Récursivité

Définition

Une fonction récursive est une fonction qui s'appelle elle même.

Exercice 1 : somme des premiers entiers

Rédiger un algorithme récursif de calcul de la somme des n premiers entiers, où n ≥ 1 est fourni par l'utilisateur.

In [1]:

```
def somme(n):
    if n == 1:
        return 1
    else:
        return somme(n - 1) + n

print(somme(5)) #Affiche 15
```

```
15
```

In [2]:

```
# Avec un seul return
def somme(n):
    r = 1
    if n != 1:
        r = somme(n - 1) + n
    return r

print(somme(5)) #Affiche 15
```

```
15
```

In [3]:

```
# Avec une ternaire
def somme(n):
    return 1 if n == 1 else somme(n - 1) + n

print(somme(5)) #Affiche 15
```

```
15
```

Exercice 2 : calcul d'une puissance

En remarquant que

```
x**3 = x**2 * x
x**4 = x**3 * X
```

(ainsi de suite), rédiger un algorithme récursif de calcul de , où et sont fournis par l'utilisateur.

In [4]:

```
def puissance(x,p):
    if p == 0:
        return 1
    else:
        return puissance(x, p-1) * x

print(puissance(5, 3))
```

```
125
```

In [5]:

```
# Avec un seul return
def puissance(x,p):
    r = 1
    if p != 0:
        r = puissance(x, p-1) * x
    return r
print(puissance(5, 3))
```

```
125
```

In [6]:

```
# Avec une ternaire
def puissance(x,p):
    return 1 if p == 0 else  puissance(x, p-1) * x

print(puissance(5, 3))
```

```
125
```

Exercice 3 : suite de Fibonacci

On rappelle que la suite de Fibonacci est la suite numérique (fn) définie par f0 = f1 = 1 et fn = fn-1 + fn-2 pour tout entier n ≥ 2.Ainsi, pour calculer un terme, il suffit de calculer les deux termes précédents. Rédiger un algorithme récursif de calcul du nième terme de la suite de Fibonacci, pour un entier n fourni par l'utilisateur.

In [7]:

```
def fibonacci(n):
    if n <= 1:
        return 1
    else:
        return fibonacci(n - 1) + fibonacci(n - 2)

print(fibonacci(5))
```

```
8
```

In [8]:

```
# Avec un seul return
def fibonacci(n):
    r = 1
    if n > 1:
        r = fibonacci(n - 1) + fibonacci(n - 2)
    return r

print(fibonacci(5))
```

```
8
```

In [9]:

```
# Avec une ternaire
def fibonacci(n):
    return 1 if n <= 1 else fibonacci(n - 1) + fibonacci(n - 2
)

print(fibonacci(5))
```

```
8
```

In [ ]:

```

```