# CODE QUEST

## AVEC PYTHON

**JEUX | ART | MUSIQUE | IA**



Une introduction sans installation pour les jeunes codeurs curieux de 8 à 12 ans

# Ce livre appartient à

Nom du codeur ou de la codeuse :

Date de début de la quête :

Mon idée de projet :

# Pour les adultes et les accompagnateurs

Ceci est une courte introduction pratique, et non une référence Python complète. Un apprenant peut la terminer en 7 à 14 jours, à raison d'environ 45 à 75 minutes par jour.

Reste à proximité lors de la première visite de l'éditeur en ligne. L'enfant doit utiliser des informations personnelles inventées, éviter les fonctions de discussion et de partage, et demander la permission avant d'ouvrir des liens. Laisse l'apprenant taper, prédire et déboguer. Voici des questions utiles : Qu'est-ce que tu attendais ? Qu'est-ce qui a changé ? Quel est le plus petit test que tu peux essayer ?

Tous les projets utilisent des personnages, des règles, des textes et des formes originaux.

Copyright 2026. Tous droits réservés. Le texte, le code, les personnages et les diagrammes vectoriels ont été créés pour cette édition. L'illustration de couverture est une œuvre originale générée par IA, dirigée pour ce livre. Aucun soutien de la part des sites web mentionnés n'est sous-entendu.

# Ouvre le labo de code

1. Avec la permission d'un adulte, ouvre play.pyroom.app dans un navigateur récent.

2. Choisis une console Python pour les programmes textuels, ou Turtle pour les projets de dessin.

3. Tape le code dans les zones sombres prévues à cet effet. Appuie sur Run (Exécuter). Des questions peuvent apparaître dans la zone de résultat.

4. Si Pyroom n'est pas disponible, utilise un autre éditeur Python 3 récent dans le navigateur pour les projets textuels. Pour Turtle, essaie skulpt.org.

Les sites web peuvent changer. Les leçons enseignent des concepts Python standards et ne nécessitent pas de compte.

# Ton bouclier de sécurité

```
print("Génial !")
```

Demande à un adulte avant de visiter un nouveau site web, de créer un compte, de télécharger un fichier, d'activer une caméra ou un microphone, ou de partager un projet.

Utilise des noms, des lieux, des mots de passe et des informations sur des créatures inventés dans les exercices. Ne colle jamais d'informations privées dans un programme ou un service d'IA.

Les éditeurs en ligne sont des outils fournis par d'autres organisations. Ce livre n'est ni parrainé ni lié à ces dernières. Leur disponibilité et leurs fonctionnalités peuvent changer.

# Carte de la quête

Quêtes 1 à 4 : ordinateurs, algorithmes et l'éditeur en ligne

Quêtes 5 à 8 : variables, saisie, décisions et boucles

Quêtes 9 et 10 : hasard et fonctions

Quêtes 11 à 13 : art avec Turtle, jeux et musique

Quête 14 : l'IA et son usage responsable

Quête 15 : planifie, construis, teste et partage un projet final original

QUÊTE 1

01

TA PREMIÈRE MISSION

# Bienvenue dans Code Quest

**Programmer, c'est donner à un ordinateur une suite d'instructions claires.**

**MISSION**

Lis, exécute, modifie et explique le code. Termine avec le Défi du Boss.

APPRENDS

# Trois idées à découvrir

1. Un ordinateur peut nous aider à dessiner, calculer, jouer des sons, raconter des histoires et créer des jeux. Il ne peut pas deviner ce que nous voulons dire. Il suit les instructions que nous lui donnons.

2. Un programme est un ensemble d'instructions enregistrées. Le code est la façon dont nous écrivons ces instructions dans un langage de programmation. Dans ce livre, notre langage est Python.

3. Tu n'as pas besoin d'être un expert en mathématiques. Tu as besoin de curiosité, de patience et de l'envie d'essayer de petits changements.

*Fais une pause et explique chaque idée avec tes propres mots.*

OBSERVE

# Lis le code comme une histoire

Fais d'abord une prédiction. Puis exécute le code dans l'éditeur en ligne.

```
print("Salut, codeur !")
print("Ma quête commence maintenant.")
```

**UN RÉSULTAT POSSIBLE**

```
Salut, codeur !
Ma quête commence maintenant.
```

**DÉTECTIVE DU CODE**

Montre la partie qui peut changer. Montre la partie qui produit un résultat. À ton avis, à quoi sert chaque signe de ponctuation ?

MODIFIE-LE

# À toi d'essayer : petits changements, grandes découvertes

1. Change codeur par ton prénom ou un nom de héros inventé.

2. Ajoute une troisième ligne print avec un objectif pour ta quête.

3. Prédis le résultat avant d'appuyer sur Run.

**MA PRÉDICTION**

_____

_____

_____

**BOUCLIER DE SÉCURITÉ**

Utilise des noms et des informations inventés. Demande à un adulte de confiance avant de visiter un nouveau site web ou de partager le lien d'un projet.

CHASSE AUX BUGS

# Le programme de Pip a un bug !

Exécute la version boguée. Lis le message d'erreur ou le résultat étrange. Entoure le plus petit élément que tu penses devoir changer.

```
print("Salut, codeur !)
```

**TA CORRECTION**

**VÉRIFIE APRÈS AVOIR ESSAYÉ**

```
print("Salut, codeur !")
```

Méthode de débogage : observer -> deviner -> changer une chose -> réexécuter.

# Point de sauvegarde

Trois mots à garder dans ton sac à dos de codeur

**programme**
un ensemble d'instructions pour l'ordinateur

**code**
des instructions écrites dans un langage de programmation

**exécuter (run)**
demander à l'ordinateur de suivre le code

**DÉFI DU BOSS**

**Crée une scène d'ouverture de cinq lignes pour un jeu original.**

Quand ça fonctionne, ajoute tes initiales ici :

**RÉFLÉCHIS**

- Qu'ai-je découvert ?
- Qu'ai-je créé ?
- Que changerais-je la prochaine fois ?

QUÊTE 2

02

ENTRÉE, TRAITEMENT, SORTIE

# Qu'est-ce qu'un ordinateur ?

**Un ordinateur reçoit des données, les traite, les stocke et produit un résultat.**

**MISSION**

Lis, exécute, modifie et explique le code. Termine avec le Défi du Boss.

APPRENDS

# Trois idées à découvrir

1. Un clavier, une souris, un microphone, une caméra et une manette de jeu peuvent fournir des données. Une entrée (input) est une information envoyée à un ordinateur.

2. Le processeur suit les instructions. La mémoire conserve les éléments utilisés en ce moment. Le stockage garde les fichiers pour plus tard, même une fois l'appareil éteint.

3. Un écran, un haut-parleur, une imprimante et une manette vibrante peuvent produire une sortie (output). Un même appareil peut être à la fois une entrée et une sortie : un écran tactile affiche des images et détecte les touchers.

*Fais une pause et explique chaque idée avec tes propres mots.*

OBSERVE

# Lis le code comme une histoire

Fais d'abord une prédiction. Puis exécute le code dans l'éditeur en ligne.

```
player = input("Nom du héros : ")
print("Bienvenue,", player)
```

**UN RÉSULTAT POSSIBLE**

```
Nom du héros : Nova
Bienvenue, Nova
```

**DÉTECTIVE DU CODE**

Montre la partie qui peut changer. Montre la partie qui produit un résultat. À ton avis, à quoi sert chaque signe de ponctuation ?

MODIFIE-LE

# À toi d'essayer : petits changements, grandes découvertes

1. Nomme deux entrées utilisées par un jeu.

2. Nomme deux sorties utilisées par une application musicale.

3. Dessine des flèches montrant entrée -> traitement -> sortie.

**MA PRÉDICTION**

_____

_____

_____

**BOUCLIER DE SÉCURITÉ**

Utilise des noms et des informations inventés. Demande à un adulte de confiance avant de visiter un nouveau site web ou de partager le lien d'un projet.

CHASSE AUX BUGS

# Le programme de Pip a un bug !

Exécute la version boguée. Lis le message d'erreur ou le résultat étrange. Entoure le plus petit élément que tu penses devoir changer.

```
player = input("Nom du héros : ")
print("Bienvenue," player)
```

**TA CORRECTION**

**VÉRIFIE APRÈS AVOIR ESSAYÉ**

```
player = input("Nom du héros : ")
print("Bienvenue,", player)
```

Méthode de débogage : observer -> deviner -> changer une chose -> réexécuter.

# Point de sauvegarde

Trois mots à garder dans ton sac à dos de codeur

**matériel (hardware)**
les parties physiques que tu peux toucher

**logiciel (software)**
les programmes et leurs données

**mémoire**
un espace de travail temporaire

**DÉFI DU BOSS**

**Invente un ordinateur caché dans un sac à dos. Liste son entrée, son traitement, son stockage et sa sortie.**
Quand ça fonctionne, ajoute tes initiales ici :

**RÉFLÉCHIS**

- Qu'ai-je découvert ?
- Qu'ai-je créé ?
- Que changerais-je la prochaine fois ?

QUÊTE 3

# 03

PENSE COMME UN ROBOT

# Algorithmes et bugs

**Un algorithme est un plan ordonné pour résoudre un problème.**

**MISSION**

Lis, exécute, modifie et explique le code. Termine avec le Défi du Boss.

APPRENDS

# Trois idées à découvrir

1. Imagine que tu dois expliquer à un robot comment dessiner un carré. « Dessine un carré » n'est pas assez précis. Le robot a besoin d'étapes : avancer, tourner, avancer, tourner, et ainsi de suite.

2. Un bug est une erreur dans le code ou dans le plan. Déboguer signifie trouver la cause, changer une chose, puis retester.

3. Les bons programmeurs n'évitent pas tous les bugs. Ils apprennent à lire les indices, à faire un petit changement, puis à réexécuter le programme.

*Fais une pause et explique chaque idée avec tes propres mots.*

OBSERVE

# Lis le code comme une histoire

Fais d'abord une prédiction. Puis exécute le code dans l'éditeur en ligne.

```
step = 1
print("Étape", step, ": ouvrir la carte")
step = step + 1
print("Étape", step, ": choisir un chemin")
```

**UN RÉSULTAT POSSIBLE**

```
Étape 1 : ouvrir la carte
Étape 2 : choisir un chemin
```

**DÉTECTIVE DU CODE**

Montre la partie qui peut changer. Montre la partie qui produit un résultat. À ton avis, à quoi sert chaque signe de ponctuation ?

MODIFIE-LE

# À toi d'essayer : petits changements, grandes découvertes

1. Écris les étapes pour se brosser les dents sans utiliser le mot « etcetera ».

2. Demande à quelqu'un de suivre tes étapes exactement.

3. Entoure toute étape qui a besoin de plus de détails.

**MA PRÉDICTION**

_____

_____

_____

**BOUCLIER DE SÉCURITÉ**

Utilise des noms et des informations inventés. Demande à un adulte de confiance avant de visiter un nouveau site web ou de partager le lien d'un projet.

CHASSE AUX BUGS

# Le programme de Pip a un bug !

Exécute la version boguée. Lis le message d'erreur ou le résultat étrange. Entoure le plus petit élément que tu penses devoir changer.

```
print("Premièrement")
print("Troisièmement")
print("Deuxièmement")
```

**TA CORRECTION**

**VÉRIFIE APRÈS AVOIR ESSAYÉ**

```
print("Premièrement")
print("Deuxièmement")
print("Troisièmement")
```

Méthode de débogage : observer -> deviner -> changer une chose -> réexécuter.

# Point de sauvegarde

Trois mots à garder dans ton sac à dos de codeur

**algorithme**
un plan ordonné

**bug**
une erreur qui cause un problème

**déboguer**
trouver et réparer la cause d'un problème

**DÉFI DU BOSS**

**Écris un algorithme qui guide Pip à travers un labyrinthe de quatre cases.**

Quand ça fonctionne, ajoute tes initiales ici :

**RÉFLÉCHIS**

- Qu'ai-je découvert ?
- Qu'ai-je créé ?
- Que changerais-je la prochaine fois ?