NomVariable> fournit le contenu de la variable <NomVariable>`**.

- Déclaration d'un pointeur

`<Type> *<NomPointeur>`
`déclare un pointeur <NomPointeur> qui peut recevoir des adresses de variables du type <Type>`

Une déclaration comme

```
int  *PNUM;
```

peut être interprétée comme suit: **"*PNUM est du type int" ou "PNUM est un pointeur sur int" ou "PNUM peut contenir l'adresse d'une variable du type int"**

EX1 :

Complétez le tableau suivant :

	A	B	C	P1	P2
Init.	1	2	3	/	/
P1=&A	1	2	3	&A	/
P2=&C					
*P1=(*P2)++					
P1=P2					
P2=&B					
*P1-=*P2					
++*P2					
P1=*P2					
A=++*P2**P1					
P1=&A					
*P2=*P1/=*P2					

Tableaux et pointeurs

Le nom d'un tableau représente l'adresse de son premier élément. En d'autres termes:

&tableau[0] et **tableau**

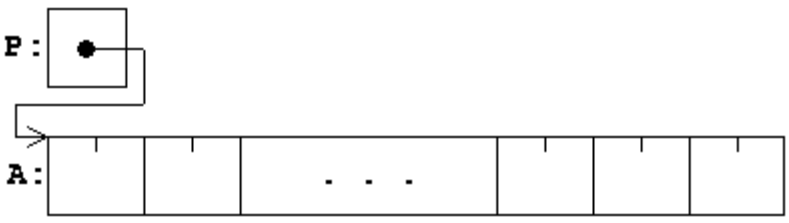
sont une seule et même adresse.

Exemple

En déclarant un tableau A de type int et un pointeur P sur int,

```
int A[10];
int *P;
```

l'instruction: **P = A;** est équivalente à **P = &A[0];**



Ainsi, après l'instruction, **`P = A;`** le pointeur P pointe sur A[0],

`*(P+1)` désigne le contenu de A[1].

`*(P+2)` désigne le contenu de A[2].

`...` **`...`**

`*(P+i)` désigne le contenu de A[i].

EX2 :

Soit P un pointeur qui 'pointe' sur un tableau A:

```
int A[] = {12, 23, 34, 45, 56, 67, 78, 89, 90};
int *P;
P = A;
```

Quelles valeurs ou adresses fournissent ces expressions:

```
a)      *P+2
b)      *(P+2)
c)      &P+1
d)      &A[4]-3
e)      A+3
f)      &A[7]-P
g)      P+(*P-10)
h)      *(P+*(P+8)-A[7])
```

EX3 :

Nous voulons écrire un programme qui, étant donné un tableau d'entiers déjà initialisé, demande à l'utilisateur quel entier chercher et affiche ensuite le nombre d'occurrences de cet entier dans le tableau.

1. Écrire le programme en utilisant l'opérateur [].

2. Écrire le programme en utilisant explicitement les pointeurs pour accéder aux éléments.

3. Ecrire une fonction **`void somme_moyenne(int* tab, int* somme, double* moyenne)`** qui sauvegarde la somme et la moyenne des éléments du tableau.