

# Python Programming - Cú Pháp Cơ Bản

**Mục tiêu học tập:** Nắm vững các khái niệm nền tảng của Python: biến, kiểu dữ liệu, toán tử và cách tương tác với người dùng.

## 1. Biến và Kiểu Dữ Liệu

### 1.1. Khái niệm Biến

#### Đặt vấn đề

Khi viết chương trình, chúng ta cần lưu trữ thông tin. Ví dụ: tuổi của một người, tên của họ, hoặc điểm số trong một bài kiểm tra. Làm thế nào để máy tính "nhớ" được những thông tin này?

#### Giải quyết

**Biến (variable)** là một "cái hộp" trong bộ nhớ máy tính, nơi chúng ta có thể lưu trữ dữ liệu. Mỗi biến có:

- **Tên:** để ta gọi và sử dụng
- **Giá trị:** dữ liệu được lưu trữ bên trong

```
# Tạo biến bằng cách gán giá trị
age = 25
name = "Nguyễn Minh Đạo"
is_student = True

# Sử dụng biến
print(age)      # In ra: 25
print(name)     # In ra: Nguyễn Minh Đạo
```

#### Quy tắc đặt tên biến

Python có một số quy tắc bắt buộc và khuyến nghị:

##### Quy tắc bắt buộc:

- Chỉ chứa chữ cái (a-z, A-Z), số (0-9) và dấu gạch dưới (\_)
- **Không được bắt đầu bằng số**
- **Phân biệt chữ hoa/thường:** `age` và `Age` là hai biến khác nhau
- Không được trùng với từ khóa của Python (như `if`, `for`, `class`, v.v.)

##### Quy ước đặt tên (PEP 8):

- Dùng **snake\_case**: `student_name`, `total_score`
- Tên biến nên có ý nghĩa: `age` tốt hơn `a`, `student_count` tốt hơn `sc`

```
# ☒ Hợp lệ
student_age = 20
_temp = 100
number2 = 50

# ☐ Không hợp lệ
2number = 50      # Lỗi: bắt đầu bằng số
student-name = ""  # Lỗi: có dấu gạch ngang
for = 10           # Lỗi: trùng từ khóa
```

## Gán và Gán lại

Python cho phép thay đổi giá trị biến bất cứ lúc nào:

```
score = 85
print(score)    # 85

score = 90      # Gán lại giá trị mới
print(score)    # 90

# Thậm chí có thể đổi kiểu dữ liệu
score = "Excellent"
print(score)    # Excellent
```

**Lưu ý:** Trong Python, biến không cần khai báo kiểu dữ liệu trước (dynamic typing).

---

## 1.2. Kiểu Dữ Liệu Cơ Bản

### Đặt vấn đề

Máy tính cần phân biệt các loại dữ liệu khác nhau để xử lý đúng cách. Ví dụ:

- Tính toán với số:  $10 + 5 = 15$
- Ghép chuỗi: `"Hello" + " World" = "Hello World"`

Làm sao Python biết cách xử lý?

### Giải quyết: 4 Kiểu Dữ Liệu Cơ Bản

#### 1.2.1. Integer (int) - Số Nguyên

Đại diện cho các số nguyên (không có phần thập phân).

```
age = 25
year = 2025
temperature = -10
```

```
# Python hỗ trợ số nguyên lớn tùy ý
big_number = 123456789012345678901234567890

print(type(age))    # <class 'int'>
```

### Ứng dụng thực tế:

- Đếm số lượng: số học sinh, số sản phẩm
- Tuổi, năm sinh
- Index của mảng/list

---

### 1.2.2. Float - Số Thực

Đại diện cho số có phần thập phân.

```
height = 1.75      # Chiều cao (m)
pi = 3.14159
temperature = 36.5

# Ký hiệu khoa học
speed_of_light = 3e8    #  $3 \times 10^8 = 300,000,000$ 

print(type(height))    # <class 'float'>
```

### ⚠ Lưu ý về độ chính xác:

```
result = 0.1 + 0.2
print(result)          # 0.30000000000000004 (không chính xác tuyệt đối)
print(result == 0.3)   # False

# So sánh float nên dùng epsilon
epsilon = 1e-10
print(abs(result - 0.3) < epsilon) # True
```

### Ứng dụng:

- Tính toán khoa học: trọng lượng, chiều cao
- Tiền tệ (chú ý làm tròn)
- Tọa độ, góc độ

---

### 1.2.3. String (str) - Chuỗi Ký Tự

Chuỗi ký tự được bao bằng dấu nháy đơn ' hoặc nháy kép ''.

```
name = "Nguyễn Minh Đạo"
message = 'Hello, World!'
address = """Số 123, đường ABC,
Quận 1, TP.HCM"""          # Chuỗi nhiều dòng

print(type(name))           # <class 'str'>
```

## Các thao tác cơ bản với String

### 1. Nối chuỗi (Concatenation):

```
first_name = "Minh Đạo"
last_name = "Nguyễn"
full_name = last_name + " " + first_name
print(full_name)           # Nguyễn Minh Đạo
```

### 2. Lặp chuỗi:

```
laugh = "ha" * 3
print(laugh)               # hahaha
```

### 3. Truy cập ký tự (Indexing):

```
text = "Python"
print(text[0])             # P (ký tự đầu tiên)
print(text[-1])            # n (ký tự cuối cùng)
print(text[2])             # t
```

### 4. Cắt chuỗi (Slicing):

```
text = "Python Programming"
print(text[0:6])           # Python
print(text[7:])            # Programming
print(text[:6])            # Python
print(text[-11:])          # Programming
```

### 5. Độ dài chuỗi:

```
message = "Hello"
print(len(message))        # 5
```

## Escape Characters - Ký tự đặc biệt

```
# \n: xuống dòng
print("Line 1\nLine 2")

# \t: tab
print("Name:\tĐạo")

# \\: dấu backslash
print("C:\\Users\\Desktop")

# \': nháy đơn trong chuỗi nháy đơn
print('It\'s a beautiful day')
```

## String Formatting - Định dạng chuỗi

```
name = "Nguyễn Minh Đạo"
age = 25
score = 8.75

# Cách 1: f-string (Python 3.6+) - KHUYẾN NGHỊ
message = f"Tên: {name}, Tuổi: {age}, Điểm: {score}"
print(message)

# Cách 2: format()
message = "Tên: {}, Tuổi: {}, Điểm: {}".format(name, age, score)
print(message)

# Cách 3: % formatting (cũ)
message = "Tên: %s, Tuổi: %d, Điểm: %.2f" % (name, age, score)
print(message)
```

## Định dạng số trong f-string:

```
pi = 3.14159265359

print(f"{pi:.2f}")      # 3.14 (2 chữ số thập phân)
print(f"{pi:.4f}")      # 3.1416 (4 chữ số)

number = 1234567
print(f"{number:,}")    # 1,234,567 (thêm dấu phẩy)

percentage = 0.859
print(f"{percentage:.1%}") # 85.9% (phần trăm)
```

### 1.2.4. Boolean (bool) - Logic

Chỉ có 2 giá trị: **True** (đúng) hoặc **False** (sai).

```
is_student = True
has_passed = False
is_adult = True

print(type(is_student)) # <class 'bool'>
```

#### Chuyển đổi sang Boolean:

```
# Các giá trị "falsy" (được coi là False)
print(bool(0))           # False
print(bool(0.0))         # False
print(bool(""))          # False (chuỗi rỗng)
print(bool(None))        # False

# Các giá trị "truthy" (được coi là True)
print(bool(1))           # True
print(bool(-5))          # True
print(bool("Hello"))     # True
print(bool([1, 2]))      # True (list không rỗng)
```

#### Ứng dụng:

- Điều kiện **if**: kiểm tra đúng/sai
- Vòng lặp **while**: tiếp tục khi còn True
- Cờ (flag) trong chương trình

---

## 1.3. Kiểm Tra và Chuyển Đổi Kiểu Dữ Liệu

### Kiểm tra kiểu

```
age = 25
name = "Nguyễn Minh Đạo"

print(type(age))           # <class 'int'>
print(type(name))          # <class 'str'>

print(isinstance(age, int)) # True
print(isinstance(name, str)) # True
print(isinstance(age, str)) # False
```

### Chuyển đổi kiểu (Type Casting)

```
# String → Integer
age_str = "25"
age_int = int(age_str)
print(age_int + 5)      # 30

# Integer → String
number = 100
text = str(number)
print(text + "50")      # 10050 (nối chuỗi, không phải cộng)

# String → Float
price_str = "19.99"
price_float = float(price_str)
print(price_float * 2)  # 39.98

# Float → Integer (cắt phần thập phân)
score = 8.9
rounded = int(score)
print(rounded)          # 8

# Boolean → Integer
print(int(True))        # 1
print(int(False))       # 0
```

### ⚠ Lỗi thường gặp:

```
# Lỗi khi chuyển đổi không hợp lệ
number = int("abc")      # ValueError: invalid literal for int()
number = int("12.5")     # ValueError: phải dùng float() trước
```

## 2. Toán Tử (Operators)

### 2.1. Toán Tử Số Học

#### 🧠 Đặt vấn đề

Làm thế nào để thực hiện các phép tính toán học cơ bản trong Python?

#### 💡 Các toán tử số học

Toán tử	Ý nghĩa	Ví dụ	Kết quả
+	Cộng	10 + 3	13
-	Trừ	10 - 3	7
*	Nhân	10 * 3	30

Toán tử	Ý nghĩa	Ví dụ	Kết quả
/	Chia (luôn trả về float)	10 / 3	3.3333...
//	Chia lấy phần nguyên	10 // 3	3
%	Chia lấy phần dư	10 % 3	1
**	Lũy thừa	2 ** 3	8

### Ví dụ thực hành:

```

a = 10
b = 3

print(f"{a} + {b} = {a + b}")      # 10 + 3 = 13
print(f"{a} - {b} = {a - b}")      # 10 - 3 = 7
print(f"{a} * {b} = {a * b}")      # 10 * 3 = 30
print(f"{a} / {b} = {a / b}")      # 10 / 3 = 3.3333333333333335
print(f"{a} // {b} = {a // b}")    # 10 // 3 = 3
print(f"{a} % {b} = {a % b}")      # 10 % 3 = 1
print(f"{a} ** {b} = {a ** b}")    # 10 ** 3 = 1000

```

## 🔗 Ứng dụng thực tế

### 1. Chia lấy phần nguyên (//):

```

# Tính số ngày từ tổng số giờ
total_hours = 50
days = total_hours // 24
print(f"{total_hours} giờ = {days} ngày")  # 50 giờ = 2 ngày

```

### 2. Chia lấy dư (%):

```

# Kiểm tra số chẵn/lẻ
number = 17
if number % 2 == 0:
    print("Số chẵn")
else:
    print("Số lẻ")          # In ra: Số lẻ

# Tìm số giờ còn lại
total_hours = 50
remaining_hours = total_hours % 24
print(f"Số giờ còn lại: {remaining_hours}")  # 2 giờ

```

### 3. Lũy thừa (\*\*):



```
# Tính diện tích hình vuông
side = 5
area = side ** 2
print(f"Diện tích: {area} m²") # 25 m²

# Tính căn bậc hai
number = 16
sqrt = number ** 0.5
print(f"Căn bậc hai của {number} là {sqrt}") # 4.0
```

## Thứ tự ưu tiên toán tử

Python tuân theo quy tắc toán học (PEMDAS):

1. **P**arentheses (Ngoặc đơn)
2. **E**xponentiation (Lũy thừa)
3. **M**ultiplication/**D**ivision (Nhân/Chia)
4. **A**ddition/**S**ubtraction (Cộng/Trừ)

```
result = 2 + 3 * 4      # 14 (nhân trước, cộng sau)
result = (2 + 3) * 4    # 20 (ngoặc đơn ưu tiên)
result = 2 ** 3 ** 2    # 512 (2^(3^2) = 2^9, từ phải qua trái)
result = 10 / 2 * 3     # 15.0 (trái qua phải với cùng độ ưu tiên)
```

---

## 2.2. Toán Tử Gán (Assignment Operators)

### Gán cơ bản

```
x = 10      # Gán giá trị 10 cho x
```

### Gán kết hợp

```
x = 10

x += 5      # x = x + 5 → x = 15
x -= 3      # x = x - 3 → x = 12
x *= 2      # x = x * 2 → x = 24
x /= 4      # x = x / 4 → x = 6.0
x //= 2     # x = x // 2 → x = 3.0
x %= 2      # x = x % 2 → x = 1.0
x **= 3     # x = x ** 3 → x = 1.0
```

Tại sao dùng toán tử gán kết hợp?

- Code ngắn gọn hơn
- Thể hiện ý định rõ ràng: "tăng x lên 5"
- Trong một số trường hợp, hiệu suất tốt hơn

2.3. Toán Tử So Sánh (Comparison Operators)

🤖 Đặt vấn đề

Làm sao để so sánh hai giá trị? Kiểm tra một học sinh có đủ điểm đỗ không? Một người có đủ tuổi trưởng thành chưa?

💡 Các toán tử so sánh

Kết quả luôn là **True** hoặc **False**.

Toán tử	Ý nghĩa	Ví dụ	Kết quả
==	Bằng	5 == 5	True
!=	Khác	5 != 3	True
>	Lớn hơn	5 > 3	True
<	Nhỏ hơn	5 < 3	False
>=	Lớn hơn hoặc bằng	5 >= 5	True
<=	Nhỏ hơn hoặc bằng	3 <= 5	True

Ví dụ thực hành:

```
age = 18
passing_score = 50
your_score = 75

print(age >= 18)           # True (đủ tuổi trưởng thành)
print(your_score >= passing_score) # True (đỗ)
print(your_score == 100)   # False (không phải điểm tuyệt đối)
print(your_score != 0)     # True (không bị điểm 0)
```

⚠️ Phân biệt = và ==

```
x = 5           # Gán: cho x giá trị 5
x == 5          # So sánh: kiểm tra x có bằng 5 không?

# Lỗi thường gặp
```

```
if x = 5: # ✗ SyntaxError
if x == 5: # ✔ Đúng
```

## So sánh chuỗi

```
# So sánh theo thứ tự từ điển (lexicographic)
print("apple" < "banana") # True
print("Python" == "Python") # True
print("Python" == "python") # False (phân biệt hoa/thường)

# So sánh độ dài
name1 = "An"
name2 = "Bình"
print(len(name1) < len(name2)) # True
```

## So sánh nhiều điều kiện

```
x = 5

# Cách 1: Dùng and
print(0 < x < 10) # True (x trong khoảng 0 đến 10)

# Tương đương với:
print(0 < x and x < 10) # True
```

---

## 2.4. Toán Tử Logic (Logical Operators)

### 🤔 Đặt vấn đề

Làm sao để kiểm tra nhiều điều kiện cùng lúc? Ví dụ: một học sinh vừa phải đủ tuổi VÀ đạt điểm tối thiểu mới được vào lớp nâng cao.

### 💡 Các toán tử logic

Toán tử	Ý nghĩa	Ví dụ
and	VÀ (cả hai đều đúng)	True and True → True
or	HOẶC (ít nhất một đúng)	True or False → True
not	PHỦ ĐỊNH (đảo ngược)	not True → False

### Bảng chân trị:

```
# AND - chỉ True khi CẢ HAI đều True
print(True and True)    # True
print(True and False)   # False
print(False and True)   # False
print(False and False)  # False

# OR - True khi ÍT NHẤT MỘT True
print(True or True)     # True
print(True or False)    # True
print(False or True)    # True
print(False or False)   # False

# NOT - đảo ngược
print(not True)         # False
print(not False)        # True
```

## Ứng dụng thực tế

### 1. Kiểm tra nhiều điều kiện với **and**:

```
age = 20
score = 85
min_age = 18
min_score = 70

# Đủ tuổi VÀ đạt điểm tối thiểu
if age >= min_age and score >= min_score:
    print("Đủ điều kiện vào lớp nâng cao")
else:
    print("Không đủ điều kiện")
```

### 2. Kiểm tra điều kiện thay thế với **or**:

```
discount_code = "SALE50"
is_vip = True

# Có mã giảm giá HOẶC là VIP thì được giảm giá
if discount_code == "SALE50" or is_vip:
    print("Được giảm giá!")
```

### 3. Phủ định với **not**:

```
is_raining = False

if not is_raining:
    print("Không mưa, có thể đi chơi")
```

```
# Kiểm tra chuỗi không rỗng
username = ""
if not username:
    print("Vui lòng nhập tên đăng nhập")
```

## Thứ tự ưu tiên

```
# Thứ tự: not > and > or
result = True or False and False
# Tương đương: True or (False and False)
# Kết quả: True

# Nên dùng ngoặc đơn để rõ ràng
result = (True or False) and False # False
```

## Short-circuit Evaluation (Đánh giá ngắn mạch)

```
# and: nếu vế trái False, không kiểm tra vế phải
x = 0
result = x != 0 and 10 / x > 5 # Không lỗi vì không thực hiện phép chia

# or: nếu vế trái True, không kiểm tra vế phải
user_input = "admin"
is_valid = user_input == "admin" or user_input == "root"
# Đã True ở vế trái, không cần kiểm tra vế phải
```

---

## 3. Input và Output

### 3.1. Output với `print()`

#### Đặt vấn đề

Làm sao để hiển thị kết quả cho người dùng? Làm sao để debug chương trình bằng cách in ra các giá trị?

#### Sử dụng `print()`

#### Cơ bản:

```
print("Hello, World!")
print(123)
print(3.14)
print(True)
```

## In nhiều giá trị:

```
name = "Nguyễn Minh Đạo"
age = 25
print("Tên:", name, "Tuổi:", age)
# Output: Tên: Nguyễn Minh Đạo Tuổi: 25
```

## Tham số của `print()`

### 1. `sep` - Ký tự phân cách (mặc định là khoảng trắng):

```
print("apple", "banana", "orange")
# Output: apple banana orange

print("apple", "banana", "orange", sep=", ")
# Output: apple, banana, orange

print("2025", "01", "11", sep="-")
# Output: 2025-01-11
```

### 2. `end` - Ký tự kết thúc (mặc định là `\n` - xuống dòng):

```
print("Hello")
print("World")
# Output:
# Hello
# World

print("Hello", end=" ")
print("World")
# Output: Hello World
```

### 3. In ra file:

```
with open("output.txt", "w", encoding="utf-8") as f:
    print("Dữ liệu ghi vào file", file=f)
```

## Các kỹ thuật format nâng cao

```
name = "Minh Đạo"
score = 8.756

# f-string với alignment
```

```
print(f"|{name:<10}|") # Căn trái: |Minh Đạo |
print(f"|{name:>10}|") # Căn phải: | Minh Đạo|
print(f"|{name:^10}|") # Căn giữa: | Minh Đạo |

# Format số
print(f"Điểm: {score:.2f}") # 8.76 (2 chữ số)
print(f"Điểm: {score:6.2f}") # " 8.76" (tổng 6 ký tự)
```

### 3.2. Input với `input()`


#### Đặt vấn đề

Làm sao để chương trình có thể nhận dữ liệu từ người dùng? Ví dụ: nhập tên, tuổi, điểm số?

#### Sử dụng `input()`

#### Cú pháp:

```
variable = input("Câu hỏi/thông báo: ")
```

 **Quan trọng:** `input()` LUÔN trả về **string** (chuỗi), dù người dùng nhập số.

#### Ví dụ cơ bản:

```
name = input("Nhập tên của bạn: ")
print(f"Xin chào, {name}!")

# Output:
# Nhập tên của bạn: Nguyễn Minh Đạo
# Xin chào, Nguyễn Minh Đạo!
```

#### Nhập input là số

```
# ✗ SAI - input() trả về string
age = input("Nhập tuổi: ")
next_year_age = age + 1 # TypeError: không cộng được string với int

# ✓ ĐÚNG - chuyển đổi sang int
age = int(input("Nhập tuổi: "))
next_year_age = age + 1
print(f"Năm sau bạn {next_year_age} tuổi")
```

#### Nhập số thực:

```
height = float(input("Nhập chiều cao (m): "))
print(f"Chiều cao của bạn: {height:.2f}m")
```

## Xử lý lỗi input

```
# Người dùng nhập sai định dạng
try:
    age = int(input("Nhập tuổi: "))
    print(f"Bạn {age} tuổi")
except ValueError:
    print("Lỗi: Vui lòng nhập một số nguyên hợp lệ")
```

## Multiple inputs trên cùng một dòng

```
# Nhập: 10 20 30
numbers = input("Nhập 3 số cách nhau bởi khoảng trắng: ")
a, b, c = numbers.split()
a, b, c = int(a), int(b), int(c)

# Hoặc ngắn gọn hơn:
a, b, c = map(int, input("Nhập 3 số: ").split())
print(f"Tổng: {a + b + c}")
```

---

## 4. Bài Tập Thực Hành

### Bài 1: Tính diện tích hình chữ nhật

```
"""
Input: Chiều dài và chiều rộng
Output: Diện tích và chu vi
"""

length = float(input("Nhập chiều dài: "))
width = float(input("Nhập chiều rộng: "))

area = length * width
perimeter = 2 * (length + width)

print(f"Diện tích: {area:.2f}")
print(f"Chu vi: {perimeter:.2f}")
```

---

### Bài 2: Chuyển đổi nhiệt độ



```
"""
Chuyển đổi từ Celsius sang Fahrenheit
Công thức:  $F = C \times \frac{9}{5} + 32$ 
"""

celsius = float(input("Nhập nhiệt độ (°C): "))
fahrenheit = celsius * 9/5 + 32

print(f"{celsius}°C = {fahrenheit:.2f}°F")
```

---

### Bài 3: Tính tuổi

```
"""
Input: Năm sinh
Output: Tuổi hiện tại
"""

birth_year = int(input("Nhập năm sinh: "))
current_year = 2025

age = current_year - birth_year
print(f"Bạn {age} tuổi")
```

---

### Bài 4: Kiểm tra số chẵn lẻ

```
"""
Input: Một số nguyên
Output: Thông báo số chẵn hay lẻ
"""

number = int(input("Nhập một số: "))

if number % 2 == 0:
    print(f"{number} là số chẵn")
else:
    print(f"{number} là số lẻ")
```

---

### Bài 5: Tính điểm trung bình

```
"""
Input: 3 điểm môn học
Output: Điểm trung bình và xếp loại
```

```
"""

math = float(input("Điểm Toán: "))
physics = float(input("Điểm Lý: "))
chemistry = float(input("Điểm Hóa: "))

average = (math + physics + chemistry) / 3

print(f"Điểm trung bình: {average:.2f}")

if average >= 8:
    print("Xếp loại: Giỏi")
elif average >= 6.5:
    print("Xếp loại: Khá")
elif average >= 5:
    print("Xếp loại: Trung bình")
else:
    print("Xếp loại: Yếu")
```

---

## Bài 6: Tính tiền mua hàng (có thuế)

```
"""
Input: Giá gốc, thuế VAT (%)
Output: Giá cuối cùng
"""

price = float(input("Nhập giá gốc: "))
tax_rate = float(input("Nhập thuế VAT (%): "))

tax_amount = price * tax_rate / 100
final_price = price + tax_amount

print(f"Tiền thuế: {tax_amount:.2f} VND")
print(f"Tổng tiền: {final_price:.2f} VND")
```

---

## Bài 7: Kiểm tra tam giác

```
"""
Input: 3 cạnh a, b, c
Output: Có phải tam giác hợp lệ không?
"""

a = float(input("Cạnh a: "))
b = float(input("Cạnh b: "))
c = float(input("Cạnh c: "))

# Điều kiện: tổng 2 cạnh > cạnh còn lại
```

```
is_valid = (a + b > c) and (b + c > a) and (a + c > b)

if is_valid:
    print("Đây là tam giác hợp lệ")
else:
    print("Đây KHÔNG phải tam giác hợp lệ")
```

---

## Bài 8: Tính BMI

```
"""
Input: Cân nặng (kg), chiều cao (m)
Output: BMI và phân loại
"""

weight = float(input("Cân nặng (kg): "))
height = float(input("Chiều cao (m): "))

bmi = weight / (height ** 2)

print(f"BMI của bạn: {bmi:.2f}")

if bmi < 18.5:
    print("Gầy")
elif bmi < 25:
    print("Bình thường")
elif bmi < 30:
    print("Thừa cân")
else:
    print("Béo phì")
```

---

## 5. Tổng Kết và Checklist

☒ Kiến thức cần nắm vững

### Biến và Kiểu dữ liệu:

- ☐ Tạo và gán giá trị cho biến
- ☐ Phân biệt 4 kiểu cơ bản: int, float, str, bool
- ☐ Chuyển đổi giữa các kiểu dữ liệu
- ☐ Sử dụng `type()` và `isinstance()`

### Toán tử:

- ☐ 7 toán tử số học: +, -, \*, /, //, %, \*\*
- ☐ Toán tử gán và gán kết hợp: =, +=, -=, \*=, ...
- ☐ 6 toán tử so sánh: ==, !=, >, <, >=, <=
- ☐ 3 toán tử logic: and, or, not

- ☐ Thứ tự ưu tiên toán tử

### Input/Output:

- ☐ Sử dụng `print()` với các tham số `sep`, `end`
- ☐ Format chuỗi với f-string
- ☐ Nhận input với `input()` và chuyển đổi kiểu

### 🔗 Lưu ý quan trọng

1. `input()` luôn trả về string - cần chuyển đổi khi làm việc với số
2. Phân biệt `=` (gán) và `==` (so sánh)
3. Division / luôn trả về float, dùng `//` cho phép chia nguyên
4. Boolean falsy values: `0`, `0.0`, `""`, `None`, `False`
5. f-string là cách format chuỗi hiện đại nhất (Python 3.6+)

### 📖 Bước tiếp theo

Sau khi nắm vững phần này, bạn sẵn sàng học:

- Cấu trúc điều khiển (`if/elif/else`)
- Vòng lặp (`for`, `while`)
- Functions (hàm)
- Cấu trúc dữ liệu (list, dict, set, tuple)

---

## 6. Câu Hỏi Thường Gặp (FAQ)

### Q1: Tại sao `0.1 + 0.2` không bằng `0.3`?

**A:** Do cách máy tính lưu trữ số thực theo chuẩn IEEE 754. Một số thập phân không thể biểu diễn chính xác dưới dạng nhị phân.

```
print(0.1 + 0.2)          # 0.30000000000000004
print(0.1 + 0.2 == 0.3)  # False

# Giải pháp: so sánh với epsilon
abs(0.1 + 0.2 - 0.3) < 1e-10 # True
```

---

### Q2: Sự khác biệt giữa `/` và `//`?

**A:**

- `/`: Phép chia thông thường, luôn trả về float
- `//`: Chia lấy phần nguyên (floor division)

```
print(7 / 2)    # 3.5
print(7 // 2)   # 3
```

```
print(-7 / 2) # -3.5
print(-7 // 2) # -4 (làm tròn xuống)
```

---

### Q3: Khi nào dùng ' và khi nào dùng "?

**A:** Cả hai đều giống nhau, dùng cái nào cũng được. Nhưng:

- Nếu chuỗi có dấu nháy đơn bên trong: dùng "
- Nếu chuỗi có dấu nháy kép bên trong: dùng '

```
text1 = "It's a beautiful day" # Dễ đọc
text2 = 'He said "Hello"'     # Dễ đọc
text3 = "He said \"Hello\""   # Phải escape
```

---

### Q4: Tại sao print(type(5/2)) trả về <class 'float'>?

**A:** Vì phép chia / trong Python 3 LUÔN trả về float, kể cả khi chia hết:

```
print(10 / 2)      # 5.0 (float, không phải 5)
print(10 // 2)     # 5 (int)
```

---

### Q5: and và or có trả về True/False không?

**A:** Không phải lúc nào cũng vậy! Chúng trả về một trong hai toán hạng:

```
# and: trả về toán hạng đầu tiên falsy, hoặc toán hạng cuối
print(5 and 10)      # 10
print(0 and 10)      # 0
print(5 and 0)       # 0

# or: trả về toán hạng đầu tiên truthy, hoặc toán hạng cuối
print(5 or 10)       # 5
print(0 or 10)       # 10
print(0 or "")       # "" (empty string)
```

Khi dùng trong điều kiện, Python tự chuyển sang boolean.

---

## 7. Thử Thách Cuối Khóa

### Challenge 1: Máy tính đơn giản

Viết chương trình nhận 2 số và một phép toán (+, -, \*, /) rồi in kết quả.

```
"""
Input:
    Số thứ nhất: 10
    Phép toán: +
    Số thứ hai: 5
Output:
    10 + 5 = 15
"""
```

Challenge 2: Tính tiền tip (hóa đơn nhà hàng)

```
"""
Input:
    - Tổng hóa đơn
    - % tip (15%, 18%, 20%)
Output:
    - Tiền tip
    - Tổng tiền phải trả
    - Số tiền mỗi người (nếu chia đều)
"""
```

Challenge 3: Đổi đơn vị thời gian

```
"""
Input: Số giây
Output: X ngày, Y giờ, Z phút, T giây

Ví dụ:
Input: 100000
Output: 1 ngày, 3 giờ, 46 phút, 40 giây
"""
```

---