TD1

Exercice 1 :

b7	b6	b5	b4	b3	b2	b1	b0

Soit une variable p1 de type char (8 bits signés), écrire les expressions en C permettant de :

1. mettre à 1 le bit b2
2. mettre à 1 le bit b3 et b6
3. mettre à 0 le bit b0
4. mettre à 0 le bit b4 et b5
5. inverser le bit b3 (se fait facilement avec un ou exclusif)
6. mettre à 1 le bit b2 et à 0 le bit b0
7. mettre à 1 les bits b0 et b7 et à 0 les bits b3 et b4

Exercice 2 :

Soit une variable char nb;

Écrire les expressions permettant de calculer les centaines, les dizaines et les unités de cette variable.

Exercice 3 :

Construire des expressions booléennes sur les tests suivants, expression vraie si :

- le bit b6 est à 1
- le bit b3 est à 0
- le bit b2 est à 1 et le bit b4 est à 0
- le bit b2 est à 1 ou le bit b7 est à 0
- le bit b6 est l'inverse du bit b3 (sans utiliser de décalages)
- le bit b6 est l'inverse du bit b3 avec l'opérateur de décalage, & et ^

Exercice 4 :

Quelle opération arithmétique est réalisée par un décalage ? Évaluer pour cela les expressions suivantes (avec a=12 et b=23) :

- a = a >> 1 (ou a >>= 1) $a/2^n$
- a = a >> 2 (ou a >>= 2)
- b = b << 1 (ou b <<=1)
- b = b << 2 (ou b <<=2) $b * 2^n$

Exercice 5 :

Soit le programme suivant :

```
#include <stdio.h>
main() {
int n=10,p=5,q=10,r;
r= n == (p = q);
printf("A : n = %d p = %d q= %d r = %d\n",n,p,q,r);
n = p = q = 5;
n += p += q;
printf("B : n = %d p = %d q= %d\n",n,p,q);
q = n++<p || p++ != 3;
printf("C : n = %d p = %d q= %d\n",n,p,q);
q = ++n == 3 && ++p == 3;
printf("D : n = %d p = %d q= %d\n",n,p,q);
return 0;
}
```

Que donnera-t-il comme affichage sur l'écran ?

Exercices 6:

1. Créer une fonction en langage C qui reçoit comme paramètres un entier C1 de type char (8 bit) et la position *n* d'un bit. La fonction doit tester le bit *n* du et renvoyer 0 si la valeur de bit est nul sinon elle renvoi 1. **short test_bit (char C1, short n)**

2. Créer une fonction en langage C qui reçoit comme paramètre un entier de type char C1. La fonction doit afficher C1 en forme binaire. **void print_binary(char C1)**

3. Créer une fonction en langage C qui reçoit comme paramètres l'adresse d'un entier de type char C1 et la position n d'un bit. La fonction doit positionner le bit n du C1 à 1.

void set bit(char* C1, short n)

4. Créer une fonction en langage C qui reçoit comme paramètres l'adresse d'un entier de type char C1 et la position n d'un bit. La fonction doit positionner le bit n du C1 à 0.

void clear bit(char* C1, short n)

5. Créer une fonction en langage C qui reçoit comme paramètres un entier de type char C1 et un nombre n. La fonction doit faire une rotation à gauche les n bits de C1. Les bits du poids fort doivent être reçus à droite dans C1. Utiliser la fonction test bit.

char rotate bits(char C1, short n)

6. Dans une fonction main, déclarer et initialiser un entier de type char Al et appeler les fonctions (de 1 à 5) pour les tester.