ISIMM **ANNEE UNIVERSITAIRE :** 2022-2023
SECTION : ING1INFO **ENSEIGNANT :** SAKKA ROUIS TAOUFIK
MATIERE : CONCEPTION ET ANALYSE D'ALGORITHMES

TD1 (VERSION 2)

EXERCICE 1 :

Ecrire la fonction récursive Factorielle de façon terminale et non terminale.

EXERCICE 2 :

Ecrire la fonction récursive Fibonacci de façon terminale et non terminale.

EXERCICE 3:

Ecrire la fonction (en récursive terminale et non terminale) qui recherche le maximum d'un tableau et retourne l'indice de la valeur maximale.
Si le tableau est null ou de longueur 0, vous retourner -1

EXERCICE 4:

Ecrire une méthode récursive terminale permettant et une autre non terminale de retourner le nombre d'occurrences d'un élément X dans un tableau d'entiers T de taille n transmis en argument d'entrée.

EXERCICE 5 :

Ecrire une fonction récursive permettant de vérifier un entier N est pair ou non.

EXERCICE 6 :

Ecrire une fonction récursive qui permet de chercher le maximum dans un tableau T.

EXERCICE 7 :

Dérécursiver les solutions des Exercices 1, 2, 3, 4, 5 et 6

EXERCICE 8 :

Ecrire une méthode récursive qui retourne la somme des carrés des X premiers entiers.
Exemple : on prend x = 4, le résultat retournera la valeur 30.

EXERCICE 9 :

Ecrire une fonction récursive terminale et une autre non terminale permettant de calculer la somme S1 pour un entier n donnée.
S1 = 1 + 1 /2 + 1/3 + 1/4 + … + 1/n

EXERCICE 10 :

1. On se propose d'écrire une fonction qui permet de déterminer le PGCD de deux entiers positifs non nuls A et B en utilisant l'algorithme d'Euclide :
Sachant que PGCD (a, b) = PGCD (b, r), avec r = a mod b.
Tant que le reste r est non nul, on remplace a par b et b par r. Le dernier reste r non nul est alors le PGCD des deux nombres
Exemple : PGCD (32, 12) = PGCD (12, 8) = PGCD (8, 4) = PGCD (4, 0) = 4.
2. On se propose d'écrire un programme C qui permet de déterminer le PGCD en utilisant la méthode de la différence :
Tant que (a != b) on répète la recherche :

Si (a>b) alors PGCD (a, b)=PGCD (a-b, b),

Sinon PGCD (a, b) = PGCD (a, b-a)

Exemple : PGCD (10,16) = PGCD (10,6) = PGCD (4,6) = PGCD (4,2) = PGCD(2,2) = 2.
Proposer une solution récursive pour les deux méthodes.

EXERCICE 11 :

La racine carrée d'un nombre a>0 peut être estimée de façon itérative à partir de la suite suivante :

$U_0 = 1$

$U_{n+1} = 1/2 * (a/U_n + U_n)$

Pour n et a données, écrire une fonction récursive qui détermine la racine carrée de a.

EXERCICE 12 :

Ecrire une fonction récursive permettant de vérifier un entier N est premier ou non.

EXERCICE 13 : Fonction de MacCarthy

1. Ecrire une fonction récursive f qui calcule f (n) selon la formule suivante :

f(n) = n - 10 Si n > 100

f(n) = f (f (n+11)) Sinon

2. Calculer f (96).

EXERCICE 14 :

Écrire la fonction récursive terminale **multiple_nb** qui renvoie le nombre de multiples de val (passée en paramètre) contenus dans une liste chaînée de réels passé en paramètre.

La structure d'un élément de la liste chaînée est :

```
typedef struct _element
{   double  value;
    struct _element * suivant;
} element;
```

EXERCICE 15:

Écrire en C la fonction récursive **tree_nnodes** qui retourne le nombre de nœuds contenus dans un arbre. La structure d'un élément de l'arbre :

```
typedef struct _node {
        int value ;
        struct _node * left ;
        struct _node * right ;
} node;
```

EXERCICE 16: Tri par fusion d'une liste chaînée

Proposer un algorithme qui fait la concaténation de deux listes (en entrée) retournant une liste (en sortie). Par exemple, si l'on a les listes (3, 7, 2) et (5, 4) on obtient la liste (3, 7, 2, 5, 4).

Proposer ensuite un algorithme séparant une liste en deux. Etant donné une liste en entrée, un élément sur deux va dans la première liste et un élément sur deux va dans la deuxième liste.

Faire une action de fusion de listes supposées triées par ordre croissant en une troisième liste triée.

Faire, en utilisant au choix les trois actions précédentes une action de tri fusion qui étant donné deux listes en entrée (pas forcément triées) retourne une liste triée.