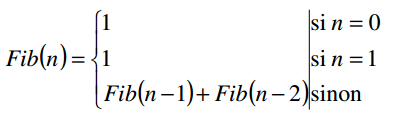
**Université IBN KHALDOUN –TIARET- Faculté Des Mathématiques et de l’informatique**

**Département d’informatique, 2ière Année Licence : -2014/2015-**

## TD N°02 : Récursivité

**Exercice 1 :**

La suite de Fibonacci est définie comme suit :



1. Ecrivez un algorithme itératif calculant Fib(n).

2. Ecrivez un algorithme récursif calculant Fib(n).

3. Calculez ça complexité en nombre d’additions.

4. Ecrire un algorithme pour une fonction récursive appelée FIB\_N\_FIB\_N\_1() qui calcule,

pour n>0, le couple (Fib(n), Fib(n-1)).

Corrigé

**Fonction FIB(n :entier) :entier**

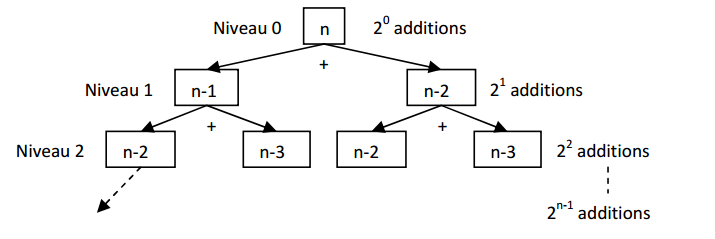
**si n=0 ou n=1 alors retourner 1**

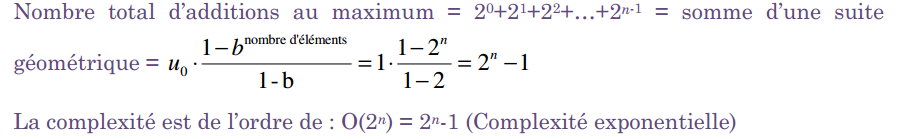
**sinon retourner FIB(n-1) + FIB(n-2)**

**(Récursivité multiple.)**

**2. Calculez ça complexité en nombre d’additions.**

**L’algorithme s’exécute sous forme d’un arbre binaire comme suit :**

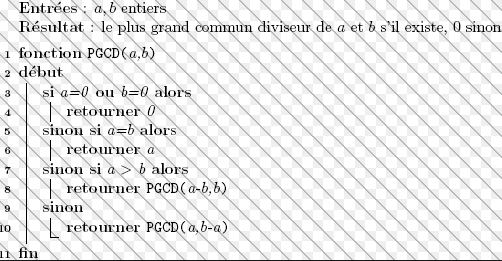




**Exercice 2 (Calcul de pgcd)**

Soit a et b des entiers. On note que pgcd(a, b) = pgcd(a − b, b).

1. Écrire un algorithme récursif permettant de calculer le pgcd.



**Exercice 3 :** Ecrireun algorithme itératif et un autre récursif qui considèrent une phrase comme un tableau de caractères et d´énumérer les éléments de ce tableau dans l’ordre inverse.

void reverse(int a[],int start,int end)

{

int temp;

temp = a[start];

a[start] = a[end];

a[end] = temp;

if(start==end ||start==end-1)

return;

reverse(a, start+1, end-1);

}

**Exercice 4 :**

Soit une fonction qui vérifie si un élément ‘x’ appartient à une partie d’un tableau ‘A’

appelée IS\_MEMBER() qui reçoit en paramètres le tableau A, l’élément ‘x’ et les indices de la partie du tableau ‘d’ et ‘f’ et retourne ‘true’ ou ‘false’.

**a.** Si l’ensemble ‘A’ n’est pas trié :

1. Ecrivez un algorithme récursif pour cette fonction ;

**IS\_MEMBER(A, x, d, f) : booleen**

**si d>f alors retourner False**

**si A[d]=x alors retourner True**

**sinon retourner IS\_MEMBER(A, x, d+1, f)**

**b.** Si l’ensemble ‘A’ est trié dans l’ordre croissant :

2. Ecrivez un algorithme récursif pour cette fonction ;

**IS\_MEMBER(A, x, d, f): ) : booleen**

**si d>f ou A[d]>x alors retourner False**

**si A[d]=x alors retourner True**

**sinon retourner IS\_MEMBER(A, x, d+1, f)**

1. Utilisez la recherche dichotomique pour améliorer votre algorithme ;

**IS\_MEMBER(A, x, d, f) :booléen**

**si d=f alors si A[d]=x alors retourner True**

**sinon retourner False**

**m←[(d+f)/2]**

**si A[m]=x alors retourner True**

**sinon si A[m]>x alors retourner IS\_MEMBER(A, x, d, m-1)**

**sinon retourner IS\_MEMBER(A, x, m+1, f)**

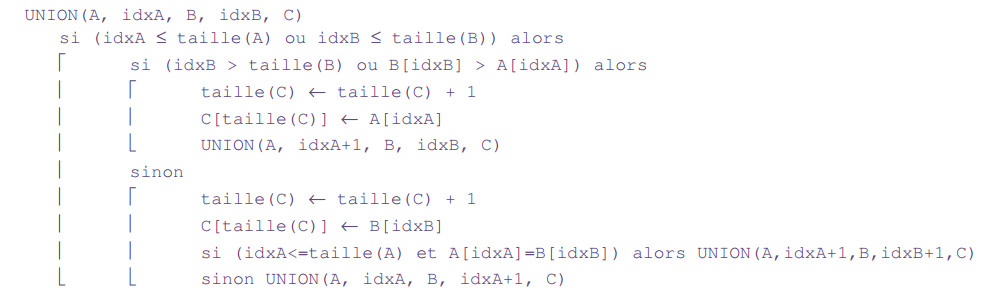
Soit les fonctions :

UNION(A,B) qui fait l’union de deux ensembles ;

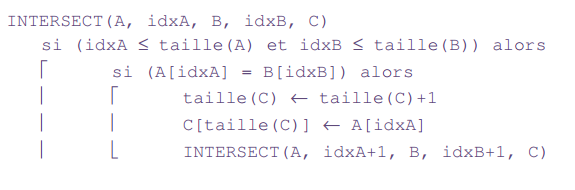
INTERSECT(A,B) qui fait l’intersection de deux ensembles ;

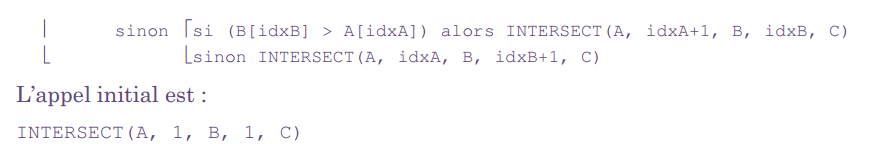
**c.** Si les ensembles ‘A’ et ‘B’ sont triés dans l’ordre croissant

4. Ecrivez des algorithmes récursifs pour ces fonctions ;









**Exercice 5 : (Maximum d’une liste)**

1. Construire une fonction max qui renvoie le maximum de deux réels.

2. Écrire un algorithme récursif maximum donnant le maximum d’une liste de nombres réels quelconques. On pourra utiliser le fait que max(a, b, c) = max(a, max(b, c))

**Exercice 6 : Le problème des tours de Hanoï**

1. Ecrire un programme C qui permette de résoudre le problème des tours de Hanoï.
2. Evaluer la complexité en nombre de déplacement des disques pour résoudre ce problème.

Corrigé

Impl´ementation

Un programme.- Ceci nous conduit au programme suivant :

/\* hanoi.c \*/

#include <stdio.h>

void deplace(int n, int p, int q, int r)

{

if (n == 1)

{

printf("Deplacer un disque du plot %d", p);

printf(" vers le plot %d\n", r);

}

else

{

deplace(n-1, p, r, q);

deplace(1, p, q, r);

deplace(n-1, q, p, r);

}

}

void main(void)

{

int n;

printf("Entrer le nombre de disques : ");

scanf("%d", &n);

deplace(n, 1, 2, 3);

}

Un exemple de session.- Pour n = 3 on obtient :

Entrer le nombre de disques : 3

D´eplacer un disque du plot 1 vers le plot 3

D´eplacer un disque du plot 1 vers le plot 2

D´eplacer un disque du plot 3 vers le plot 2

D´eplacer un disque du plot 1 vers le plot 3

D´eplacer un disque du plot 2 vers le plot 1

D´eplacer un disque du plot 2 vers le plot 3

D´eplacer un disque du plot 1 vers le plot 3

