

**Exercice 1:**

Compiler et exécuter le programme suivant :

```
1 #include <stdio.h>
2 int main (void)
3 {
4     unsigned char i;
5     i=250;
6     i=i+6;
7     printf("La valeur de i est %d \n", i);
8     return(0);
9 }
```

Expliquer l'affichage obtenu.

**Exercice 2:**

Ecrire une fonction C qui :

- lit un nombre entier positif et
- affiche les deux derniers chiffres du nombre lu.

**Exercice 3:**

Cet exercice illustre le comportement de la fonction scanf lorsque la donnée lue ne correspond pas au format attendu.

Ecrire un programme C qui :

- lit un premier entier et l'affiche, puis
- lit un deuxième entier et l'affiche, et
- lit un dernier entier et l'affiche.

Tester votre programme avec les valeurs :

- 3, 5 et 19.
- A, 12 et 14.

Nous verrons plus tard des conséquences plus importantes et proposerons des solutions.

**Exercice 4:**

Considérons le programme suivant :

```
#include <stdio.h>
int main (void)
{
    printf ("\n Impression avec deux caractères de retour à la ligne \n");
    return(0);
}
```

Un étudiant, n'ayant pas la touche `\n`, a re-codé ce programme par le suivant :

```
#include <stdio.h>
int main (void)
{
    printf ("%cn Impression avec les codes ASCII ... %cn", 92, 92);
    return(0);
}
```

- A votre avis, pourquoi l'étudiant a choisi le code ASCII 92 ?
- Est-ce que son programme fonctionne correctement ? Justifier votre réponse.

## Exercice 5:

Soit  $i$  une variable de type "unsigned int".

- Compléter le tableau suivant avec  $i=14$  :

Opérations bit à bit	valeur de l'expression en binaire	valeur en décimal
$i$		
$i \ll 1$		
$i \ll 4$		
$i \gg 1$		
$i \gg 4$		
$i \& 1$		
$i \& 4$		
$1 \ll 1$		
$1 \ll 4$		

Nous supposons que la valeur de  $i$  n'est pas modifiée d'une étape à une autre.

- Ecrire un programme C qui confirme vos réponses.

## Exercice 6:

La bibliothèque <limits.h> contient une liste de constantes qui donnent les domaines des différents types de variables entières. Parmi ces constantes, on trouve :

CHAR_BIT	Nombre de bits mot
SCHAR_MIN	Valeur minimale pour signed char.
SCHAR_MAX	Valeur maximale pour signed char.
UCHAR_MAX	Valeur maximale pour unsigned char.
CHAR_MIN	Valeur minimale pour char.
CHAR_MAX	Valeur maximale pour char.
SHRT_MIN	Valeur minimale pour short int.
SHRT_MAX	Valeur maximale pour short int.
USHRT_MAX	Valeur maximale pour unsigned short int.
INT_MIN	Valeur minimale pour int.
INT_MAX	Valeur maximale pour int.
UINT_MAX	Valeur maximale pour unsigned int.
LONG_MIN	Valeur minimale pour long int.
LONG_MAX	Valeur maximale pour long int.
ULONG_MAX	Valeur maximale pour unsigned long int.

Ecrire un programme C qui affiche la valeur de ces constantes.

## Exercice 7:

Le but de cet exercice est d'expliquer que l'emplacement de l'opérateur "cast" est important. Exécuter les programmes suivants :

```
1. #include <stdio.h>
   int main (void)
   {
       int i=5, j=2;
       float f;
       f=4*(i/j);
       printf("La valeur de f est : %.f. \n", f);
       return(0);
   }
```

```

2. #include <stdio.h>
   int main (void)
   {
       int i=5, j=2;
       float f;
       f=4*(float)(i/j);
       printf("La valeur de f, après une application globale de l'opérateur cast, est : %f. \n", f);
       return(0);
   }

3. #include <stdio.h>
   int main (void)
   {
       int i=5, j=2;
       float f;
       f=4*((float)i/(float)j);
       printf("La valeur de f, après des applications locaux de l'opérateur cast, est : %f. \n", f);
       return(0);
   }

```

## Exercice 8:

Grâce à la fonction pré-définie "sizeof", écrire un programme C qui affiche la taille des différents types de variables vus en cours (utiliser aussi les séquences "\t" et "\n" pour avoir un bon affichage).

## Exercice 9:

Un étudiant a écrit le programme C ci-dessous. Il s'est rendu compte qu'il avait fait 8 erreurs. Proposer une correction de ces erreurs. Compiler et tester votre programme.

```

/****
  Les huit erreurs de compilation
  ***/
#include <stdia.h>
int main (void)
{
    int b==1,c;
    const int f=0;
    printf ("Merci de saisir un premier nombre. \n");
    scanf ("%c", &b);
    printf ("Merci de saisir un deuxieme nombre. \n");
    scanf ("%d", c);
    a=++(b+c);
    f=(a>1);
    if (f)
        printf ("La somme des deux nombres lus est strictement positive. \n");
    return(0)
}

```

## Exercice 10:

Ecrire un programme qui permet de traiter le problème des n-reines. Ce problème consiste à placer n reines sur un échiquier n x n tel qu'aucune reine ne puisse attaquer un autre. Nous rappelons qu'une reine se déplace en horizontal, en diagonal et en vertical.