

100%  
**EXOS**

1<sup>re</sup>

NOUVEAU  
BAC

# Maths

SPÉCIALITÉ

*l'entraînement  
intensif!*

**220** EXERCICES  
progressifs & minutés

**30** SUJETS de contrôle

**CORRIGÉS** détaillés  
& commentés

**COURS & MÉTHODES**



GRATUIT\*: des ressources interactives  
et des parcours de révision  
sur **annabac.com**





**100%**  
**EXOS**

**1<sup>re</sup>**

# Maths

**SPÉCIALITÉ**

**Sophie Barache**

Professeur agrégée de mathématiques  
au lycée Jean-Lurçat de Martigues

**Fabrice Barache**

Professeur agrégé de mathématiques  
à l'école de l'air de Salon-de-Provence

**Sophie Bauer**

Professeur agrégée de mathématiques  
au lycée Saint-Exupéry de La Rochelle

**Raphaël Bauer**

Professeur agrégé de mathématiques  
au lycée Saint-Exupéry de La Rochelle

Maquette de principe : Frédéric Jély  
Mise en pages : Grafatom  
Schémas : Grafatom  
Édition : Julien Lionnet

---

Sous réserve des exceptions légales, toute représentation ou reproduction intégrale ou partielle, faite, par quelque procédé que ce soit, sans le consentement de l'auteur ou de ses ayants droit, est illicite et constitue une contrefaçon sanctionnée par le Code de la Propriété Intellectuelle. Le CFC est le seul habilité à délivrer des autorisations de reproduction par reprographie, sous réserve en cas d'utilisation aux fins de vente, de location, de publicité ou de promotion de l'accord de l'auteur ou des ayants droit.

## Votre ouvrage 100 % exos

- ▶ Conforme au nouveau programme de Maths 1<sup>re</sup> (spécialité) entré en vigueur à la rentrée 2019, ce « 100 % exos » vous propose une **méthode de travail** complète et un **entraînement intensif** sur mesure, faisant une large place à la préparation du **nouveau bac**.
- ▶ Pour chaque thème du programme, vous trouverez un **COURS** structuré, les **MÉTHODES** qu'il faut maîtriser, des **EXERCICES** progressifs et leurs **CORRIGÉS** détaillés.
- ▶ Assortis d'**indications de solutions**, de **commentaires** et de **conseils** des auteurs, tous les exercices corrigés vous permettent :
  - de **comprendre** les notions essentielles et de maîtriser le cours ;
  - de **progresser** et de vous **entraîner** à votre rythme ;
  - de vous **évaluer** et de **réussir** vos devoirs et contrôles ;
  - de vous **préparer** à l'entrée en Terminale.

## Le site de vos révisions

- ▶ L'achat de cet ouvrage vous permet de bénéficier d'un **ACCÈS GRATUIT\*** à toutes les **ressources** d'annabac.com : fiches, quiz, sujets corrigés... et à ses **parcours de révision** personnalisés.
- ▶ Pour profiter de cette offre, rendez-vous sur **www.annabac.com**, dans la rubrique « Je profite de mon avantage client ».



\* Selon les conditions précisées sur le site.

ALGORITHMIQUE  
ET PROGRAMMATION

1 Prise en main de Python

|                            |                               |    |
|----------------------------|-------------------------------|----|
| <b>COURS</b><br>& METHODES | .....                         | 9  |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 17 |
|                            | S'entraîner .....             | 18 |
|                            | Préparer un contrôle.....     | 19 |
|                            | Aller plus loin .....         | 20 |
| <b>CORRIGÉS</b>            | .....                         | 22 |

ALGÈBRE

2 Suites numériques

|                            |                               |    |
|----------------------------|-------------------------------|----|
| <b>COURS</b><br>& METHODES | .....                         | 29 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 38 |
|                            | S'entraîner .....             | 39 |
|                            | Préparer un contrôle.....     | 44 |
|                            | Aller plus loin .....         | 45 |
| <b>CORRIGÉS</b>            | .....                         | 48 |

3 Fonctions polynômes du second degré

|                            |                               |    |
|----------------------------|-------------------------------|----|
| <b>COURS</b><br>& METHODES | .....                         | 65 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 71 |
|                            | S'entraîner .....             | 73 |
|                            | Préparer un contrôle.....     | 75 |
|                            | Aller plus loin .....         | 76 |
| <b>CORRIGÉS</b>            | .....                         | 79 |

## ANALYSE

### 4 Nombre dérivé, fonction dérivée

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 95  |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 101 |
|                            | S'entraîner .....             | 104 |
|                            | Préparer un contrôle.....     | 109 |
|                            | Aller plus loin .....         | 111 |
| <b>CORRIGÉS</b>            | .....                         | 114 |

### 5 Applications de la dérivation

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 131 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 136 |
|                            | S'entraîner .....             | 138 |
|                            | Préparer un contrôle.....     | 142 |
|                            | Aller plus loin .....         | 144 |
| <b>CORRIGÉS</b>            | .....                         | 149 |

### 6 La fonction exponentielle

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 167 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 173 |
|                            | S'entraîner .....             | 179 |
|                            | Préparer un contrôle.....     | 163 |
|                            | Aller plus loin .....         | 181 |
| <b>CORRIGÉS</b>            | .....                         | 183 |

### 7 Fonctions trigonométriques

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 199 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 203 |
|                            | S'entraîner .....             | 205 |
|                            | Préparer un contrôle.....     | 207 |
|                            | Aller plus loin .....         | 208 |
| <b>CORRIGÉS</b>            | .....                         | 201 |

# PROBABILITÉS ET STATISTIQUES

## 8 Probabilités conditionnelles

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 227 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 232 |
|                            | S'entraîner .....             | 235 |
|                            | Préparer un contrôle.....     | 237 |
|                            | Aller plus loin .....         | 238 |
| <b>CORRIGÉS</b>            | .....                         | 241 |

## 9 Variables aléatoires

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 257 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 262 |
|                            | S'entraîner .....             | 265 |
|                            | Préparer un contrôle.....     | 267 |
|                            | Aller plus loin .....         | 268 |
| <b>CORRIGÉS</b>            | .....                         | 272 |

# GÉOMÉTRIE

## 10 Calcul vectoriel et produit scalaire

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 293 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 298 |
|                            | S'entraîner .....             | 300 |
|                            | Préparer un contrôle.....     | 302 |
|                            | Aller plus loin .....         | 303 |
| <b>CORRIGÉS</b>            | .....                         | 305 |

## 11 Droites, cercles et paraboles

|                            |                               |     |
|----------------------------|-------------------------------|-----|
| <b>COURS</b><br>& MÉTHODES | .....                         | 319 |
| <b>EXOS</b><br>& SUJETS    | Tester ses connaissances..... | 326 |
|                            | S'entraîner .....             | 329 |
|                            | Préparer un contrôle.....     | 330 |
|                            | Aller plus loin .....         | 332 |
| <b>CORRIGÉS</b>            | .....                         | 334 |

|              |       |     |
|--------------|-------|-----|
| <b>Index</b> | ..... | 351 |
|--------------|-------|-----|

# Algorithmique et programmation

```
mirror_mod = modifier_ob.modifiers.new("MIRROR")
# add mirror object to mirror_ob
mirror_mod.mirror_object = mirror_ob

# operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
# operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
# operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True

# Selection at the end -add back the deselected objects
mirror_ob.select = 1
mirror_ob.select = 1
bpy.context.scene.objects.active = mirror_ob
print("selected" + str(modifier_ob)) # modifier
mirror_ob.select = 0
from bpy.context.selected_objects[0]
bpy.data.objects[one.name].select = 1

print("please select exactly two objects,")
```

OPERATOR CLASSES

```
def __init__(self, context):
    """Initialize the mirror to the selected object"""
    self.mirror_mirror_x = "Mirror X"

    def __init__(self, context):
        """Initialize the mirror to the selected object"""
        self.mirror_mirror_x = "Mirror X"

    def __init__(self, context):
        """Initialize the mirror to the selected object"""
        self.mirror_mirror_x = "Mirror X"
```



## 1

# Prise en main de Python

## I INTRODUCTION

- ▶ On peut installer Python, en allant le télécharger sur le site officiel : <https://www.python.org/>.
- ▶ Lancer Python et créer un nouveau fichier (File> New File).
- ▶ Taper le code :

```
x=2
print(x,'est de type',type(x))

y=1/x
print(«l'inverse de «x,» est «,y)
print(y,' est de type ',type(y))

z=True
print(z,' est de type ',type(z))
```

- ▶ Exécuter-le (Run>Run Module ou F5).

### REMARQUES :

- x est de type entier (int), y est de type flottant (« nombre à virgule flottante », float) et z est de type booléen (boolean).
- En Python, toute ligne précédée du symbole # n'est pas exécutée : c'est un commentaire.

## II FONCTION

→ Voir la méthode 1 et les exos 5 et 7.

### SYNTAXE :

La syntaxe Python pour la définition d'une fonction est la suivante :

```
def f(par1, par2,...) :
    y=
    return sortie1, sortie2
```

où :

- par1, par2,... désignent les paramètres en entrées ;
- sortie1, sortie2,... désignent les sorties retournées par la fonction.



Le décalage des instructions qui suivent la ligne def (on dit l'indentation) est essentiel en Python pour délimiter une fonction : habituellement l'indentation est de quatre espaces en Python.

Une fonction peut avoir plusieurs arguments en entrées :

**EXEMPLE :**

La fonction suivante renvoie la somme et le produit de deux nombres

```
def somme_produit(x,y):
    return x+y, x*y
# Un calcul
print('la somme et le produit de 2 par 3 valent respectivement',
      somme_produit(2,3))
```



Dans le cas où une fonction ne retourne aucune valeur, le return est facultatif.

## III PROGRAMMATION

### 1. Instruction conditionnelle if

→ Voir la méthode 2 et les exos 5 et 7.

**SYNTAXE :**

La syntaxe Python d'une instruction conditionnelle **if** est la suivante:

```
if condition 1 :
    Bloc 1 d'instructions à exécuter
elif condition 2 :
    Bloc 2 d'instructions à exécuter
....
else :
    Bloc d'instructions à exécuter
```



Comme pour les fonctions, après les mots-clés if, elif et else, l'indentation est obligatoire !

**COMPARAISON :**

Pour définir la condition permettant l'exécution de la clause if, on est amené à utiliser les opérateurs de comparaison.

| En Python  | Signification                       |
|------------|-------------------------------------|
| $x < y$    | $x$ est inférieur strictement à $y$ |
| $x \leq y$ | $x$ est inférieur ou égal à $y$     |
| $x > y$    | $x$ est supérieur strictement à $y$ |
| $x \geq y$ | $x$ est supérieur ou égal à $y$     |
| $x == y$   | $x$ est égal à $y$                  |
| $x != y$   | $x$ est différent de $y$            |

On peut aussi « combiner » ces opérateurs entre eux à l'aide des trois opérateurs suivants (appelés opérateurs booléens) : or (ou), and (et), not (non)

**EXEMPLE :**

$x < -2$  ou  $x > 2$  s'écrit en Python :  $(x < -2)$  or  $(x > 2)$

$2 < x < 3$  s'écrit en Python sous la forme  $(2 < x)$  and  $(x < 3)$ , mais aussi :  $2 < x < 3$ .

## 2. Boucle itérative while

→ Voir la méthode 3 et les exos 9 et 10

**SYNTAXE :**

La syntaxe Python d'une boucle itérative **while** est la suivante :

```
while condition :  
    Bloc d'instructions à exécuter
```



Comme pour les fonctions, après le mot-clé **while** l'indentation est obligatoire !

## 3. Boucle itérative For

→ Voir la méthode 4 et les exos 1 et 6.

**SYNTAXE :**

La syntaxe Python d'une boucle itérative **for** est la suivante :

```
for x in range (n):  
    Bloc d'instructions à exécuter
```

où  $\text{range}(n)$  est un « itérateur » qui parcourt les entiers  $0, 1, \dots, n-1$

Comme pour les fonctions, après le mot-clé **for** l'indentation est obligatoire !

# IV LISTES

## 1. Définition élémentaire

Une liste est une collection d'éléments.

**EXEMPLE :**

```
nombres=[1,4,5] # liste d'entiers  
couleurs=['rouge','bleu', 'jaune'] # liste de chaînes de caractères  
melange=[1,'bleu',2,'vert'] # mixte
```

**Remarques :**

- Les éléments d'une liste ne sont pas obligatoirement du même type comme le montre l'exemple précédent.
- Les éléments d'une liste sont indexés à partir de 0 ; ainsi dans la liste `nombres` :
  - `nombres[0]` correspond au premier élément de la liste `nombres`.
  - `nombres[1]` correspond au deuxième élément de la liste `nombres`.
  - `nombres[-1]` correspondant au dernier élément de la liste `nombres`.

## 2. Opérations sur les listes

Le tableau suivant donne quelques méthodes associées aux listes :

| Méthode                               | Signification                                     |
|---------------------------------------|---------------------------------------------------|
| <code>len(liste)</code>               | Renvoie la longueur de liste                      |
| <code>liste.append(x)</code>          | Ajoute l'élément $x$ à la fin de liste            |
| <code>liste.insert(i,x)</code>        | Ajoute l'élément $x$ en position $i$ de liste     |
| <code>liste.count(x)</code>           | Compte le nombre d'occurrences de $x$ dans liste  |
| <code>liste.remove(x)</code>          | Supprime la première occurrence de $x$ dans liste |
| <code>liste.sort()</code>             | Trie la liste dans l'ordre croissant              |
| <code>liste.sort(reverse=True)</code> | Trie la liste dans l'ordre décroissant            |

## 3. Liste définie en compréhension

→ Voir l'exo 8.

On voudrait construire le tableau de valeurs de la fonction carré suivant :

|       |    |    |      |    |      |   |     |   |     |   |   |
|-------|----|----|------|----|------|---|-----|---|-----|---|---|
| $x$   | -3 | -2 | -1,5 | -1 | -0,5 | 0 | 0,5 | 1 | 1,5 | 2 | 3 |
| $x^2$ |    |    |      |    |      |   |     |   |     |   |   |

En Python, on écrit alors :

```
# Tableau de valeurs de la fonction carrée
antecedents=[-3, -2,-1.5,-1,-0.5,0,0.5,1,1.5,2,3]
carres=[x**2 for x in antecedents]
print(carres)
```

### Remarques :

- L'instruction `list(range(n))` crée la liste des entiers de 0 à  $n-1$ .
- La syntaxe générale de la boucle `for` est :

```
for l in liste :
    bloc d'instructions à exécuter
```

On peut aussi associer une condition `if` à cette définition de liste :

### EXEMPLE :

On veut calculer les images de la liste `antecedents` par la fonction inverse ; sachant que l'inverse de 0 n'existe pas, on écrit :

```
inverses=[1/x for x in antecedents if (x!=0)]
print(inverses)
```

**V COMPLÉMENTS****1. Communication avec l'utilisateur**

`print()` : permet d'afficher du texte et (ou) des valeurs de variables. Le texte à afficher doit être compris entre deux guillemets : "texte" ou entre deux apostrophes : 'texte'.

`input()` : permet de demander à l'utilisateur d'entrer une valeur par exemple (et de l'enregistrer) ; elle s'utilise avec `eval` :

**EXEMPLE :**

```
n=eval(input('entrer un nombre '))
print('la multiplication de ',n,'par 3 vaut ',3*n)
```

**2. Modules math et matplotlib**

→ Voir l'exo 3.

► Si Python reconnaît par défaut les **opérateurs arithmétiques** (+, −, \*, /) et les puissances (par exemple, `x**2`), beaucoup de fonctions (comme `cos` et `sin`) et même la constante  $\pi$  ne sont pas reconnues par Python ; elles sont contenues dans le module `math`, qu'il faut importer avant de les utiliser.

**EXEMPLE :**

```
from math import * # importe tous les éléments du module math
print('la racine carrée de 2 vaut environ',sqrt(2))
print('la constante pi vaut environ',pi)
y=cos(pi/4)
print(y)
```

► Pour tracer des courbes représentatives de fonctions, on utilise le module `matplotlib`

**EXEMPLE :**

```
from matplotlib.pyplot import * # importe tous les éléments de matplotlib
antecedents=[-3+i/10 for i in range(60)]
carres=[x**2 for x in antecedents]

plot(antecedents,carres)
show()
close()
print(carres)
```



Pour installer `matplotlib`, sous Windows, ouvrir un terminal (taper `cmd` dans le menu rechercher) puis y entrer :

```
python -m pip install -U matplotlib.
```

**MÉTHODE 1****Écrire un script permettant de calculer les images d'une fonction**

→ Voir les exos 5 et 7.

**Exo résolu**

a. Écrire un script permettant de calculer les images de la fonction  $f$  définie par.

$$f(x) = \begin{cases} 0 & \text{si } x < 0 \\ x & \text{si } x \in [0 ; 1] \\ 1 & \text{si } x > 1 \end{cases}$$

b. Tester ce script en calculant les images de  $-2$  ;  $0,25$  et  $7$ .

**CORRIGÉ**

a.

```
def f(x) :
    if x<0:
        return 0
    elif 0<=x<=1:
        return x
    else:
        return 1
```

**Remarque :** les clauses elif et else sont facultatives.

b.

Calculs de quelques images :

```
print("l'image de -2 par f est",f(-2))
print("l'image de 0.25 par f est",f(0.25))
print("l'image de 7 par f est",f(7))
```

## MÉTHODE 2

## Définir la fonction carré sur Python et la tester

→ Voir l'exo 3.

## Exo résolu

Définir la fonction carré sur Python et la tester en calculant les images de 2 et de -3.

## CORRIGÉ

*# Définition de la fonction carrée*

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

*# Calcul de quelques images de la fonction carré*

```
print('le carré de 2 est ',carre(2))  
print('le carré de -3 est ',carre(3))
```



Comme précisé dans la syntaxe, une fonction peut avoir plusieurs paramètres en entrées et retourner plusieurs sorties.

## MÉTHODE 3

## Écrire un script qui détermine le premier entier vérifiant une condition

→ Voir les exos 9 et 10.

## Exo résolu

Écrire un script qui détermine le premier entier  $n$  tel que  $2^n > 100$ .

## CORRIGÉ

```
n=0  
while (2**n<100):  
    n+=1  
print('le 1er entier n tel que 2^n dépasse 100 est ',n)
```



L'instruction  $n+=1$  est équivalente à  $n=n+1$  (à chaque itération, elle incrémente de 1 la variable  $n$ ).