

La programmation en sciences de la nature

CHAPITRE 1 INTRODUCTION GÉNÉRAL

1 Introduction général

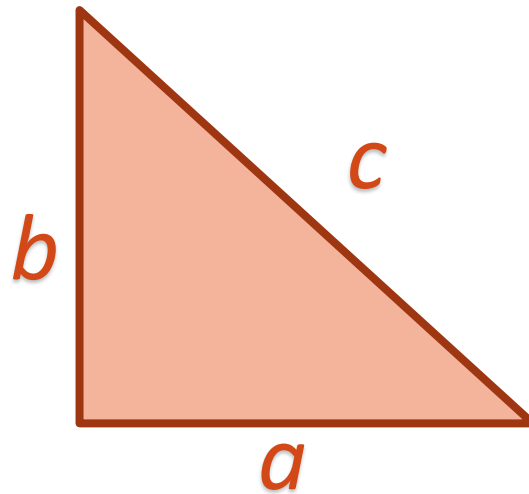
Qu'est-ce que la programmation ?

Qu'est-ce que les sciences numériques ?

1.1 Introduction à l'analyse numérique

1.1 Introduction à l'analyse numérique

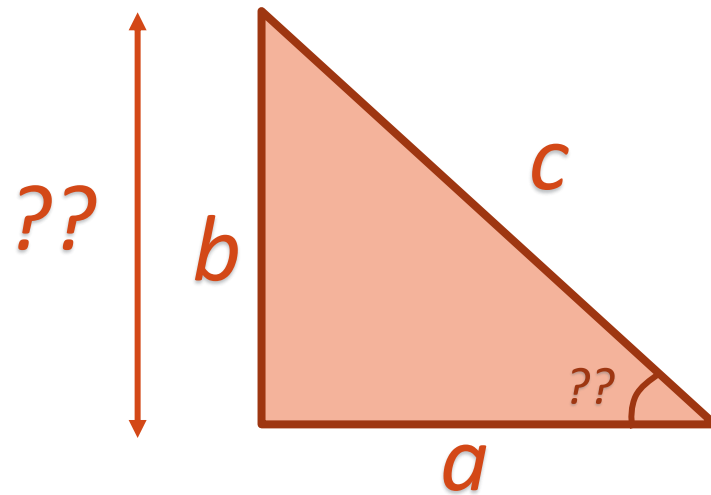
Scientifique théorique : définir les propriétés d'un concept (Exemple le triangle rectangle) à l'aide de langage mathématique pour définir les propriétés.



$$a^2 + b^2 = c^2$$

1.1 Introduction à l'analyse numérique

Scientifique expérimentale : développer des outils/méthodes/protocoles pour construire les triangles rectangles et d'en mesurer les propriétés avec la meilleure précision possible.



$$a^2 + b^2 = c^2$$

1.1 Introduction à l'analyse numérique

Sciences numériques : Construire des algorithmes et des dispositifs de calcul efficaces.

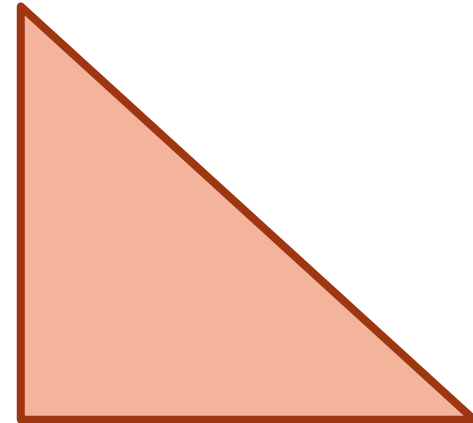
Algorithme

```
# Exemple d'utilisation
height = 5
draw_right_triangle(height)

def draw_right_triangle(height):
    for i in range(1, height + 1):
        print('*' * i)
```

Résultat

```
*
**
***
****
*****
```



1.1 Introduction à l'analyse numérique

L'analyse numérique est une branche des mathématiques qui s'intéresse au développement des **méthodes de calcul qui permettent de résoudre des problèmes mathématiques à l'aide de nombre.**

1.1 Introduction à l'analyse numérique

L'analyse numérique est une branche des mathématiques qui s'intéresse au développement des **méthodes de calcul qui permettent de résoudre des problèmes mathématiques à l'aide de nombre.**

En analyse numérique on cherche un chemin efficace pour obtenir une valeur suffisamment proche de la réponse.

1.1 Introduction à l'analyse numérique

Exemple, créons un algorithme pour la fonction suivante: \sqrt{a}

1.1 Introduction à l'analyse numérique

Exemple, créons un algorithme pour la fonction suivante: \sqrt{a}

Étape 1. Choisir une valeur $x_0 > 0$

Étape 2. Calculer la valeur telle que: $x = \frac{1}{2} \left(x_0 + \frac{a}{x_0} \right)$

1.1 Introduction à l'analyse numérique

Exemple, créons un algorithme pour la fonction suivante: \sqrt{a}

Étape 1. Choisir une valeur $x_0 > 0$

Étape 2. Calculer la valeur telle que: $x = \frac{1}{2} \left(x_0 + \frac{a}{x_0} \right)$

Étape 3. Redéfinir $x_0 = x$ puis recalculer x avec l'équation de l'étape 2.

1.1 Introduction à l'analyse numérique

Exemple, créons un algorithme pour la fonction suivante: \sqrt{a}

Étape 1. Choisir une valeur $x_0 > 0$

Étape 2. Calculer la valeur telle que: $x = \frac{1}{2} \left(x_0 + \frac{a}{x_0} \right)$

Étape 3. Redéfinir $x_0 = x$ puis recalculer x avec l'équation de l'étape 2.

Étape 4. Répéter l'étape 3 jusqu'à l'obtention de la précision désiré.

1.1 Introduction à l'analyse numérique

Exemple, créons un algorithme pour la fonction suivante: \sqrt{a}

```
# Exemple d'utilisation
a = 10
resultat = racine_carree_heron(a)
print(f"La racine carrée de {a} est approximativement {resultat}")

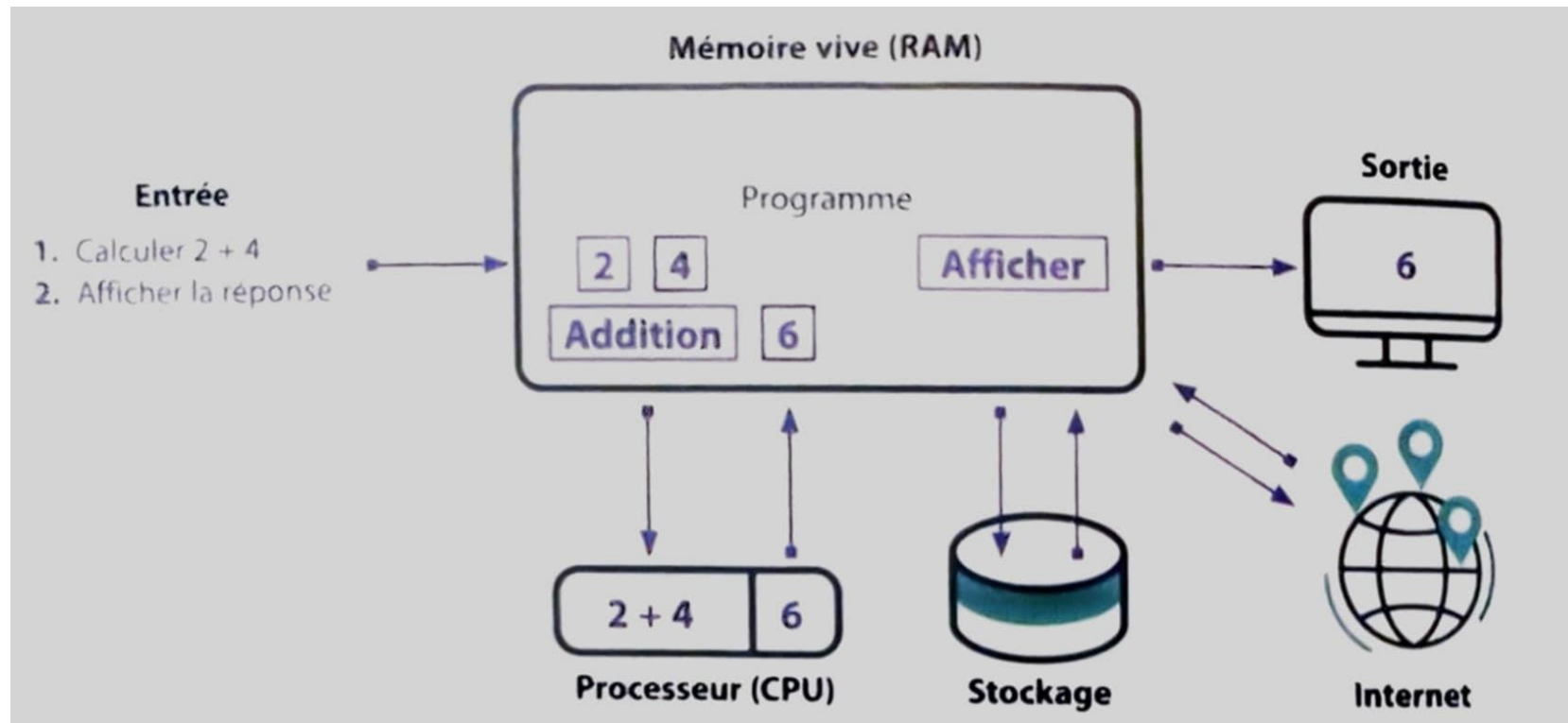
def racine_carree_heron(a, precision=1e-10):
    x0 = a / 2 # Choisir une valeur de départ
    while True:
        x = (x0 + a / x0) / 2 # Calculer la valeur de x
        if abs(x - x0) < precision: # Vérifier si la précision est atteinte
            return x
        x0 = x # Redéfinir x0 pour la prochaine itération
```

Résultat

```
La racine carrée de 10 est approximativement 3.162277660168379
```

1.2 L'informatique et la programmation

1.2 L'informatique et la programmation



1.2 L'informatique et la programmation

Éditeur

```
# Programme pour  
afficher le résultat  
d'une addition
```

```
x = 2 + 4
```

```
print(x)
```


1.2 L'informatique et la programmation

Éditeur

```
# Programme pour  
afficher le résultat  
d'une addition
```

x représente une variable.

```
x = 2 + 4
```

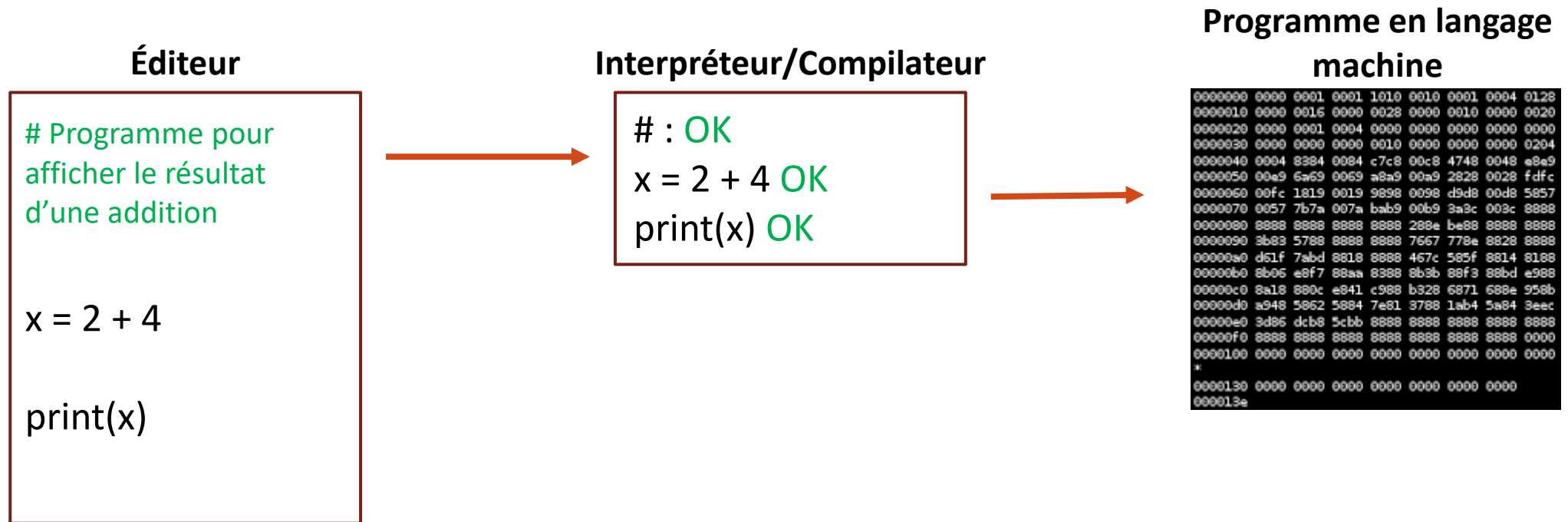
print est la fonction de base
pour afficher à l'écran.

```
print(x)
```

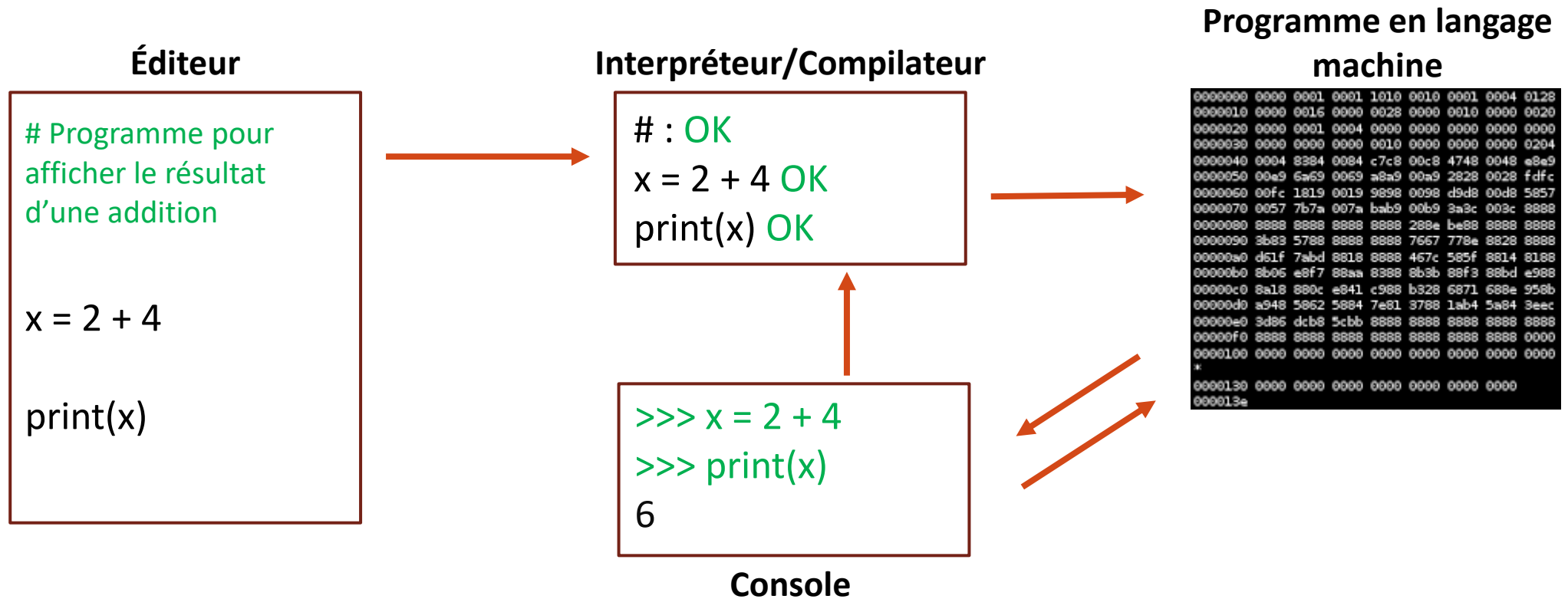
Variable : caractéristique mesurable
qui peut prendre différentes valeurs.

Fonction : portion de code
informatique nommée, qui accomplit
une tâche spécifique.

1.2 L'informatique et la programmation

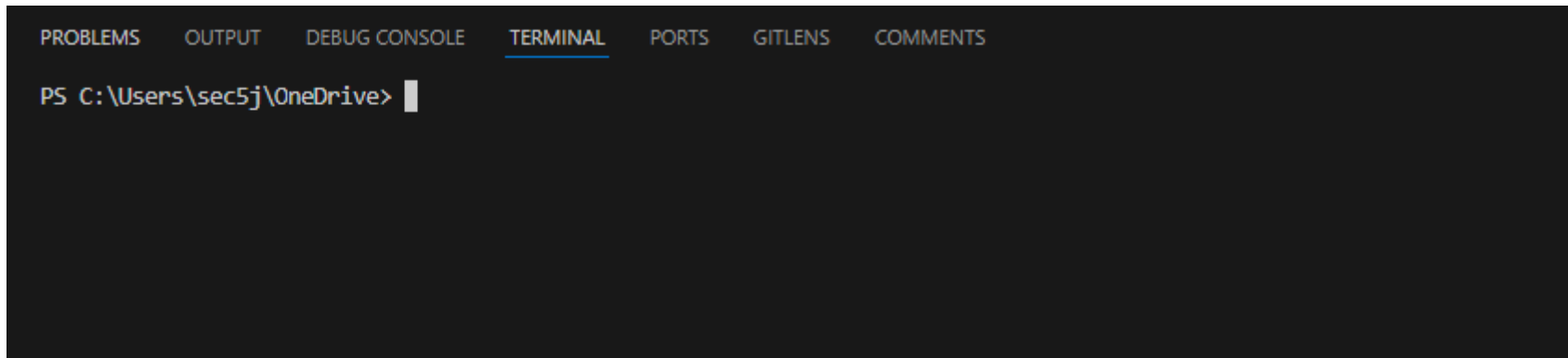


1.2 L'informatique et la programmation



1.4 L'environnement de développement intégré

La **console** ou aussi appelé shell (en anglais) ou terminal, permet **d'entrer des instructions et d'afficher vos résultats**.

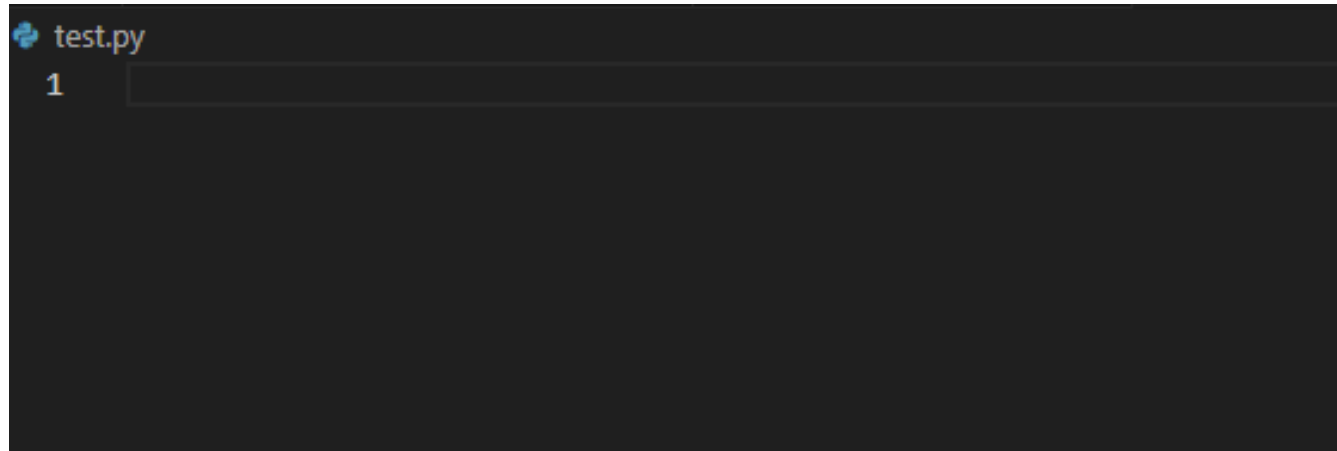
A screenshot of a terminal window within an IDE. The terminal has a dark background with light-colored text. At the top, there is a horizontal menu bar with several tabs: PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is currently selected and underlined), PORTS, GITLENS, and COMMENTS. Below the menu bar, the terminal displays the command prompt 'PS C:\Users\sec5j\OneDrive>' followed by a white cursor block.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS  COMMENTS

PS C:\Users\sec5j\OneDrive> 
```

1.4 L'environnement de développement intégré

Lorsque nos programmes comportent plusieurs lignes d'instruction, il est préférable d'utiliser **l'éditeur de programme**. Celui-ci peut être vu comme un **traitement de texte spécialisé pour la programmation**. Il faut noter que le code de nos programmes s'exécute de façon procédurale (on exécute les lignes de haut en bas).



1.4 L'environnement de développement intégré

La console peut également afficher des messages d'erreurs. Voici un exemple d'erreur ou il manque une parenthèse dans notre code.

```
x = 2 + 4
print(x

>>>
Traceback (most recent call last):
  ❶ File "Programme 1.4-2.py", line 2
    print(x
        ^ ❷
  ❸ SyntaxError: '(' was never closed
```