Rappel :

Le premier contrôle du langage C aura lieu le lundi 20 octobre 2025.

Exercice 1:

Ecrire une fonction récursive qui réalise la fonction de Hanoï vue en cours. Tester votre fonction avec n=2, n=3, n=10, n=50 (Hum!)

Exercice 2:

1. Ecrire une fonction itérative (non récursive) qui calcule a^b où a et b sont des entiers.
2. Ecrire une fonction récursive qui calcule a^b où a et b sont des entiers en utilisant les propriété suivantes :
   - Si $b = 0$ alors $a^b = 1$
   - Si b est pair alors $a^b = a^{(b/2)} * a^{(b/2)}$
   - Si b est impair alors $a^b = a * a^{(b-1)}$

Exercice 3 :

Que retourne comme valeur la fonction mystère suivante pour n=3 ? Justifier votre réponse.

```
int mystere (int n){
    int i, s=0;
    for (i=1; i<=n; i++)
        s = s + i/n;
    return s;
}
```

Exercice 4:

Considérons la fonction f suivante :

```
int f(int *a, int n)
{
    /* n est strictement positif*/
    if (n==1)
        return 1;
    if (a[0]>a[1])
        return 0;
    else
        return f(a+1, n-1);
}
```

- Que fait la fonction f ?
- Justifier votre réponse (quels sont les rôles joués par les paramètres a et n) ?
- Ecrire un programme main qui appelle cette fonction.

Exercice 5:

Considérons la séquence suivante (appelée la séquence de Hofstadter) :

$$Q(n) = Q(n - Q(n-1)) + Q(n - Q(n-2)),$$

avec

$$Q(1) = Q(2) = 1.$$

- Ecrire une fonction récursive qui prend en paramètre un entier positif n et qui retourne la valeur Q(n).
- Proposer une version efficace du calcul de Q(n) (plus efficace est compris dans le sens où la nouvelle fonction décrite dans cette question est de complexité calculatoire plus petite que celle écrite dans la première question).
- Tester votre fonction depuis le main avec différentes valeurs de n.

Exercice 6:

Un étudiant a écrit le programme suivant :

```
#include <stdio.h>
void echange_adr (unsigned int *a, unsigned int *b){
  unsigned int *c;
  *c=*a;
  *a=*b;
  *b=*c;
}

int main (void){
  void echange_adr (unsigned int *a, unsigned int *b);
  unsigned int x, y;
  unsigned int *p=&x, *p1=&y;
  x=5;
  y=10;
  echange_adr(p, p1);
  printf ("Grâce aux pointeurs, après l'appel à la
        fonction echange_adr x=%u et y=%u.\n", x,y);
  return(0);
}
```

Il pense que son programme n'est pas juste. Pouvez-vous l'aider?

Exercice 7:

On se donne une table (que l'on appellera baguenaudière) à n cases, chacune peut contenir un pion. Chaque case est numérotée de "1" à "n". La baguenaudière peut être vide ou pleine. Le but du jeu est de remplir la baguenaudière si elle est vide, ou de la vider si elle est pleine, en respectant les règles suivantes :

- Pour la case 1, on peut enlever un pion ou mettre un pion sans contrainte.
- Pour la case 2, on peut enlever un pion ou mettre un pion uniquement si la case 1 est pleine
- De manière générale, pour la case i (différente de 1 et 2), on peut enlever un pion ou mettre un pion si :
  - La case $i-1$ est pleine (contient un pion)
  - Toutes les cases de 1 à $i-2$ sont vides.

Ecrire les fonctions remplir et vider. On utilisera la récursivité croisée pour écrire vos fonctions.