Python Syntax Basics

Python is designed for readability and simplicity, making it an excellent language for beginners. Let's dive into its core syntax:

---

1. Writing and Running Python Scripts

Writing a Python Script

A Python script is simply a text file with the `.py` extension that contains Python code.

Steps to Write and Run a Script:

1. **Create a Script File:**
    - Use any text editor or IDE. For example, create a file called `script.py`.

Example content of `script.py`:
print("Hello, World!")
print("Welcome to Python Programming!")

    -
2. **Run the Script:**

Using Command Line/Terminal:
Navigate to the directory where the file is saved and run:
python script.py

    -
    - **Using an IDE (e.g., VS Code or PyCharm):**
        - Open the file in the IDE and press the "Run" button or shortcut (usually `F5` or `Ctrl+Shift+B`).

---

2. Comments in Python

Comments are non-executable lines of code that help explain what the code does. They are ignored by the Python interpreter.

Types of Comments:

Single-line Comments:
Use the # symbol.

```
# This is a single-line comment
print("Hello, World!")  # Inline comment
```

-

Multi-line Comments:
Use triple quotes (`''' '''` or `""" """`). While not strictly comments, they're often used for this purpose.

```
"""
This is a multi-line comment.
It can span multiple lines.
"""
print("Python is fun!")
```

-

---

3. Indentation Rules in Python

Python uses indentation to define blocks of code. This makes the code visually organized and enforces good coding practices.

Key Points:
No Braces or Keywords for Block Definition:
Instead of `{}` or keywords like `begin`/`end`, Python uses indentation to denote a block.

```
if 5 > 2:
    print("Five is greater than two!")  # Indented block
```

1.

Consistency is Key:
Use either **spaces** or **tabs**, but not both. The recommended standard is 4 spaces per indentation level.

```
# Correct indentation
for i in range(5):
    print(i)
```

2.

3. **Indentation Errors:**
   Mixing spaces and tabs or missing indentation will result in an `IndentationError`.

Nested Blocks:

Each deeper level of code requires additional indentation.

```
x = 10
if x > 5:
    print("x is greater than 5")
    if x > 8:
        print("x is also greater than 8")
```

Examples: Combining Syntax Elements

Single Script with Comments and Proper Indentation:

```
 # Program to check if a number is positive
num = 10  # Define a number

if num > 0:  # Check if the number is positive
    print("The number is positive")  # Indented block
else:
    print("The number is not positive")
```

1.

Indentation Error Example:

```
 if True:
print("This will raise an IndentationError")
```

2.

By following these basics, you'll be able to write clear and executable Python programs!