

# Chapter 3: Conditional Execution ( Cấu trúc rẽ nhánh )

## 1) Conditional steps

**Khái niệm**

- Một chương trình có thể thực hiện các bước khác nhau tùy theo điều kiện.
- Mỗi điều kiện được kiểm tra bằng biểu thức so sánh (Boolean Expression).
- Kết quả là True (Đúng) hoặc False (Sai).
- Dựa vào đó, chương trình sẽ chọn nhánh lệnh phù hợp để chạy.

```

x = 5
if x < 10:
    print('Smaller')
if x > 20:
    print('Bigger')
print('Finis')

```

**Output:** Smaller Finis

## 2) Comparison Operators

**Khái niệm**

- Biểu thức Boolean (Boolean expression) đặt ra một câu hỏi logic và trả về:
  - True (Đúng) hoặc
  - False (Sai)
- Kết quả này được dùng để điều khiển luồng chương trình (ví dụ trong các lệnh if).
- Các toán tử so sánh chỉ kiểm tra giá trị, không thay đổi biến.

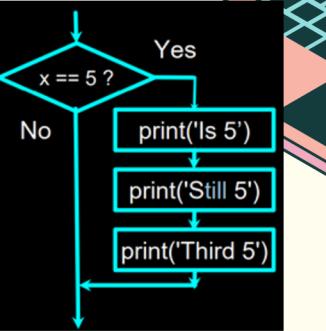
Python	Meaning
<	Less than
<=	Less than or Equal to
==	Equal to
>=	Greater than or Equal to
>	Greater than
!=	Not equal

## 3) One - Way Decisions

```

x = 5
print('Before 5')
if x == 5 :
    print('Is 5')
    print('Is Still 5')
    print('Third 5')
print('Afterwards 5')

```



## 4) Indentation

- Khái niệm**
- Python dùng thụt lề (indentation) để xác định khối lệnh (block of code).
  - Không giống các ngôn ngữ khác (như C, Java, C++), Python không dùng dấu ngoặc {}.
  - Mức thụt vào sau dấu ":" cho biết dòng nào thuộc về khối lệnh của if, for, while, v.v.

### Quy tắc cơ bản

- Sau lệnh if, for, while, def, class... phải có dấu :
- Sau đó thụt vào một cấp (thường là 4 khoảng trắng).
- Các dòng cùng mức thụt lề = cùng một khối lệnh.
- Giảm thụt lề trở lại → kết thúc khối lệnh.
- Dòng trống hoặc comment không ảnh hưởng đến thụt lề.
- Không pha trộn TAB và SPACE, rất dễ gây lỗi.

## 6) Two - Way Decisions

- Khái niệm**
- Dùng khi bạn muốn chương trình chọn giữa hai hướng xử lý:
    - Nếu điều kiện đúng (True) → thực hiện khối lệnh thứ nhất.
    - Nếu điều kiện sai (False) → thực hiện khối lệnh thứ hai.
  - Cấu trúc này dùng if kèm theo else

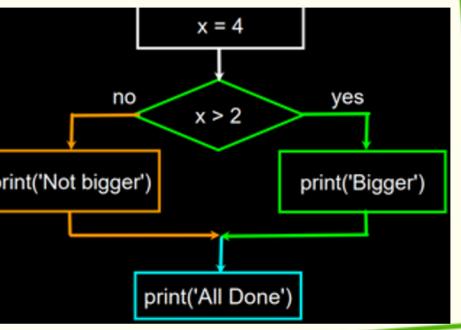
```

x = 4

if x > 2 :
    print('Bigger')
else :
    print('Smaller')

print('All done')

```



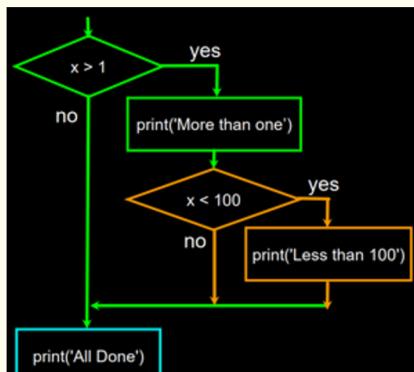
## 5) Nested Decisions

- Khái niệm**
- "Nested" nghĩa là lồng vào bên trong.
  - Trong Python, bạn có thể đặt một câu lệnh if bên trong một câu lệnh if khác.
  - Dạng này dùng khi một điều kiện phụ thuộc vào điều kiện khác.
  - Mỗi cấp lồng nhau cần thụt lề thêm một cấp (4 dấu cách).

```

x = 42
if x > 1 :
    print('More than one')
    if x < 100 :
        print('Less than 100')
    print('All done')

```



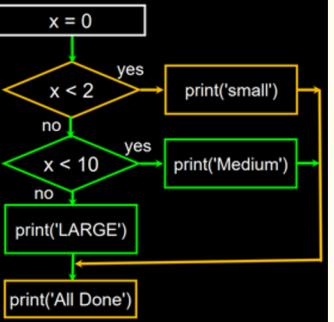
## 7) Multi - Way

- Khái niệm**
- Khi có nhiều điều kiện khác nhau cần kiểm tra, bạn dùng if ... elif ... else.
  - Python sẽ kiểm tra lần lượt từ trên xuống,
    - gặp điều kiện đúng đầu tiên → thực hiện khối đó,
    - rồi bỏ qua phần còn lại.

```

x = 0
if x < 2 :
    print('small')
elif x < 10 :
    print('Medium')
else :
    print('LARGE')
print('All done')

```



## 8) Try / Except

- Khái niệm**
- Dùng để xử lý lỗi khi chạy chương trình, tránh việc chương trình bị dừng đột ngột.
  - Python thử (try) chạy một đoạn mã.
    - Nếu chạy thành công, bỏ qua phần except.
    - Nếu lỗi xảy ra, Python nhảy sang phần except để xử lý.
  - Giúp chương trình an toàn hơn, đặc biệt khi xử lý dữ liệu đầu vào từ người dùng.

```

astr = 'Bob'
try:
    print('Hello')
    istr = int(astr)
    print('There')
except:
    istr = -1
print('Done', istr)

```



## 8) Try / Except

