

**Entraînement : calcul de complexité**

Pour tous les exercices, la grille d'évaluation est la suivante.

A (20)	Toutes les complexité ont été données de façon correcte. La notation $O$ a été bien utilisée.
B (16)	Les complexités sont toutes correctes mais quelques imprécisions sur la notation $O$ .
C (11)	Entre 1 et 3 erreurs sur les complexités, mais la classe de complexité (logarithmique, linéaire /polynomiale ou exponentielle) est respectée.
D (8)	Entre 1 et 3 erreurs dont des erreurs de classe de complexité.
E (1)	4 erreurs ou plus

**Exercice 1.**

Donner la complexité de chacun des algorithmes suivants (sans justification). Chaque algorithme prend en entrée un entier  $n$ . La réponse sera donnée en utilisant la notation  $O$ .

<p>Algo_a :</p> <pre> c ← 0 Pour i allant de 1 à n :     Pour j allant de 1 à i :         c ← c + 1 </pre> <p>Algo_b :</p> <pre> c ← 0 Pour i allant de 1 à n :     Algo_a(i) </pre> <p>Algo_c :</p> <pre> c ← 0 Tant que c &lt; n :     c ← c + 1000 </pre> <p>.</p>	<p>Algo_d :</p> <pre> c ← 0 Tant que c*c &lt; n :     c ← c+1 </pre> <p>Algo_e :</p> <pre> c ← 1 Tant que c &lt; n :     c ← c*2 </pre> <p>Algo_f :</p> <pre> f ← 1 c ← 0 Tant que c &lt; n :     i ← 0     Tant que i &lt; f :         i ← i+1     f ← f+i     c ← c+1 </pre>
---	--

Exemple de réponses et note finale associée.

Remarque : parfois certaines cases sont laissées vide si la réponse n'a pas été donnée (il vaut mieux avouer qu'on ne sait pas plutôt que d'écrire quelque chose de complètement faux).

**La première ligne donne les réponses correctes attendues**

a	b	c	d	e	f	note
$O(n^2)$	$O(n^3)$	$O(n)$	$O(\sqrt{n})$	$O(\log(n))$	$O(2^n)$	A
$O(\frac{n(n-1)}{2})$	$O(n^3)$	$O(\frac{n}{1000})$	$O(\sqrt{n})$	$O(\log(n))$	$O(2^n)$	B
$O(\frac{n(n-1)}{2})$	$O(n^2)$	$O(\frac{n}{1000})$	$\sqrt{n}$	$\log(\frac{n}{2})$		C
$O(n^2)$	$O(n^3)$	$O(n)$	$O(\frac{n}{2})$			C
$O(n^2)$	$O(n^3)$	$O(n)$	$O(\sqrt{n})$	$O(\frac{n}{2})$		D
$O(n^2)$	$O(n)$		$O(\log(n))$	$O(\log(n))$	$O(\log(n))$	E

**Exercice 2.**

Donner la complexité de chacun des algorithmes suivants (sans justification). Chaque algorithme prend en entrée un entier  $n$ .

<p>Algo_a :</p> <pre> c ← 0 Pour i allant de 1 à n :     Pour j allant de 1 à i :         Pour k allant de 1 à j :             c ← c + 1 </pre> <p>Algo_b :</p> <pre> c ← 0 Pour i allant de 1 à n :     c ← c + 1  Pour i allant de 1 à n :     c ← c + 1 </pre> <p>.</p>	<p>Algo_c :</p> <pre> c ← 1 d ← 0 Pour i allant de 1 à n :     c ← 2*c     Pour j allant de 1 à c :         d ← d+1 </pre> <p>Algo_d :</p> <pre> c ← n Tant que c &gt; 0 :     c ← c / 2 </pre> <p>Algo_e :</p> <pre> c ← 1 Tant que c &lt; n :     c ← c + 2 </pre>
--	--

Algo\_f :

```

c ← 0
Tant que c < n :
    Algo_b(c)
    c ← c + 1

```

Exemple de réponses et note finale associée.

Remarque : parfois certaines cases sont laissées vides si la réponse n'a pas été donnée (il vaut mieux avouer qu'on ne sait pas plutôt que d'écrire quelque chose de complètement faux).

**La première ligne donne les réponses correctes attendues**

a	b	c	d	e	f	note
$O(n^3)$	$O(n)$	$O(2^n)$	$O(\log(n))$	$O(n)$	$O(n^2)$	A
$O(n^3)$	$O(2n)$	$O(2^n)$	$O(\log(n))$	$O(\frac{n}{2})$	$O(n^2)$	B
$O(n^3)$	$O(n^2)$	$O(2^n)$	$O(\log(n))$	$O(2n)$	$O(n^2)$	C
$O(n^3)$	$O(n)$			$O(n)$	$O(n^2)$	C
$O(n^3)$	$O(n)$	$O(n^2)$	$O(\frac{n}{2})$	$O(n)$	$O(n^2)$	D
$O(n^3)$	$O(n^2)$			$O(n)$	$O(n)$	E

**Exercice 3** (Partiel 2017-18).

Donner la complexité de chacun des algorithmes suivants (sans justification). Chaque algorithme prend en entrée un entier  $n$ .

<p>Algo_a :</p> <pre> c ← 0 Pour i allant de 1 à n :     Pour j allant de 1 à i :         c ← c + 1      Pour j allant de i à n :         c ← c + 1  Algo_b : c ← 1 Tant que c &lt; n :     c ← c * 2 .</pre>	<p>Algo_c :</p> <pre> c ← 1 Tant que c &lt; n :     c ← c + 2  Algo_d : c ← n Tant que c &gt; 0 :     c ← c / 2  Algo_e : Pour i allant de 1 à n :     Si i * i &gt; n :         Retourner i</pre>
---	--

<p>Algo_f :</p> <pre> c ← 0 Pour i allant de 1 à n :     Pour j allant de 1 à n :         Pour k allant de 1 à n :             Pour l allant de 1 à n :                 c ← c + 1</pre>
---

Exemple de réponses et note finale associée.

**La première ligne donne les réponses correctes attendues**

a	b	c	d	e	f	note
$O(n^2)$	$O(\log(n))$	$O(n)$	$O(\log(n))$	$O(\sqrt{n})$	$O(n^4)$	A
$O(n^2)$	$O(\log(n))$	$O(\frac{n}{2})$	$O(\log(n))$	$O(\sqrt{n})$	$O(n^4)$	B
$O(n^2)$	$O(\log(n))$	$O(n)$	$O(\log(n))$	$O(n)$	$O(n^4)$	C
$O(n^3)$	$O(\log(n))$	$O(n)$	$O(\log(n))$	$O(\sqrt{n})$	$O(n^4)$	C
$O(n^3)$	$O(\log(n))$	$O(n)$	$O(n)$	$O(\sqrt{n})$	$O(n^4)$	D
$O(n^2)$	$O(\frac{n}{2})$	$O(n)$	$O(\log(n))$	$O(\sqrt{n})$	$O(n^4)$	D
$O(n^2)$	$O(\log(n))$	$O(n)$	$O(\log(n))$	$O(n)$	$O(4^n)$	D

Question donnée au partiel 1 2017-2018, résultats obtenus :

A	B	C	D	E
85%	0%	0%	15%	0%

**Exercice 4** (Partiel 2018-2019).

Donner la complexité de chacun des algorithmes suivants (sans justification). Chaque algorithme prend en entrée un entier positif  $n$ .

<p>Algo_a :</p> <pre> c ← 0 Pour i allant de 1 à n :     c ← c+2 </pre> <p>Algo_b :</p> <pre> c ← 1 Tant que c &lt; n :     c ← c*2 </pre> <p>Algo_c :</p> <pre> c ← 0 Pour i allant de 1 à n :     c ← c * c     Pour j allant de 1 à n :         c ← c * 2 </pre>	<p>Algo_d :</p> <pre> c ← 1 Pour i allant de 1 à n :     c ← c + 1     Pour j allant de 1 à i :         c ← c + 1 </pre> <p>Algo_e :</p> <pre> Tant que n &gt; 0 :     n ← n//2 </pre> <p>Algo_f :</p> <pre> c ← 0 Pour i allant de 1 à n :     Pour j allant de 1 à i :         Pour k allant de 1 à j :             c ← c + 1 </pre>
---	--

Exemple de réponses et note finale associée.

**La première ligne donne les réponses correctes attendues**

a	b	c	d	e	f	note
$O(n)$	$O(\log(n))$	$O(n^2)$	$O(n^2)$	$O(\log(n))$	$O(n^3)$	A
$O(\frac{n}{2})$	$O(\log(n))$	$O(n^2)$	$O(n^2)$	$O(\log(n))$	$O(n^3)$	B
$O(n)$	$O(\log(n))$	$O(n^2)$	$O(n^3)$	$O(\log(n))$	$O(n^3)$	C
$O(n)$	$O(\log(n))$	$O(n^2)$	$O(n^2)$	$O(\log(n))$	$O(n^2)$	C
$O(n)$	$O(n)$	$O(n^2)$	$O(n^2)$	$O(\log(n))$	$O(n^3)$	D

Question donnée au partiel 1 2018-2019, résultats obtenus :

A	B	C	D	E
60%	0%	18%	22%	0%

### Exercice 5 (2019-20).

Donner la complexité de chacun des algorithmes suivants (sans justification). Chaque algorithme prend en entrée un entier positif  $n$ .

```

Algo_a :
  c <- n
  Tant que c > 1 :
    c <- c/2 (division entière)

```

```

Algo_b :
  c <- 1
  Tant que c < n :
    c <- c*2

```

```

Algo_c :
  c <- 0
  Pour i allant de 1 à n :
    Pour j allant de 1 à n :
      c <- c+1

```

```

Algo_d :
  c <- 1
  Pour i allant de 1 à n :
    c <- c + 1
    Pour j allant de 1 à i :
      c <- c + 1

```

```

Algo_e :
  c <- 1
  Pour i allant de 1 à n :
    c <- c + 1
  Pour i allant de 1 à n :
    c <- c + 1
  Pour i allant de 1 à n :
    c <- c + 1

```

```

Algo_f :
  c <- 1
  Pour i allant de 1 à n :
    c <- c + 1
  Pour i allant de 1 à n :
    c <- c + 1
    Pour i allant de 1 à n :
      c <- c + 1

```

### Solution

- a —  $O(\log(n))$
- b —  $O(\log(n))$
- c —  $O(n^2)$
- d —  $O(n^2)$
- e —  $O(n)$
- f —  $O(n^2)$

Question donnée au partiel 1 2019-2020, résultats obtenus :

A	B	C	D	E
59%	0%	26%	7.5%	7.5%