



# CÁC CẤU TRÚC ĐIỀU KHIỂN

- Python hỗ trợ một số cấu trúc điều khiển thông dụng.
- Hầu hết các cấu trúc điều khiển đều dựa vào thụt đầu dòng (4 spaces) để tạo thành một block xử lý, thay vì sử dụng { ... } như các ngôn ngữ khác.
- Câu lệnh if được sử dụng để kiểm tra một điều kiện: nếu điều kiện là đúng sẽ chạy một khối các câu lệnh (được gọi là if-block).

```
if condition:  
    statements
```

- Khối else sẽ được thực thi nếu điều kiện trong if là sai

```
if 1 > 2:  
    print("1 is greater than 2")  
else:  
    print("1 is not greater than 2")
```

- Khi ta phải kiểm tra thêm một điều kiện nữa nếu trong if là sai ta có thể sử dụng elif

```
numberInput = int(input('Nhập vào 1 số: '))  
if numberInput == 1:  
    print("numberInput is one")  
elif numberInput == 2:  
    print("numberInput is two")  
else:  
    print("numberInput is not one or two")
```

- Cũng có thể đặt các khối lệnh if lồng nhau

```
if numberInput == 1:  
    print("numberInput is one")  
else:  
    if numberInput == 2:  
        print("numberInput is two")  
    else:  
        print("numberInput is not one or two")
```

- Vòng lặp for ở trong Python có tác dụng lặp các biến dữ liệu có kiểu dữ liệu list, tuple hoặc string,... Sử dụng cú pháp như sau:

```
for variable in data:  
    #code
```

- Câu lệnh for trong Python khác một chút so với các ngôn ngữ khác, thay vì lặp theo các bước, câu lệnh for của Python lặp qua các phần tử của 1 danh sách truyền vào, tương tự với foreach - lặp tất cả các phần tử.

```
words = ['cat', 'dog', 'bird'];  
for w in words:  
    print(w)
```

cat  
dog  
bird

```
for c in 'Hello':  
    print(c)
```

H  
e  
l  
l  
o

- Hàm `range()`: giúp tạo ra một list một cách nhanh chóng. Với cú pháp:

```
range(elementNumbers)
```

- Cú pháp này sẽ tạo ra 1 list bắt đầu từ 0 liên tục và kết thúc khi có đủ `elementNumbers`:

```
for number in range(5):  
    print(number)
```

0  
1  
2  
3  
4

- Cú pháp đầy đủ: range(start, stop, step)

```
for number in range(1,10,2):  
    print(number)
```

1  
3  
5  
7  
9



- Vòng lặp while tương tự trong các ngôn ngữ khác, lặp khi thỏa mãn điều kiện cho trước. Cú pháp:


```
while <condition>:  
    # code
```

- break và continue: break cho phép thoát khỏi vòng lặp, còn continue cho phép bỏ qua lượt chạy hiện tại của vòng lặp và chạy tiếp.

```
i = 0
while (i <= 10):
    print(i)
    if i == 5:
        break
    i += 1

for i in 'i am a robot':
    if i == ' ':
        continue
    print(i, end='')
```

---

 0  
1  
2  
3  
4  
5  
iamarobot

---

## **Bài tập 1:** Tìm số lớn nhất

Viết chương trình yêu cầu người dùng nhập ba số nguyên và sau đó hiển thị số lớn nhất.

## **Bài tập 2:** Tính giai thừa

Viết chương trình yêu cầu người dùng nhập một số nguyên dương và sau đó tính giai thừa của số đó. Gợi ý: Sử dụng vòng lặp for.

## **Bài tập 3:** Kiểm tra số nguyên tố

Viết chương trình yêu cầu người dùng nhập một số nguyên dương và sau đó kiểm tra xem số đó có phải là số nguyên tố hay không.

## **Bài tập 4:** In hình tam giác

## **Bài tập 5:** Đảo ngược chuỗi

Viết chương trình yêu cầu người dùng nhập một chuỗi và sau đó in ra chuỗi đó theo thứ tự ngược lại. Ví dụ, nếu người dùng nhập "Python", chương trình sẽ in ra "nohtyP".

## **Bài tập 6:** Đoán số

Viết chương trình tạo một số ngẫu nhiên từ 1 đến 100 và yêu cầu người dùng đoán số đó. Chương trình sẽ đưa ra gợi ý liệu số đoán là lớn hơn hoặc nhỏ hơn số cần đoán, cho đến khi người dùng đoán đúng số.

## **Bài tập 7:** Tính tổng dãy số

Viết chương trình yêu cầu người dùng nhập một số nguyên dương  $n$ , sau đó tính tổng của dãy số từ 1 đến  $n$ . Gợi ý: Sử dụng vòng lặp for.

## **Bài tập 8:** Sắp xếp danh sách

Viết chương trình yêu cầu người dùng nhập một danh sách các số nguyên, sau đó sắp xếp danh sách theo thứ tự tăng dần và in danh sách đã sắp xếp.

## **Bài tập 9:** Đọc số thành chữ

Viết chương trình yêu cầu người dùng nhập một số nguyên từ 1 đến 999 và sau đó in số đó thành chữ. Ví dụ, nếu người dùng nhập 123, chương trình sẽ in ra "một trăm hai mươi ba".

## Bài tập 4: In hình tam giác

```
n = int(input("Nhập số hàng cho tam giác vuông: "))

# Dùng vòng lặp for để in tam giác

for i in range(n):

    for j in range(i + 1):

        print("*", end=" ")

    print() # In xuống dòng để chuyển sang hàng mới
```

Nhập số hàng cho tam giác vuông: 6

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

## Bài tập 9: Đọc số thành chữ

```
def number_to_words(number):  
    # Định nghĩa các từ số từ 1 đến 19  
    ones = ["", "một", "hai", "ba", "bốn", "năm", "sáu", "bảy", "tám", "chín",  
            "mười", "mười một", "mười hai", "mười ba", "mười bốn", "mười lăm",  
            "mười sáu", "mười bảy", "mười tám", "mười chín"]  
    # Định nghĩa các từ hàng trăm  
    hundreds = ["", "một trăm", "hai trăm", "ba trăm", "bốn trăm", "năm trăm",  
                "sáu trăm", "bảy trăm", "tám trăm", "chín trăm"]  
    if number == 0:  
        return "Không"  
    elif number < 20:  
        return ones[number]  
    elif number < 100:  
        tens_digit = number // 10  
        ones_digit = number % 10  
        return ones[tens_digit] + " mươi " + ones[ones_digit]  
    else:  
        hundreds_digit = number // 100  
        remainder = number % 100  
        return hundreds[hundreds_digit] + " " + number_to_words(remainder)
```

## Bài tập 9: Đọc số thành chữ

```
def main():  
    number = int(input("Nhập một số từ 1 đến 999: "))  
    if 1 <= number <= 999:  
        words = number_to_words(number)  
        print(f"{number} được viết thành chữ là: {words}")  
    else:  
        print("Số bạn nhập không nằm trong khoảng từ 1 đến 999.")  
  
if __name__ == "__main__":  
    main()
```

Nhập một số từ 1 đến 999: 234

234 được viết thành chữ là: hai trăm ba mươi bốn