Amit Saha

Nouvelle matière du programme du collège et du lycée

PYTHON pour les MATHS

Dès 14 ans

no starch press

EYROLLES

PYTHON pour les MATHS

Explore les mathématiques en codant !

Cet ouvrage va te montrer comment utiliser Python pour maîtriser des sujets du niveau fin de collège/lycée comme les statistiques, la géométrie, les probabilités et le calcul infinitésimal. Tu débuteras par des projets simples, comme créer un programme calculant la factorielle ou résolvant une équation quadratique, puis, une fois que tu auras appris les bases, tu pourras t'attaquer à des projets plus compliqués.

Au fil de ta lecture, tu vas découvrir de nouvelles manières d'explorer les mathématiques tout en acquérant des compétences en programmation qui te serviront tout au long de tes études. Tu verras entre autres comment :

- décrire des données avec les statistiques et les organiser sous forme de courbes, diagrammes en bâtons ou de dispersion ;
- explorer la théorie des ensembles et les probabilités avec des programmes simulant des lancers de pièce et de dé, et autres jeux de chance ;
- résoudre des problèmes d'algèbre avec les symboles en Python ;
- tracer des formes géométriques et explorer les fractales comme la fougère de Barnsley, le triangle de Sierpiński et l'ensemble de Mandelbrot ;
- écrire des programmes calculant des dérivées et des intégrales de fonctions.

Coder un programme résolvant des inégalités, tracer le graphique d'un projectile en mouvement, mélanger un paquet de cartes, estimer l'aire d'un cercle en jetant 100 000 fléchettes virtuelles sur une surface, explorer la relation entre la suite de Fibonacci et le nombre d'or, voilà ce qui t'attend dans ce livre et bien plus encore.

Que tu sois intéressé par les mathématiques mais encore débutant en programmation ou que vous soyez un professeur cherchant à introduire la programmation dans vos cours, vous verrez que Python rend la programmation simple et pratique.

À propos de l'auteur

Ingénieur en informatique, Amit Saha a travaillé pour Red Hat et Sun Microsystems. Il a également créé et s'est occupé de la maintenance de Fedora Scientific, un logiciel distribué par Linux pour un usage scientifique et éducationnel.

À qui s'adresse cet ouvrage ?

Aux collégiens, lycéens, parents, enseignants et associations.

Sur www.editions-eyrolles.com/go/pythonmaths

Télécharge le code source des exemples et les solutions des défis du livre.

Dans la même collection

www.editions-eyrolles.com

no starch press

PYTHON pour les MATHS

Amit Saha

PYTHON pour les MATHS

EYROLLES

ÉDITIONS EYROLLES
61, bd Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

Traduction autorisée de l'ouvrage en langue anglaise intitulé
Doing Math with Python
de Amit Saha (ISBN : 978-1-59327-640-9),
publié par No Starch Press.

À Protyusha, pour m'avoir toujours soutenu

REMERCIEMENTS

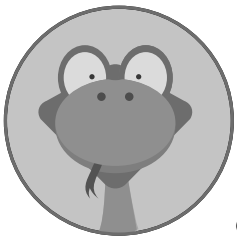
Je voudrais remercier toute l'équipe de No Starch Press pour avoir rendu ce livre possible. Depuis les premiers courriels échangés avec Bill Pollock et Tyler Ortman à propos de ce livre et pour tout le reste du processus, j'ai eu plaisir à travailler avec chaque personne de l'équipe. Seph Kramer fut exceptionnel par son regard technique et les suggestions qu'il a su apporter et, grâce à la méticulosité de Riley Hoffman, nous avons pu vérifier et revérifier que tout soit correct. Il est juste de dire que, sans ces deux personnes, le livre n'aurait jamais été ce qu'il est aujourd'hui. Merci à Jeremy Kun et Otis Chodosh qui se sont assurés de l'exactitude des parties mathématiques. Je voudrais aussi remercier le réviseur technique Julianne Jigour pour sa rigueur.

Le module `SymPy` est au centre de plusieurs des chapitres de ce livre et je voudrais donc remercier toutes les personnes de sa liste de courriel qui ont su répondre avec patience à mes questions et réviser mes corrections avec promptitude. Je voudrais encore remercier la communauté de `matplotlib` pour avoir clarifié et répondu à mes doutes.

Je voudrais aussi remercier David Ash pour m'avoir prêté son MacBook, lequel m'a aidé à écrire les instructions d'installation des programmes.

Je dois finalement remercier chaque écrivain et chaque penseur qui a pu m'inspirer pour écrire ce livre.

AVANT-PROPOS



Le but de ce livre est d'apporter sous un même sujet trois domaines qui me sont chers : la programmation, les mathématiques et les sciences.

Qu'est-ce que cela signifie précisément ? Au travers de ces pages, nous allons explorer de manière informatique des sujets du niveau fin de collège/lycée, comme manipuler des unités de mesure, examiner la trajectoire d'un projectile, calculer la moyenne, la médiane et le mode, déterminer une corrélation linéaire, résoudre des équations algébriques, décrire le mouvement d'un pendule, simuler un jeu de dés, créer des formes géométriques ou trouver des limites, des dérivées et des intégrales de fonctions. Pour beaucoup de gens, ces sujets sont déjà familiers mais, plutôt qu'une feuille et un stylo, nous utiliserons ici l'outil informatique pour les explorer.

Nous allons écrire des programmes qui prendront des nombres et des formules en entrée, feront les calculs souhaités et afficheront la solution ou dessineront un graphique. Certains de ces programmes sont des calculateurs puissants capables de résoudre des problèmes mathématiques. Entre autres, ils vous aideront à trouver les solutions

d'équations, à calculer la corrélation entre deux jeux de données et à déterminer le maximum d'une fonction. Dans d'autres programmes, nous allons simuler des événements de la vie courante comme le mouvement d'un projectile, un lancer de pièce ou de dé. Utiliser des programmes pour simuler de tels événements est une manière simple de les analyser et d'en apprendre plus à leur propos.

Vous trouverez aussi des sujets qui sont très difficiles à explorer sans l'aide de l'informatique. Par exemple, dessiner des fractales à la main est très fastidieux, voire impossible, alors qu'il nous suffit d'utiliser une boucle « for » contenant l'opération adéquate pour les créer avec un programme. Vous trouverez certainement que ce nouveau contexte rend l'apprentissage des mathématiques et de l'informatique plus ludique, amusant et gratifiant.

Qui devrait lire ce livre ?

Si vous êtes en train d'apprendre la programmation, vous apprécierez la manière qu'a ce livre d'exposer les différentes voies de résolution de problèmes à l'aide de l'outil informatique. De même, si vous enseignez à un tel public, j'espère que vous trouverez dans ce livre des idées utiles pour illustrer l'utilité de la programmation au sein du monde abstrait des sciences.

Ce livre suppose que le lecteur connaisse les bases essentielles de la programmation en Python 3 (notamment, ce qu'est une fonction, les arguments des fonctions, le concept de classe de Python et des objets de classe ou les boucles). L'annexe B couvre une partie des autres fonctionnalités de Python utilisées dans nos programmes, mais le livre ne nécessite pas que vous les connaissiez d'avance. Si vous souhaitez avoir des bases plus solides je vous recommande de lire *Python pour les kids* de Jason Briggs (Eyrolles, 2015).

Que contient ce livre ?

Ce livre consiste en sept chapitres et deux annexes. Chaque chapitre finit par une collection de défis pour le lecteur. Je vous recommande d'essayer de les résoudre. Vous apprendrez davantage en essayant d'écrire vos propres programmes. Certains de ces défis vous amèneront à explorer de nouveaux domaines et vous permettront d'améliorer votre apprentissage.

- Le chapitre 1, **Les nombres**, commence avec les opérations mathématiques classiques et avance progressivement vers des sujets d'un niveau mathématique supérieur.
- Le chapitre 2, **Visualiser des données à l'aide d'un graphique**, se concentre sur la création des graphiques avec `matplotlib`.
- Le chapitre 3, **Décrire des données à l'aide des statistiques**, approfondit l'analyse des données via l'apprentissage des outils statistiques de base comme la moyenne, la médiane, le mode ou la corrélation

linéaire entre les variables d'un jeu de données. Vous apprendrez aussi à lire des données depuis un format populaire de fichiers pour le stockage des données : le CVS.

- Le chapitre 4, **Algèbre et mathématiques symboliques avec SymPy**, vous initie à l'utilisation de la bibliothèque `SymPy` pour traiter des problèmes d'algèbre. Il commence par les bases de la représentation des expressions algébriques en informatique avant de vous amener sur des sujets plus complexes comme la résolution d'équations.
- Le chapitre 5, **Jouer avec les ensembles et les probabilités**, traite de la représentation des ensembles en mathématiques et vous initie aux probabilités discrètes. Vous y apprendrez aussi à simuler des événements probabilistes à loi uniforme et non uniforme.
- Le chapitre 6, **Tracer des figures géométriques et des fractales**, vous apprend à tracer divers types de figures et à créer des figures animées à l'aide de `matplotlib`.
- Le chapitre 7, **Résoudre des problèmes de calcul infinitésimal**, vous expose quelques fonctions mathématiques disponibles dans la bibliothèque standard de Python et dans `SymPy`, puis vous présente la résolution de problèmes du domaine du calcul infinitésimal.
- L'annexe A, **Installation des logiciels**, couvre l'installation de Python 3, de `matplotlib` et de `SymPy` sous Windows, Linux et Mac OS X.
- L'annexe B, **Quelques caractéristiques de Python**, traite plusieurs fonctionnalités de Python qui pourraient être utiles aux débutants.

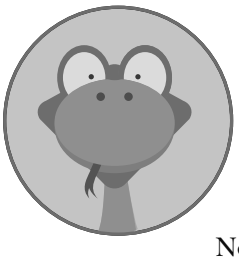
Scripts, solutions et indices

Le site web de ce livre est *http://www.editions-eyrolles.com/dl/14364*.
Vous pourrez y télécharger tous les programmes de ce livre et vous y trouverez des indices et les solutions à vos défis. Il fournit aussi des liens additionnels concernant les mathématiques, les sciences ou les ressources de Python que je trouvais utiles ainsi que des corrections ou ajouts au livre.
Les logiciels changent régulièrement. De nouvelles versions de Python, `SymPy` ou `matplotlib` pourraient causer des dysfonctionnements dans les programmes de ce livre. Vous trouverez ces changements et les moyens d'adapter vos programmes sur ce site web.
J'espère que ce livre rendra votre voyage de découverte de la programmation plus amusant et d'une pertinence immédiate. Place aux mathématiques !

TABLE DES MATIÈRES

1

LES NOMBRES



Faisons nos premiers pas dans la découverte de l'utilisation de Python pour explorer le monde des mathématiques et des sciences. Nous commencerons en douceur pour que vous puissiez vous habituer à Python lui-même. Nous commencerons par exécuter des opérations mathématiques élémentaires puis nous rédigerons des programmes simples pour manipuler et comprendre les nombres. Au travail !

Opérations mathématiques

L'interpréteur interactif de Python nous sera utile tout au long de ce livre. Démarrez l'interpréteur Python 3 IDLE et dites « bonjour » (voir figure 1-1) en écrivant `print('bonjour IDLE')`, puis pressez la touche **Entrée** (pour plus d'informations concernant l'installation de Python et le démarrage de l'IDLE, voir l'annexe A.) IDLE exécute votre commande et affiche les mots entre guillemets sur l'écran. Félicitations, vous venez d'écrire un programme !

Lorsque vous voyez >>> s'afficher de nouveau, IDLE est prêt à recevoir de nouvelles instructions.

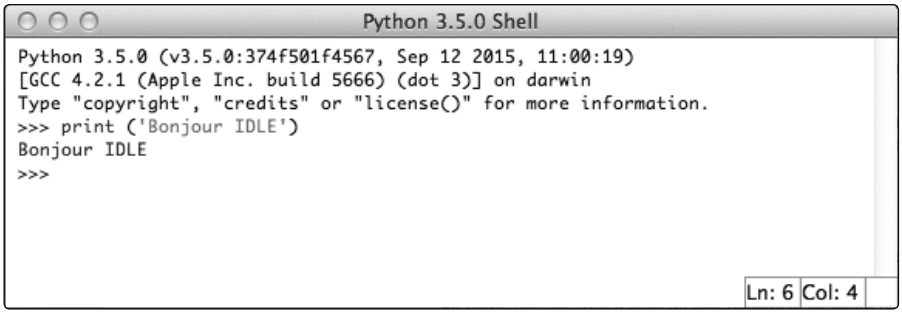


Figure 1-1 : Interpréteur Python 3 IDLE

Python peut être utilisé comme un simple calculateur, exécutant des calculs basiques. Il suffit de taper une expression mathématique et Python se charge de l'évaluer. Lorsque la touche **Entrée** est pressée, le résultat apparaît immédiatement.
Essayez par vous-même. Vous pouvez additionner et soustraire des nombres en utilisant respectivement les opérateurs + et - :

```
>>> 1+2
3
>>> 1+3.5
4.5
>>> -1+2.5
1.5
>>> 100-45
55
>>> -1.1+5
3.9
```

Pour multiplier, utilisez l'opérateur * :

```
>>> 3*2
6
>>> 3.5*1.5
5.25
```

Pour diviser, utilisez l'opérateur / :

```
>>> 3/2
1.5
>>> 4/2
2.0
```

Comme vous le voyez ici, lorsque vous demandez à Python d'effectuer une division, il retourne le nombre suivi de sa partie fractionnelle. Si vous souhaitez que le résultat soit un entier, en supprimant sa partie décimale, vous devrez utiliser l'opérateur de division arrondie au nombre inférieur // :

```
>>> 3//2
1
```

L'opérateur // divise le premier nombre par le second puis arrondit le résultat à l'entier inférieur. Cela devient intéressant lorsque l'un des chiffres est négatif. Par exemple :

```
>>> -3//2
-2
```

Le résultat final est l'entier inférieur au résultat de la division (-3/2=-1.5, donc le résultat est -2).
Par ailleurs, si vous ne voulez que le reste de la division, vous devrez utiliser l'opérateur modulo `%` :

```
>>> 9%2
1
```

Vous calculez la puissance d'un nombre en utilisant l'opérateur ** :

```
>>> 2**2
4
>>> 2**10
1024
>>> 1**10
1
```

Les exposants servent également pour calculer des puissances inférieures à 1. Par exemple, la racine carrée d'un nombre n peut être exprimée comme $n^{1/2}$ et la racine cubique comme $n^{1/3}$:

```
>>> 8**(1/3)
2.0
```

Comme le montre cet exemple, vous devez utiliser des parenthèses pour combiner des expressions afin d'en créer de plus compliquées. Python évalue l'expression selon la règle standard pour l'ordre de calcul PEMDAS – parenthèses, exposants, multiplications, divisions, additions et soustractions. Considérons les deux expressions suivantes, l'une sans les parenthèses et l'autre avec :

```
>>> 5+5*5
30
>>> (5+5)*5
50
```

Dans le premier exemple, Python calcule d'abord la multiplication : 5 fois 5 font 25, 25 et 5 font 30. Dans le second exemple, l'expression entre parenthèses est calculée en premier, comme nous l'aurions pensé : 5 et 5 font 10, 10 fois 5 font 50.
Nous venons de voir les bases nécessaires à la manipulation des nombres en Python. Voyons maintenant comment affecter des noms aux nombres.

Les labels : attacher des noms aux nombres

Comme nous commençons à concevoir des programmes plus complexes, nous allons devoir affecter des noms aux nombres – parfois par commodité, mais le plus souvent par nécessité. Voici un exemple simple :

```
❶ >>> a=3
  >>> a+1
  4
❷ >>> a=5
  >>> a+1
  6
```

En ❶, nous donnons le nom `a` au chiffre `3`. Quand nous demandons à Python d'évaluer le résultat de l'expression `a+1`, il comprend que `a` renvoie à `3`, lui ajoute alors `1` puis affiche le résultat (`4`). En ❷, nous changeons la valeur affectée à `a` en un `5` et cela se reflète sur le résultat de la seconde addition. Utiliser le nom `a` est pratique car il suffit de changer le nombre vers lequel `a` pointe et Python se servira de cette nouvelle valeur lorsque `a` sera appelé de nouveau dans la suite de votre code.

Ce type de nom est appelé « label » (qui pourrait être traduit par « étiquette » en français). Vous avez déjà entendu parler du terme « variable » qui décrit la même chose dans d'autres circonstances. Toutefois, étant donné que « variable » est aussi un terme mathématique (utilisé pour se référer par exemple au x dans l'équation $x+2=3$), nous l'utiliserons seulement dans le cas d'équations ou d'expressions mathématiques.

Différents types de nombres

Vous avez sûrement remarqué que nous avons énoncé deux types de nombres lors de la présentation des opérations mathématiques : les nombres sans partie décimale, dont vous connaissez déjà le nom d'*entiers*, et les nombres avec une partie décimale, que les programmeurs nomment « nombres à virgule flottante » (ou *réels*). Pour nous humains, cela ne fait aucune différence que les nombres soient sous forme d'entiers, de réels, de fractions ou encore écrits en chiffres romains. Cependant, certains programmes que nous allons écrire par la suite n'auront de sens qu'en utilisant un certain type de nombres ; nous demanderons donc souvent au programme de vérifier si les nombres que nous entrons sont du bon type en écrivant quelques lignes dans le code.

Python considère les entiers et les réels comme étant de types différents. Si vous utilisez la fonction `type()`, Python vous indique quel type de nombre vous venez d'entrer :

```
>>> type(3)
<class 'int'>

>>> type(3.5)
<class 'float'>
```

```
>>> type(3.0)
<class 'float'>
```

Vous constatez ici que Python considère le nombre `3` comme un entier (type `'int'`) et `3.0` comme un réel (type `'float'`). Nous savons tous que `3` et `3.0` sont mathématiquement équivalents, mais dans beaucoup de cas Python va traiter ces deux nombres de manière différente parce qu'ils sont de types différents.

Certains des programmes que nous écrivons dans ce chapitre ne fonctionnent proprement que si l'entrée est un entier. Comme nous venons de le voir, Python ne reconnaît pas les nombres 1.0 et 4.0 comme des entiers, donc si nous voulons quand même les utiliser comme entrées dans ces programmes, nous devons les convertir en entiers. Heureusement, il existe une fonction Python pour cela :

```
>>> int(3.8)
3
>>> int(3.0)
3
```

La fonction `int()` prend en entrée un réel, se débarrasse de tout ce qui vient après la virgule et retourne l'entier résultant. La fonction `float()` fonctionne de la même manière pour faire la conversion inverse :

```
>>> float(3)
3.0
```

`float()` prend en entrée un entier et lui ajoute une partie décimale nulle afin de le changer en réel.

Travailler avec les fractions

Python sait aussi gérer les fractions, mais pour utiliser cette fonctionnalité nous devrons utiliser le module `fractions`. On peut voir un module comme un programme écrit par quelqu'un d'autre, que vous pouvez réutiliser dans votre propre programme. Dans un module peuvent être définis des classes, des fonctions et même des labels. Ils peuvent faire partie de la bibliothèque standard de Python ou être mis à disposition par un tiers. Dans ce dernier cas, vous devrez installer le module avant de pouvoir vous en servir.

Le module `fractions` fait partie de la bibliothèque standard, ce qui signifie qu'il est déjà installé. Il définit la classe `Fraction`, dont on va se servir dans nos programmes. Pour cela, nous devons l'*importer*, c'est-à-dire signifier à Python que nous allons l'utiliser. Voyons un exemple rapide (création d'un nouveau label, `f`, qui renvoie à la fraction ¾) :

```
❶ >>> from fractions import Fraction
❷ >>> f=Fraction(3,4)
```

```
❸ >>> f
Fraction(3,4)
```

Nous importons tout d'abord la classe Fraction depuis le module ❶. Ensuite, nous créons un objet de cette classe en fournissant le numérateur et le dénominateur comme paramètres ❷. Ceci crée un objet Fraction pour la valeur ¾. Lorsque nous demandons à Python d'afficher l'objet ❸, il le fait sous la forme `Fraction(`*`numérateur, dénominateur`*`)`.
Vous pouvez soumettre aux fractions les opérations mathématiques classiques, notamment les comparaisons. Vous pouvez aussi utiliser une fraction, un entier et un réel dans une même expression :

```
>>> Fraction(3,4)+1+1.5
3.25
```

Lorsqu'il y a un réel dans une expression, le résultat de l'expression est toujours un réel.
En outre, lorsque vous n'avez qu'une fraction et un entier dans une expression, le résultat est une fraction, même si celle-ci a 1 pour dénominateur.

```
>>> Fraction(3,4)+1+Fraction(1,4)
Fraction(2,1)
```

Vous connaissez maintenant les bases pour manipuler les fractions en Python. Enchaînons avec un autre type de nombres.

Les nombres complexes

Les nombres que nous avons vus jusqu'ici font tous partie de la famille des réels. Python fonctionne aussi avec les nombres complexes, dont la partie imaginaire est identifiée par la lettre `j` ou `J` (contrairement au i utilisé dans les notations mathématiques). Par exemple, le complexe 2+3i est écrit `2+3j` en Python :

```
>>> a=2+3j
>>> type(a)
<class 'complex'>
```

Comme vous pouvez le voir, lorsque nous appliquons la fonction `type()`, Python nous informe que l'objet est de type `complex`.
Vous pouvez aussi définir un tel nombre en utilisant la fonction `complex()` :

```
>>> a=complex(2,3)
>>> a
(2+3j)
```

Dans ce dernier cas, nous plaçons les parties réelle et imaginaire en arguments de la fonction `complex()` et celle-ci retourne le nombre correspondant.

Vous pouvez additionner et soustraire des nombres complexes de la même manière qu'avec les réels :

```
>>> b=3+3j
>>> a+b
(5+6j)
>>> a-b
(-1+0j)
```

Les multiplications et divisions entre nombres complexes s'appliquent de façon similaire.

```
>>> a*b
(-3+15j)
```

```
>>> a/b
(0.8333333333333334+0.16666666666666666j)
```

Le modulo `%` et la division arrondie à l'entier inférieur `//` ne fonctionnent pas sur les complexes.
Les parties réelle et imaginaire d'un nombre complexe peuvent être récupérées en utilisant ses attributs `real` et `imag` :

```
>>> z=2+3j
>>> z.real
2.0
>>> z.imag
3.0
```

Le conjugué d'un complexe a la même partie réelle mais une partie imaginaire de signe opposé. Il est retourné par la méthode `conjugate()` :

```
>>> z.conjugate()
(2-3j)
```

Les parties imaginaire et réelle sont toutes deux des nombres à virgule flottante. Vous pouvez calculer le module du nombre complexe avec la formule suivante, où x et y sont respectivement les parties réelle et imaginaire du nombre : $\sqrt{x^2 + y^2}$. En Python, cela s'écrit de la manière suivante :

```
>>> (z.real**2+z.imag**2)**0.5
3.605551275463989
```

Une manière plus simple de trouver le module d'un nombre complexe est d'utiliser la fonction `abs()`. Celle-ci retourne la valeur absolue lorsqu'elle est utilisée sur un réel. Par exemple, `abs(5)` et `abs(-5)` retournent toutes deux `5`. Cependant, appliquée à un nombre complexe, la fonction renvoie son module :

```
>>> abs(z)
3.605551275463989
```

APPRENDS À CODER DÈS 5 ANS !

NOUVELLE MATIÈRE DU PROGRAMME, DU CE1 À LA 3E !

SCRATCH

Dès 5 ans
160 pages – 15,90 €

Dès 8 ans
160 pages – 15,90 €

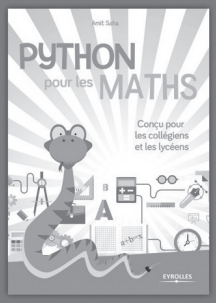
Dès 8 ans
64 pages – 12 €

Dès 10 ans
288 pages – 25 €
À paraître en septembre

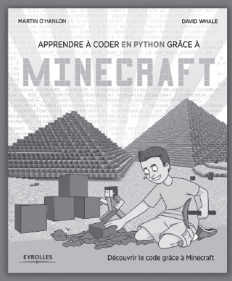
PYTHON

Dès 10 ans
64 pages – 12 €

Dès 10 ans
332 pages – 22,90 €



Dès 10 ans
300 pages – 22,90 €
À paraître en juin



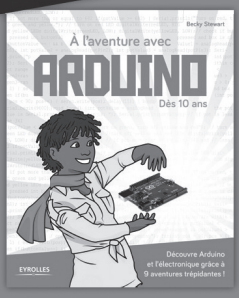
Dès 10 ans
350 pages – 24,90 €
À paraître en juin

WEB

Dès 10 ans
364 pages – 22,90 €

Dès 10 ans
152 pages – 19,90 €
À paraître en septembre

MAKERS



Dès 10 ans
352 pages – 25 €

Dès 8 ans
64 pages – 12 €
À paraître en septembre

EYROLLES

Pour en savoir plus : www.editions-eyrolles.com/go/coderaveceyrolles