

# TD - Analyse d'algorithmes en Python

## Ex0 : Classements et comparaisons

- $n!$ ,
- $(n-1)!$
- $3^n$
- $n \cdot 2^n$
- $2^n$
- $2^{(n+7)}$
- $n^3$
- $7n^3 + 5n^2 - 2n + 13$
- $n^2$
- $C(n, 2)$
- $n \log n$
- $n$
- $2^{(\log_2 n)}$
- $n \cdot \log(n^n)$
- $\sqrt[3]{n}$
- $\log(n^2)$
- $\log(\log(n))$
- $\log(\sqrt{\log n})$
- 24096.

---

## Ex1 -Boucle linéaire

```
for i in range(n):  
    print(i)
```

- 1) Donnez le nombre exact d'appels à print pour  $n = 10$  puis  $n = 20$ .
  - 2) Donnez une borne asymptotique  $\Theta(\cdot)$ .
- 

## Ex2 -Pas constant et demi-plage

```
for i in range(1, n//2, 5):  
    print(i)
```

- 1) Comptez exactement pour  $n = 10$  et  $n = 20$ .
  - 2) Donnez la borne asymptotique.
- 

### Ex3 -Deux boucles consécutives

```
for i in range(n):  
    x = i  
for j in range(n, 0, -1):  
    y = j
```

- 1) Nombre total d'itérations ?
  - 2) Borne asymptotique.
- 

### Ex4 -Boucle logarithmique

```
m = n  
while m > 1:  
    m //= 2
```

- 1) Comptez le nombre d'itérations pour  $n = 10$  et  $n = 20$ .
  - 2) Borne asymptotique.
- 

### Ex5 -Doubles boucles carré et triangulaire

(a) Carré

```
for i in range(n//2):  
    for j in range(n//2):  
        pass
```

(b) Triangulaire

```
for i in range(n):  
    for j in range(i, n):  
        pass
```

- 1) Comptez pour  $n = 10$  et  $n = 20$ .
  - 2) Donnez la borne asymptotique.
- 

### Ex6 -Logarithmes imbriqués

1ere Année Cycle Ingénieurs Génie Informatique

```
m = n
while m > 0:
    t = m
    while t > 1:
        t = int(t**0.5)
    m //= 2
```

Donnez une borne asymptotique serrée (indice:  $\log n \cdot \log(\log(n))$ ).

---

### Ex7 -Condition aléatoire

```
import random
m = 0
for i in range(n):
    m += i
if random.random() > 0.5:
    pass
else:
    for k in range(m * n):
        pass
```

- 1) Donnez **des bornes inférieure et supérieure** en fonction de  $n$ .
  - 2) Donnez la **complexité en espérance**.
- 

### Ex8 -Triple boucle triangulaire $\times 2$ (facile)

```
c = 0
for i in range(n):
    for j in range(i):
        for k in range(2):
            c += 1
```

- 1) Comptez exactement  $c$  pour  $n = 10$  puis  $n = 20$ .
  - 2) Donnez la borne asymptotique serrée.
- 

### Ex9 -Sauts multiplicatifs

```
i = 1
while i < n:
    i *= 3
```

1. Nombre d'itérations pour  $n = 10$  et  $n = 20$ .

1ere Année Cycle Ingénieurs Génie Informatique

2. Donnez une expression en fonction de  $\log_3 n$  et la borne asymptotique.
- 

### Ex10 -Boucle sur multiples

```
c = 0
for i in range(1, n+1):
    for j in range(i, n+1, i):
        c += 1
```

1. Calculez c exactement pour  $n = 10$  et  $n = 20$ .
  2. Montrez que  $c = \Theta(n \log n)$  (somme des  $\lfloor n/i \rfloor$ ).
- 

### Ex11 -Produit cartésien filtré (moyen)

```
c = 0
for i in range(n):
    for j in range(n):
        if (i + j) % 3 == 0:
            c += 1
```

1. Donnez c en fonction de n (indice :  $\sim 1/3$  du carré, justifier précisément).
  2. Vérifiez pour  $n = 9$  et  $n = 18$ .
- 

### Ex12 -Sélection (tri) : nombre exact de comparaisons

On exécute le tri par sélection sur n éléments distincts.

- 1) Montrez que le **nombre exact** de comparaisons vaut  $n(n-1)/2$ .
  - 2) Donnez la borne asymptotique et comparez à l'insertion triée naïve.
- 

### Ex13 -Détection de doublons

```
c = 0
for i in range(n):
    for j in range(i+1, n):
        c += 1
        if a[i] == a[j]:
            break
```

- 1) Bornez **pire cas** du nombre de comparaisons.

1ere Année Cycle Ingénieurs Génie Informatique

- 2) Avec un set auxiliaire, discutez le coût **moyen** (hachage uniforme) et le pire cas.
- 

### Ex14 -Tests d'appartenance : liste vs set

On effectue q requêtes «  $x \in S$  » sur un conteneur S de taille n.

- 1) Donnez le coût total en comparaisons pour S une **liste**.
  - 2) Donnez le coût attendu pour S un **set** Python (modèle de hachage uniforme).
  - 3) Pour  $n = 10^5$  et  $q = 10^5$ , estimez l'écart d'ordres de grandeur.
- 

### Ex15 -Euclide (PGCD)

```
def pgcd(a, b):  
    while b != 0:  
        a, b = b, a % b  
    return a
```

- 1) Donnez une **borne** sur le nombre d'itérations en fonction de  $\log a + \log b$ .
  - 2) Expliquez pourquoi le **pire cas** est atteint par des couples de nombres de Fibonacci consécutifs.
- 

### Ex16 -Boucle avec arrêt précoce

```
steps = 0  
for i in range(n):  
    steps += 1  
    if i*i >= n:  
        break
```

- 1) Donnez le nombre exact d'itérations en fonction de  $\lceil \sqrt{n} \rceil$ .
  - 2) Calculez pour  $n = 10$  et  $n = 20$ .
- 

### Ex17 -Somme des chiffres et taille d'entrée

```
s = 0  
m = n  
while m > 0:
```

```
s += m % 10  
m /= 10
```

- 1) Exprimez la complexité en fonction de la **valeur**  $n$  (base 10).
- 2) Re-exprimez en fonction de la **taille binaire** de l'entrée (nombre de bits), et discutez la différence.