Exercice 1:

Ecrire une fonction, appelée entier_palindrome, qui lit un entier positif et affiche 1 si l'entier est un nombre palindrome et affiche 0 sinon. Un nombre positif est un entier palindrome s'il se lit aussi bien de gauche à droite que de droite à gauche. Le nombre 35453 est un nombre palindrome; alors que le nombre 45652 ne l'est pas.

Exercice 2:

En utilisant les opérations bit à bit, écrire une fonction récursive en C qui prend en paramètre un entier positif nb et retourne 1 si le nombre passé en paramètre est une puissance de 2. La fonction retourne 0 sinon. Par exemple, si l'entier passé en paramètre est 14, alors la fonction retournera 0. Par contre, si l'entier passé en paramètre est 16, alors la fonction retournera 1

Exercice 3:

En utilisant les opérations bit à bit, écrire une fonction itérative en C qui prend en paramètre un entier positif nb et retourne 1 si le nombre passé en paramètre est une puissance de 2. La fonction retourne 0 sinon. Par exemple, si l'entier passé en paramètre est 14, alors la fonction retournera 0. Par contre, si l'entier passé en paramètre est 16, alors la fonction retournera 1

Exercice 4:

Que fait le programme suivant :

```
#include <stdio.h>
int main (void)
{
  unsigned char  i;
  unsigned short int j=256;
  for (i=0; i!=j; i++)
  {
        printf ("Le caractère numéro %d est : %c \n", i, i);
  }
return(0);
}
```

Justifier votre réponse.

Exercice 5:

Ecrire une fonction C qui prend en paramètres un chiffre c et un entier positif n et retourne le nombre de fois que le chiffre c est présent dans les nombres entre 0 et n. Par exemple, si c=7 et n=100, alors la fonction retournera 20 (car il y a 20 fois le chiffre 7 dans les nombres compris entre 0 et 100).

Exercice 6:

On souhaite écrire un programme qui réalise le chiffrement par substitution mono alphabétique. Pour chaque lettre de l'alphabet (ou caractère) on se donne une autre lettre (un autre caractère) utilisée dans le texte chiffré.

On s'intéresse à coder des lettres en majuscules, par le Chiffrement par décalage : chaque lettre est remplacée par son $k^{ème}$ voisin dans l'alphabet.

Ecrire un programme C qui :

- lit un entier positif k, qui représente la clef de décalage,
- lit un caractère qui représente une lettre majuscule m et
- affiche le caractère chiffré m' associé à m (attention le voisin de 'Z' est 'A').

Ecrire un programme C qui réalise le processus de déchiffrement (c'est-à-dire on lit k et m' et on affiche m).

Exercice 7:

Cet exercice (d'intérêt pédagogique seulement) montre que l'on peut utiliser plusieurs indirections imbriquées.

Compléter le programme suivant pour afficher la valeur de i, en utilisant uniquement la variable pointeur p4 :

```
#include <stdio.h>
int main (void)
{
    int i=-20;
    int *p1;
    int **p2;
    int ***p3;
    int ****p4;
    p1=&i;
    p2=&p1;
    p3=&p2;
    p4=&p3;
    printf ("La valeur de i est : %d.\n ", .........);
    return(0);
}
```

Exercice 8:

- Dites dans quel ordre les instructions suivantes, associées à une boucle for, sont exécutées :

```
#include <stdio.h>
int main (void)
{....
    for (instructions 1; instructions 2; instructions 3)
    {
    instuctions 4
    }
    return(0);
}
```

- Est-il possible de modifier le programme ci-dessous afin de confirmer l'ordre d'exécution des instructions de la boucle "for".

```
#include <stdio.h>
int main (void)
{
  int i;
    for (i=0; i!=5; i++)
    {
    printf ("La valeur de i est %d :\n", i);
    }
    return(0);
}
```

Indications:

- Le caractère "," (virgule) peut-être utilisé pour écrire une suite d'instructions (qui sera terminée par le ";").
- Les trois champs peuvent contenir toute suite d'instructions.
- La suite d'instructions, donnée dans le champs "instructions 2", peut elle aussi être composée de plusieurs instructions élémentaires séparées par une virgule ",". Dans ce cas là, c'est l'évaluation de la dernière instruction élémentaire qui sera utilisée pour déterminer la condition d'arrêt de la boucle "for".

Exercice 9 :

On souhaiterait réaliser des divisions réelles avec une précision (nombre de chiffres après la virgule) fixée par l'utilisateur.

- Ecrire une fonction C qui a l'entête suivante :

```
void diviser (int p, int q, int precision)
```

Elle consiste à imprimer le résultat de la division de p sur q avec une précision égale à "precision". La fonction ne retourne aucune valeur. Par exemple, la fonction appelée avec les paramètres (1, 97, 1000) imprimera:

0.0103092783505154639175257731958762886597938144329896907216494845360824742268041237
1134020618556701030927835051546391752577319587628865979381443298969072164948453608247422680412371134020618556701030927835051546391752577319587628865979381443298969072164948453608247422680412371134020618556701030927835051546391752577319587628
8659793814432989690721649484536082474226804123711340206185567010309278350515463917525
7731958762886597938144329896907216494845360824742268041237113402061855670103092783505 1
5463917525773195876288659793814432989690721649484536082474226804123711340206185567010
3092783505154639175257731958762886597938144329896907216494845360824742268041237113402 0
6185567010309278350515463917525773195876288659793814432989690721649484536082474226804 1
2371134020618556701030927835051546391752577319587628865979381443298969072164948453608 2
4742268041237113402061855670103092783505154639175257731958762886597

Remarque : La fonction division et de type void et elle ne retourne aucune valeur; ce qui signifie que l'on ne vous demande pas de retourner le résultat de la division. Par ailleurs, il n'est pas nécessaire de stocker le résultat de la division dans un tableau. Ce qui est demandé est un simple affichage du résultat de la division (et l'affichage peut se faire chiffre par chiffre). Attention le %.xf ne permettra pas de répondre de manière satisfaisante à la question.

- Tester votre fonction depuis le programme main.

Exercice 10:

Les séquences suivantes sont connues sous le nom de suites de Conway :

1
11
21
1211
111221
312211
13112221
1113213211

Chaque séquence i (avec $i >= 2$), représentée par un long entier positif, est obtenue depuis la séquence $(i-1)$ en comptant chaque occurrence des chiffres (1,2,3), puis on imprime le nombre d'occurrences et l'occurrence en question.

Par exemple, supposons que l'on part de "1211". Nous avons "une occurrence de 1", "une occurrence de 2" et "deux occurrences de 1", ce qui donne : "111221" comme séquence successeur de "1211".

Les suites de conway sont uniquement composées des chiffres 1, 2 et 3 (si le point de départ est 1).

Ecrire une fonction C qui a l'entête suivante :

```
unsigned long int suivant (unsigned long int  s)
```

Cette fonction prend en paramètre un entier positif qui représente une séquence de conway donnée et retourne la séquence suivante.