MATHÉMATIQUES

Algorithmique et Python

Première spé maths

Liste • Extension • Compréhension • Indices • Parcours • Fonction
Suites • Seuil • Sécantes • Newton • Euler • Monte-Carlo • Espérance

Repères du programme

Ce cours suit le **programme de l'enseignement de spécialité de mathématiques de première générale** (arrêté du 26 février 2026, BO n°14 du 2 avril 2026, rentrée 2026), partie transversale « Algorithmique et programmation » et exemples d'algorithmes des différentes parties. Il couvre :

- consolidation : variables, affectation, instructions conditionnelles, boucles, fonctions ; programmation modulaire ;
- la **liste**, seule notion nouvelle : générer (extension, ajouts successifs, compréhension), manipuler les éléments et leurs indices, parcourir, itérer ;
- algorithmes du programme : termes, sommes et seuils de suites, factorielle, Syracuse, Fibonacci ; coefficients directeurs de sécantes ; méthode de Newton ; méthode d'Euler et valeur approchée de e ; méthode d'Archimède ; méthode de Monte-Carlo ; espérance, variance et écart type.

Conformément au programme, on se limite aux listes, sans autre type de collection.

Objectif : générer, manipuler et parcourir des listes en Python, structurer un programme en fonctions, et programmer les algorithmes du programme de première (suites, dérivation, exponentielle, trigonométrie, probabilités).

Document pédagogique – mathlvl.fr

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite. Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

Table des matières

1 Rappels et fonctions **3**

2 Les listes **3**

3 Suites **4**

4 Dérivation et exponentielle **4**

5 Trigonométrie : Archimède **5**

6 Probabilités **5**

7 Erreurs classiques **6**

8 Fiche de synthèse **6**

9 Exercices d'application **6**

10 Corrigés des exercices **7**

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite.
Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

1. Rappels et fonctions

Programmation modulaire

On découpe un problème en fonctions courtes : chacune a des entrées (paramètres) et une sortie (`return`). L'affectation s'écrit `x = 3` en Python et $x \leftarrow 3$ en langage naturel.

```
def f(x):
    return x**2 - 2

def signe(y):
    if y > 0:
        return "positif"
    elif y < 0:
        return "négatif"
    return "nul"

print(signe(f(1.5)))    # positif
```

2. Les listes

Générer une liste

```
L = [2, 3, 5, 7]                          # en extension
C = []
for k in range(1, 6):
    C.append(k**2)                        # ajouts successifs : [1, 4, 9, 16, 25]
P = [k for k in range(20) if k % 3 == 0] # en compréhension : [0, 3, 6, ..., 18]
```

La liste en compréhension traduit l'ensemble $\{k \in \{0, 1, \ldots, 19\} \mid k \text{ multiple de } 3\}$: la condition est un filtre logique.

Manipuler et parcourir

`L[0]` est le premier élément, `L[-1]` le dernier, `len(L)` la longueur ; `L.append(x)` ajoute, `L.remove(x)` supprime la première occurrence de `x`, `del L[i]` supprime l'élément d'indice `i`.

```
def moyenne(L):
    s = 0
    for x in L:              # itérer sur les éléments
        s = s + x
    return s / len(L)

def indice_max(L):
    i_max = 0
    for i in range(len(L)):  # parcourir les indices
        if L[i] > L[i_max]:
            i_max = i
    return i_max
```

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite. Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

3. Suites

Terme, somme, seuil

Pour $u_0 = 100$ et $u_{n+1} = 0{,}9u_n + 20$:

```
def terme(n):
    u = 100
    for k in range(n):
        u = 0.9 * u + 20
    return u

def seuil(s):            # premier n tel que u_n > s
    n, u = 0, 100
    while u <= s:
        u = 0.9 * u + 20
        n = n + 1
    return n
```

Factorielle, Syracuse, Fibonacci

```
def factorielle(n):
    p = 1
    for k in range(1, n + 1):
        p = p * k
    return p                # factorielle(10) = 3628800

def syracuse(n):
    L = [n]
    while n != 1:
        if n % 2 == 0:
            n = n // 2
        else:
            n = 3 * n + 1
        L.append(n)
    return L                # syracuse(6) = [6, 3, 10, 5, 16, 8, 4, 2, 1]

def fibonacci(n):
    F = [1, 1]
    while len(F) < n:
        F.append(F[-1] + F[-2])
    return F                # fibonacci(10) = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
```

4. Dérivation et exponentielle

Coefficients directeurs des sécantes

```
def secantes(f, a, pas):
    return [(f(a + h) - f(a)) / h for h in pas]

def carre(x):
    return x**2
```

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite. Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

```
print(secantes(carre, 1, [1, 0.1, 0.01]))   # [3.0, 2.1, 2.01]
```

Les coefficients se rapprochent de 2, le nombre dérivé de $x \mapsto x^2$ en 1.

Méthode de Newton

Pour approcher une solution de $f(x) = 0$, on remplace la courbe par sa tangente : $x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$. Avec $f(x) = x^2 - 2$ et $x_0 = 1$: 1,5 ; 1,416 7 ; 1,414 216… qui converge très vite vers $\sqrt{2}$.

```
def newton(f, fp, x, n):
    for k in range(n):
        x = x - f(x) / fp(x)
    return x
```

Exponentielle : méthode d'Euler et nombre e

$\exp' = \exp$ et $\exp(0) = 1$: avec un pas h, $y_{k+1} = y_k + h\,y_k = (1+h)y_k$. Avec $h = 0{,}1$, on obtient $\exp(1) \approx 1{,}1^{10} \approx 2{,}594$. La suite $\left(1 + \frac{1}{n}\right)^n$ donne 2,594 ; 2,705 ; 2,717 pour $n = 10$, 100, 1 000 : elle s'approche de $e \approx 2{,}718$.

5. Trigonométrie : Archimède

Approcher π

Archimède encadre π par les demi-périmètres de polygones réguliers inscrits dans un cercle de rayon 1 : en doublant le nombre de côtés, le côté c devient $\sqrt{2 - \sqrt{4 - c^2}}$.

```
from math import sqrt

def archimede(etapes):
    n, c = 6, 1.0            # hexagone : côté 1
    for k in range(etapes):
        c = sqrt(2 - sqrt(4 - c**2))
        n = 2 * n
    return n * c / 2         # demi-périmètre
```

On obtient 3 ; 3,106 ; 3,133 ; 3,139 ; 3,141 pour 6, 12, 24, 48, 96 côtés.

6. Probabilités

Monte-Carlo

```
from random import random

def estime_pi(N):
    c = 0
    for k in range(N):
        if random()**2 + random()**2 <= 1:
            c = c + 1
    return 4 * c / N
```

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite. Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

Une exécution avec $N = 100\,000$ a donné 3,138. Le même principe estime l'aire sous la parabole $y = x^2$ entre 0 et 1 (environ $\frac{1}{3}$).

Espérance, variance, écart type

```
from math import sqrt

def esperance(X, P):
    return sum([X[i] * P[i] for i in range(len(X))])

def variance(X, P):
    E = esperance(X, P)
    return sum([P[i] * (X[i] - E)**2 for i in range(len(X))])

X, P = [0, 1, 2, 3], [0.1, 0.3, 0.4, 0.2]
print(esperance(X, P), variance(X, P), sqrt(variance(X, P)))   # 1.7  0.81  0.9
```

7. Erreurs classiques

À éviter

- Oublier que les indices commencent à 0 : le dernier élément est `L[len(L) - 1]`, soit `L[-1]`.
- Utiliser `/` au lieu de `//` pour une division entière (Syracuse).
- Placer `return` à l'intérieur de la boucle.
- Confondre la condition d'arrêt d'une boucle `while` et la condition qu'on cherche : on boucle **tant que** le seuil n'est pas dépassé.

8. Fiche de synthèse

Générer	`[2, 3, 5]` ; `append` dans une boucle ; `[f(k) for k in … if …]`
Manipuler	`L[i]`, `L[-1]`, `len`, `append`, `remove`, `del`
Parcourir	`for x in L` ; `for i in range(len(L))`
Suites	boucle `for` (terme, somme) ; `while` (seuil)
Newton	$x \leftarrow x - \frac{f(x)}{f'(x)}$
Euler	$y \leftarrow y + h\,y'$

9. Exercices d'application

Exercice 1 – Liste en compréhension

Écrire la liste des carrés des entiers pairs compris entre 0 et 10.

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite. Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).

Exercice 2 – Somme

Écrire une fonction `somme(L)` qui renvoie la somme des éléments d'une liste, sans utiliser `sum`.

Exercice 3 – Seuil

Pour la suite $u_0 = 1\,000$, $u_{n+1} = 1{,}05u_n$, écrire une fonction qui renvoie le premier n tel que $u_n > 2\,000$, et donner le résultat.

10. Corrigés des exercices

Corrigé de l'exercice 1

Corrigé. `[k**2 for k in range(0, 11, 2)]`, soit `[0, 4, 16, 36, 64, 100]`.

Corrigé de l'exercice 2

Corrigé.

```
def somme(L):
    s = 0
    for x in L:
        s = s + x
    return s
```

Corrigé de l'exercice 3

Corrigé.

```
def seuil():
    n, u = 0, 1000
    while u <= 2000:
        u = 1.05 * u
        n = n + 1
    return n
```

Elle renvoie 15 ($1{,}05^{14} \approx 1{,}98$ et $1{,}05^{15} \approx 2{,}08$).

mathlvl.fr — Tous droits réservés. Toute reproduction, diffusion, revente ou exploitation commerciale, même partielle, est strictement interdite. Tout contrevenant s'expose à des poursuites en vertu du Code de la propriété intellectuelle (art. L.335-2 et suivants).