UNIVERSITE CENTRALE
Ecole IT
Département Informatique
1ière année Licence
Atelier de programmation 1 : Langage C
Enseignants :
S. ZGHAL (Cours))
2020-2021

TD 1 : Récursivité

Exercice 1: Factorielle
Ecrire une fonction récursive qui permet de calculer la factorielle d'un entier positif.

Exercice 2: Somme
Ecrire une fonction récursive qui permet de calculer la somme des n (n>0) premiers termes (à partir de 1 jusqu'à n).

Exercice 3 : Fibonacci
Ecrire une fonction récursive qui permet de calculer les nombres de "Fibonacci" qui sont définis par :

- fib (0) = 0
- fib (1) = 1
- et pour tout n > 1, fib(n) = fib(n-1) + fib(n-2)

Exercice 4 : PGCD
Ecrire une fonction récursive qui permet de calculer le plus grand diviseur commun de deux entiers naturels strictement positifs.

Exercice 5 : Ackerman
Ecrire une fonction récursive qui permet de calculer la valeur de la fonction d'Ackerman A définie de la manière suivante pour m>0 et n>0 :

- Ack(m, n) = Ack(m-1, Ack(m, n-1))

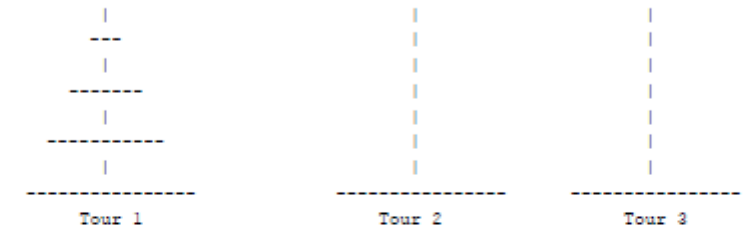
- Ack(0, n) = n+1 pour n > 0
- Ack(m, 0) = Ack(m-1, 1) pour m > 0

Exercice 6 : Tour de Hanoï

Supposons qu'on dispose de 3 disques à la tour 1 et on souhaite les déplacer à la deuxième tour en utilisant la tour 3 comme ntermédiaire. Les règles du jeu sont les suivantes :

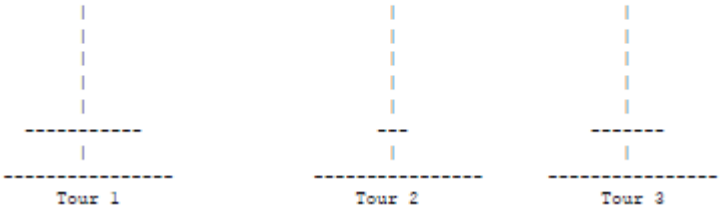
- Les disques ont un diamètre différent;
- On déplace un disque à la fois;
- on n'a pas le droit de placer un disque de diamètre supérieur sur un de diamètre inférieur.

étape 0 :

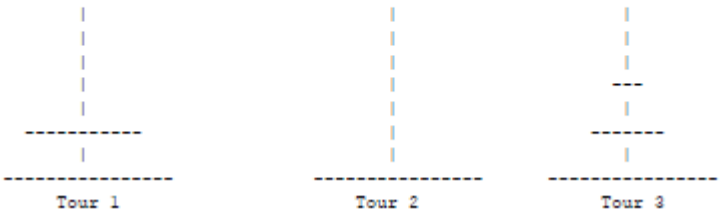


étape 1 : déplacer un disque de Tour 1 à Tour 2

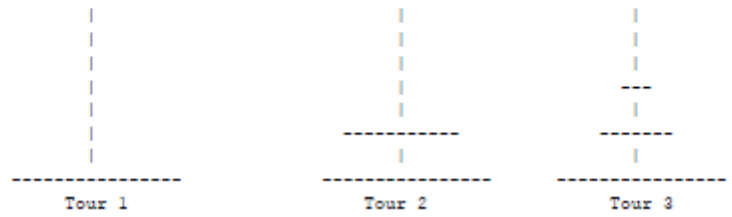
étape 2 : déplacer un disque de Tour 1 à Tour 3



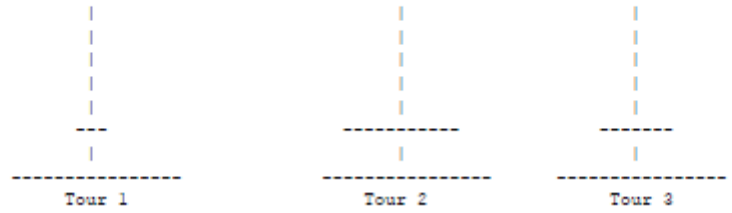
étape 3 : déplacer un disque de Tour 2 à Tour 3



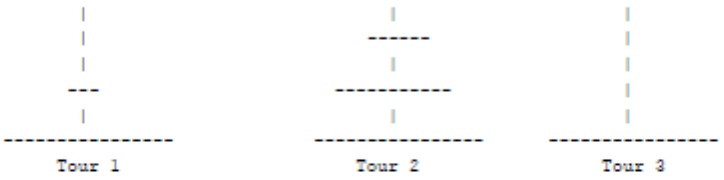
étape 4 : déplacer un disque de Tour 1 à Tour 2



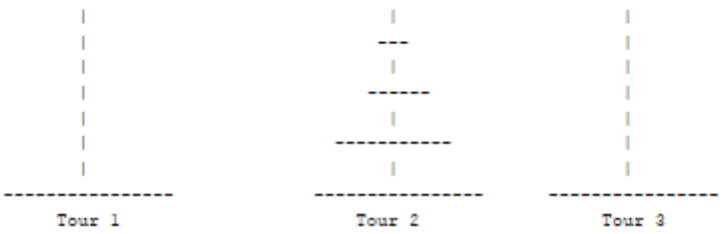
étape 5 : déplacer un disque de Tour 3 à Tour 1



étape 6 : déplacer un disque de Tour 3 à Tour 2



étape 7 : déplacer un disque de Tour 1 à Tour 2



Exercice 7: Conversion

1. Ecrire une fonction itérative en langage C qui permet de convertir un nombre décimal en bianire.

2. Ecrire une fonction récursive en langage C qui permet d'imprimer à l'écran la représentation binaire d'un nombre décimal.

Exercice 8: Pair et Impair

Un nombre N est pair si (N-1) est impair, et un nombre N est impair si (N-1) est pair. Ecrire deux fonctions récursives mutuelles pair(N) et impair(N) permettant de savoir si un nombre N est pair et si un nombre N est impair.

Exercice 9: Maximum tableau

Soit un tableau X de N entiers. Ecrire une fonction récursive simple permettant de déterminer le maximum du tableau

Exercice 10: Tableau trié

Un tableau X est trié par ordre croissant si x(i)<=x(i+1) pour i. Elaborer un algorithme récursif permettant de vérifier qu'un tableau X est trié ou non

Exercice 11: Palindrome

Un mot est un palindrome si on peut le lire dans les deux sans de gauche à droite et de droite à gauche. Exemple KAYAK est un palindrome. Ecrire une fonction récursive permettant de vérifier si un mot est palindrome.

TD 1 : Récursivité (Correction)

Exercice 1: Factorielle

```
int Fact(int n)
{
        int r;

        if(n==0)
        {r=1;}
        else
        {r=n*Fact(n-1);}
        return n;
}
```

Exercice 2: Somme

```
int somme(int n)
{
        if(n==0)
        {return 0;}
        else
    {return (n+somme(n-1));}
}
```

Exercice 3 : Fibonacci

```
int Fib(int n)
{
        int r;

        if(n==0)
        {r=0;}
        else
        {
                if(n==1)
                {r=1;}
                else
                {r=Fib(n-1)+Fib(n-2);}
        }
        return r;
}
```

Exercice 4 : PGCD

```
int pgcd(int a, int b)
{
  if (a % b == 0)
  return b;
  else
  return pgcd(b,a%b) ;
}
```

Exercice 5 : Ackerman

```
int Ack(int m, int n)
{
        int r;

        if(m==0)
        {r=n+1;}
        else
        {
                if(n==0)
                {r=Ack(m-1,1);}
                else
                {r=Ack(m-1,Ack(m,n-1));}
        }
        return r;
}
```

Exercice 6 : Tour de Hanoï

```
void hanoi (int n, int t1, int t2, int t3)
{
        if (n==1)
                printf("De la tour %d à la tour %d\n", t1, t2) ;
        else
        {
                hanoi(n-1,t1,t3,t2) ;
                hanoi(1,t1,t2,t3) ;
                hanoi(n-1,t3,t2,t1) ;
        }
}
```