TD3 : ASD.

Exercice 1 : Dénombrement avec les boucles

Pour chacune des fonctions, compter le nombre exact d'itérations de la boucle la plus interne réalisées en fonction de n. Lorsqu'il est trop difficile de calculer le nombre exact, proposer une fonction de n qui majore ce nombre. Donner la notation Θ de ce nombre d'itérations.

```python
def f1 (n):
  r = 0
  for i in range(1,n+1):
    for j in range(1,n+1):
      r = r + 3
  return r
```

n opérations) n^2 opérations

```python
def f2 (n):
  r = 0
  for i in range(1,n+1):
    for j in range(1,i+1):
      r = r + 3
  return r
```

$\frac{n(n+1)}{2}$

```python
def f3 (n):
  r = 0
  for i in range(5,n-5+1):
    for j in range(i-5,i+5+1):
      r = r + 3
  return r
```

$\sum_{j=i-5}^{i+5} 1$ $\quad \sum_{i=5}^{n-5} \left(\sum_{j=i-5}^{i+5} 1 \right) = \sum_{i=5}^{n-5} 10 = (n-9) * 11$

```python
def f4 (n):
  r = 0
  for i in range(1,n+1):
    for j in range (1,n+1):
      for k in range (1,n+1):
        r = r + 3
  return r
```

n^3

```python
def f5 (n):
  r = 0
  for i in range(1,n+1):
    for j in range (1,i+1):
      for k in range(1,j+1):
        r = r + 3
  return r
```

$\sum_{k=1}^{j} 1$ $\quad \sum_{j=1}^{i} j = \frac{i(i+1)}{2}$ $\quad \sum_{i=1}^{n} \frac{i(i+1)}{2} = \frac{1}{2}\left(\sum_{i=1}^{n} i^2 + \sum_{i=1}^{n} (i+1) \right) \dots$

```python
def f6 (n):
  r = 0
  i = n
  while i > 1:
    for j in range(1,i+1):
      r = r + 3
    i = i // 2
  return r
```

$\sum_{j=1}^{i} 1 = i$

$n + \frac{n}{2} + \frac{n}{4} + \dots + 2 = 2^k + \frac{2^k}{2} + \dots + \frac{2^k}{2^{k-1}}$

$= 2(2^{k-1} + 2^{k-2} + \dots + 1) = 2 \frac{1-2^k}{1-2} = 2(n-1)$

long :)

Exercice 2 :

Nous disposons de plusieurs algorithmes réalisant la même tache mais avec un nombre d'additions différent (colonne de gauche des tableaux).

Q 2.1 Indiquer le nombre d'opérations nécessaires à l'exécution de chaque algorithme pour les différentes tailles de données.

Complexité	$n = 10$	$n = 1\,000$	$n = 1\,000\,000$
n	10	1000	1000 000
n^2	100	10 000	10^{12}
n^3	1000	10^9	10^{18}
$\sqrt{n}$	3	100	1000
2^n	2^{10}	2^{1000}	$2^{1\,000\,000}$
$\log_2(n)$	3	9	19
$n\log_2(n)$	30	9000	20000000

Q 2.2 Indiquer le temps nécessaire à l'exécution de chaque algorithme pour les différentes tailles de données en supposant que l'ordinateur effectue 10^9 opérations à la seconde.

Complexité	$n = 10$	$n = 1\,000$	$n = 1\,000\,000$
n	1×10^{-8} s	10^{-6} s	10^{-3} s
n^2	10^{-7} s	10^{-5} s	1000s
n^3	10^{-6}	1 s	10^9
$\sqrt{n}$	$\sim 10^{-8}$ s	10^{-7}	10^{-6}
2^n	$\sim 10^{-6}$ s	10^{282} s	trop long <2 gg mob
$\log_2(n)$	$\sim 10^{-8}$ s	$\sim 10^{-8}$ s	$\sim 2\times10^{-8}$ s
$n\log_2(n)$	3×10^{-8} s	$\sim 10^{-5}$ s	2×10^{-2} s