Objectif :

- Etudier les notions suivantes:
  - La notion de pointeurs,
  - L'arithmétique des pointeurs,
  - Le passage d'arguments par adresse.

I - Exercice n°1.

I.1 - Travail demandé.

Compléter le programme suivant, de telle sorte que le contenu de la variable **c** soit affiché à l'écran, par l'intermédiaire de la variable pointeur **pt_x**.

Programme partiel :

```
char  c = 'A' ;
char  *pt_x ;          /* pt_x est un pointeur sur un caractere */
```

II - Exercice n°2.

II.1 - Travail demandé.

Compléter le programme suivant, de telle sorte que le contenu de la variable **a** soit divisé par 5, par l'intermédiaire de la variable pointeur **b**. Bien évidemment, il n'est pas question de manipuler directement la variable **a**.

Programme partiel :

```
float  a ;
float  *b ;          /*  b est un pointeur sur un reel  */
a = 10 ;
```

III - Exercice n°3.

III.1 - Travail demandé.

On considère le programme partiel ci-dessous. Définissez le contenu de la variable i après exécution de chacune des trois dernières instructions de ce programme.

Programme partiel:

```
int i = 15 ;
int *ptr ;           //  ptr est un pointeur sur un entier
ptr = &i ;           // Le pointeur ptr pointe sur la variable i
*ptr = *ptr + 10 ;
*ptr++ ;
--*ptr ;
```

IV - Exercice n°4.

Soit le programme partiel suivant :

```
int x = 1 ;
int y = 2 ;
int *p ;
int **pp ;
p = &x ;
x = x + y ;
pp = &p ;
```

IV.1 - Travail demandé.

1) Définir ce que représente la variable pp.
2) Définir le contenu de : x, *p, **pp, p et pp.

V - Exercice n°5

V.1 - Travail demandé.

Soit le programme partiel ci-dessous.

Programme partiel :

```
int  A[ ] = {9, 15, 26, 28, 33, 45, 46 , 51, 52, 67} ;
int *ptr ;
ptr = A ;
```

Déterminer les valeurs ou adresses que fournissent les expressions suivantes :

a) ptr
b) ptr + 1
c) *ptr+2
d) *(ptr + 2)
e) & A[4] – 3
f) A + 3
g) & A[7] – ptr
h) *(ptr + *(ptr+8) – A[7])

VI - Exercice n°6

VI.1 - Travail demandé.

On considère le tableau A comportant les valeurs suivantes {-3, 4, 0, -7, 3, 8, 0, -1, 4, -9}.

Ecrire un programme qui transfère le contenu du tableau A dans un second tableau B en inversant l'ordre des éléments. En fin d'exécution, on obtient B = {-9, 4, -1, 0, 8, 3, -7, 0, 4, -3}.

Pour cet exercice, on utilisera **exclusivement** des pointeurs pour parcourir les tableaux et réaliser les affectations des éléments.

VII - Exercice n°7

VII.1 - Travail demandé.

On souhaite écrire une fonction qui place des 1 sur la diagonale d'un tableau bi-dimentionnel de dimension N quelconque.

Ecrire le code de cette fonction en utilisant les pointeurs pour parcourir le tableau.

Le prototype de la fonction diag est : void diag(int *p, int n) ;

VIII - Exercice n°8

VIII.1 - Travail demandé.

Soit le tableau d'entier Tab tel que : Tab[10] = {9, -1, 4, -8, 7, 15, 0, -3, 11, -5} ;

On souhaite programmer une fonction qui ne renvoie rien et qui détermine l'élément minimum et l'élément maximum du tableau Tab.

Les valeurs du minimum et du maximum du tableau doivent être affichées par le programme principal.

Le prototype de la fonction MinMax est : void MinMax (int *p, int *Min, int *Max) ;

L'appel de la fonction MinMax est : MinMax(Tab, &ValMini, &ValMaxi) ;