Auteur : Mohamed Messabihi
Matière : Programmation et structures de données
Mohamed.Messabihi@gmail.com

Université Abou Bakr Belkaïd - Tlemcen
Faculté des sciences
1ère L.M.D. MI

5 TP : Les pointeurs

5.1 Reconstitution d'un programme

Écrire un programme C qui permet de déclarer les variables a, b et p et montre, après son exécution, l'affichage ci-dessous :

```
 Reconstitution d'un programme utilisant la notion de pointeur
------------------------------------------------------------------------
 a = 10, b = 5 sont deux variables entières et *p est un entier
 &a est l'adresse de a --> 0028FF1C
 &b est l'adresse de b --> 0028FF18
 p stocke l'adresse de a --> 0028FF1C
 *p stocke la valeur de a --> 10
 &p est l'adresse de p --> 0028FF14
 b stocke le double de la valeur de *p --> 20
 p pointe maintenant sur b --> 0028FF18
 *p stocke la valeur de b --> 20
```

NB. Les adresses indiquées sur l'affichage ci-dessus sont données à titre indicatif et peuvent changer d'une machine à une autre.

5.2 Toto & Loulou

Montrez l'historique d'exécution détaillé ainsi que l'affichage du programme ci-dessous :

```c
#include <stdio.h>
int Toto (int *a, int *b){
    (*a)++;
    ++(*b);
    return *a+*b ;
}
void Loulou (int *a, int *b, int *c){
    printf ("Avant Toto : *a=%d, *b=%d, *c=%d\n", *a, *b, *c);
    *c = Toto(a, b);
    printf ("Apres Toto : *a=%d, *b=%d, *c=%d\n", *a, *b, *c);
    (*a)++;
    ++(*b);
}
void main(){
    int a=3, b=4, c=4 ;
    Loulou (&a, &b, &c) ;
    printf ("Apres Loulou ; a=%d, b=%d, c=%d\n", a, b, c);
}
```

En déduire le rôle de la fonction toto() et celui de la fonction loulou().

5.3 Traces

Remplacez les commentaires dans le programme suivant par les instructions correspondantes puis exécutez-le.

```
1 #include <stdio.h>
2 void Premiere(int aval)
3 {
4  int a;
5  a = aval;
6  /* Afficher la valeur et l'adresse de a ici */
7 }
8 void Deuxieme(int bval)
9 {
10  int b;
11  /* Afficher la valeur et l'adresse de b ici */
12 }
13 void main()
14 {
15  Premiere(3);
16  Deuxieme(8);
17  Deuxieme(17);
18 }
```

Que remarquez-vous ? Expliquez ce qui s'est passé.

5.4 L'arithmétique avec des pointeurs

Montrez l'historique d'exécution détaillé ainsi que l'affichage du programme ci-dessous :

```
1 void main(void){
2     int A = 1, B = 3, C = 5;
3     int *P1 = &A, *P2= &B, *P3 = &C;
4     printf("*P1=%d, *P2=%d, *P3=%d\n", *P1, *P2, *P3);
5     *P2 = ++(*P3);
6     printf("A=%d, B=%d, C=%d\n", A, B, C);
7     *P2 = (*P3)++;
8     printf("*P1=%d, *P2=%d, *P3=%d\n", *P1, *P2, *P3);
9     printf("P1=%d, P2=%d, P3=%d\n", P1, P2, P3);
10    *P2 = 10-*P3++;
11    printf("*P1=%d, *P2=%d, *P3=%d\n", *P1, *P2, *P3);
12    printf("P1=%p, P2=%p, P3=%p\n", P1, P2, P3);
13    *P3 = *P2**P3;
14    printf("*P1=%d, *P2=%d, *P3=%d\n", *P1, *P2, *P3);
15    *P2 = ++*P1**P3++;
16    printf("*P1=%d, *P2=%d, *P3=%d\n", *P1, *P2, *P3);
17    printf("P1=%p, P2=%p, P3=%p\n", P1, P2, P3);
18 }
```

1. Observer et analyser bien le résultat des instructions des lignes 6, 8 et 11. En déduire la différence entre les expressions **a** = ++***P** , **a** = (***P**)++ et **a** = ***P**++.
2. Justifier la valeur de *P3 dans la ligne 12 en expliquant ce qui s'est passé à l'instruction juste d'avant ?
3. En déduire l'origine de la valeur de *P3 au niveau de la ligne 17.

5.5 Permutation par « ou exclusif »

Le langage C possède des opérateurs de calcul binaire (appelés opérateurs de modification de bits) qui traitent les nombres entiers sous leur forme binaire (base 2).

L'opérateur | effectue un « **ou** » binaire sur chaque bit du premier nombre avec le bit correspondant du second. Par exemple : 5|12 donne 13 car :

```
    00000101
  |
    00001100
  = 00001101
```

De la même manière, il existe un opérateur **&** pour le « **et** » binaire (5 & 12 = 4) , un opérateur ~ pour le « **non** » binaire (qui inverse chaque bit d'un nombre) et un opérateur ^ pour le « **ou exclusif** » binaire. Par exemple 5 ^ 12 = 9 car :

```
    00000101
  ^
    00001100
  = 00001001
```

Une propriété remarquable du « **ou exclusif** » est qu'il permet, en l'appliquant trois fois, de permuter les valeurs de deux variables entières distinctes sans en utiliser une troisième. En effet, si a et b sont deux variables entières, les instructions :

```
a = a^b;
b = a^b;
a = a^b;
```

permettent au final d'obtenir la valeur initiale de b dans a et celle de a dans b.

Exemple : si a = 5 et b = 12, alors a = a^b ; donne a = 5^12 = 9 (b reste 12) puis b = a ^b ; donne b = 9^12 = 5, puis a = a^b ; donne a = 9^5 = 12 ; au final on obtient bien a = 12 et b = 5.

1. Utilisez la méthode de la permutation par « ou exclusif » dans une fonction qui permet d'échanger les valeurs de deux variables puis testez la dans une fonction main.
2. Quel problème rencontre-t-on si l'on appelle cette fonction avec la même variable passée au deux paramètres (ou même deux pointeurs différents mais qui pointent vers la même variable) ? Modifier la fonction pour pouvoir éviter ce problème.