**BÁO CÁO TIẾN ĐỘ HỌC TẬP (TUẦN 1)**

**Nguyễn Nguyên Hảo**

1. **Câu lệnh print và các kiểu dữ liệu trong Python**
2. **Câu lệnh Print**

*+ Câu lệnh print giúp in ra nội dung lên màn hình, nội dung có thể là xâu kí tự, các đối tượng bất kì trong Python. Trước khi được in ra các đối tượng này phải được chuyển thành xâu kí tự.*

**+ Cú pháp:**

**Print( object, sep = seperator, end = end)**

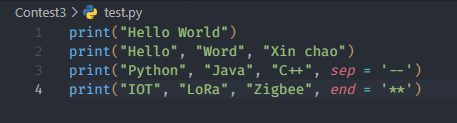
**Các tham số của hàm print:**

**Object:** Các đối tượng trong Python

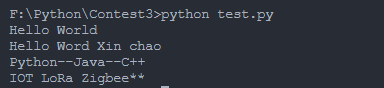
**Sep :** Phân các các đối tượng khi in, nếu không có tham số này thì mặc định sẽ là dấu cách

**End :** Chỉ ra kí tự được in ở cuối dòng, nếu không có tham số này mặc định sẽ là dấu xuống dòng

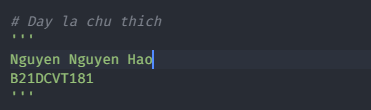
**Ví dụ:**



**Output:**



1. **Comment trong Python**



1. **Biến và Kiểu dữ liệu**

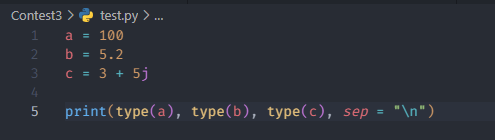
**+ Biến :**

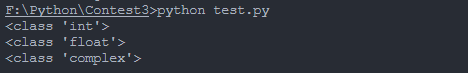
*Không được đặt tên biến có dấu cách, bắt đầu bằng số hoặc kí tự đặc biệt*

*Tên biến trong Python phân biệt hoa thường*

**+ Kiểu dữ liệu:**

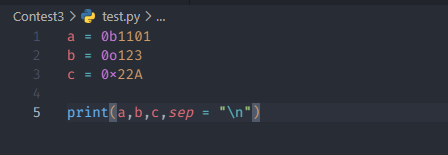
**Kiểu dữ liệu số :**

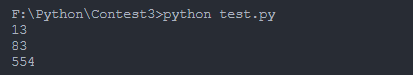




**Số nguyên:**

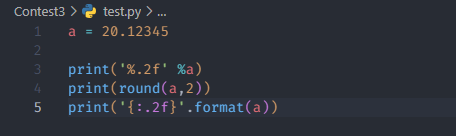
*Ngoài việc in ra các số ở hệ 10, ta có thể in ra các số ở hệ 2*, 8 16

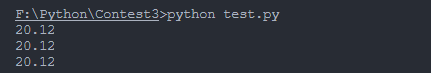




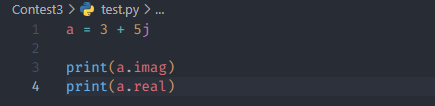
**Số thực:**

*In số thực với số lượng chữ số sau dấu phẩy*



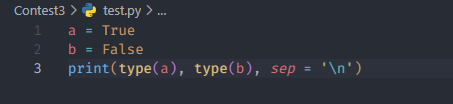


**Số phức:**



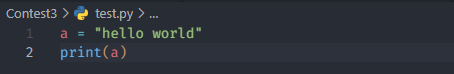


**Kiểu Boolen:**



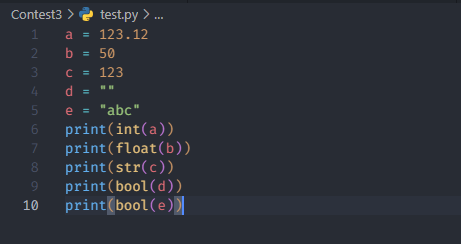


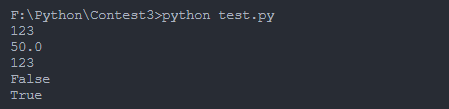
**Kiểu xâu:**



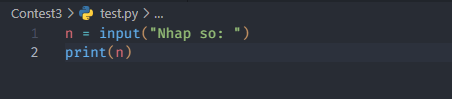


**Ép kiểu:**

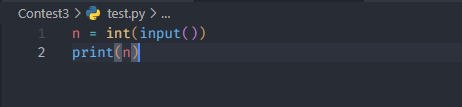




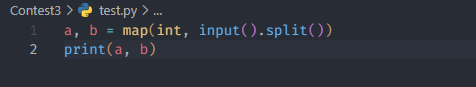
1. **Nhập dữ liệu với hàm input và nhập nhiều dữ liệu trên 1 dòng bằng hàm map**
2. **Nhập dữ liệu bằng input**





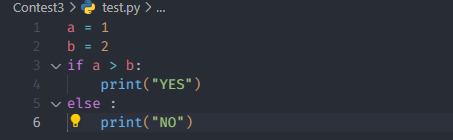


1. **Nhâp nhiều số bằng map**



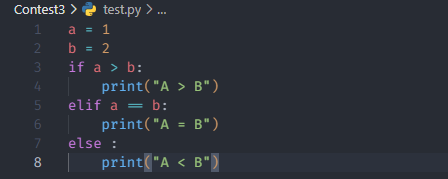


1. **Cấu trúc rẽ nhanh trong Python**
2. **Cấu trúc if else:**





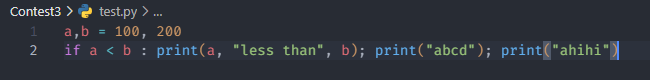
1. **Cấu trúc elif:**

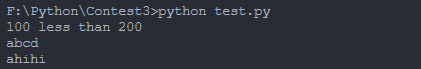




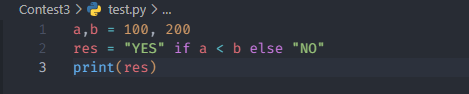
1. **Cấu trúc shorthand if và toán tử 3 ngôi**

*Công thức Shorthand if:*





*Toán tử 3 ngôi:*





1. **Vòng lặp**
2. **Vòng lặp for và hàm range():**

*Khác với các ngôn ngữ khác, thay vì duyệt qua các giá trị số thì python sẽ lặp trên các iterable (list, tupple, string, set,..) . Thứ tự duyệt sẽ là các phân tử trong interable. Để thực hiện vòng lặp for ta sử dụng built-in function là range()*

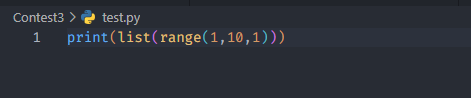
Hàm range sẽ sinh ra 1 dãy số và ta sẽ duyệt qua từng số trong dãy trên:

**Cú pháp: range (start, stop, step)**

**Start:** giá trị bắt đầu (mặc định là 0)

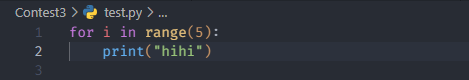
**Stop:** giá trị kết thúc (không lấy cận này)

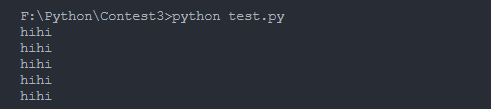
**Step:** Giá trị bước nhảy (mặc định là 1)





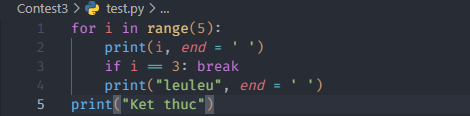
**Vòng for:**





1. **Câu lệnh break và continue**

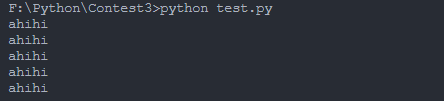
*Ví dụ 1:*



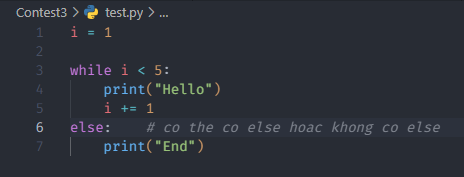


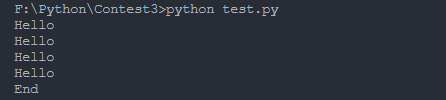
*Ví dụ 2:*



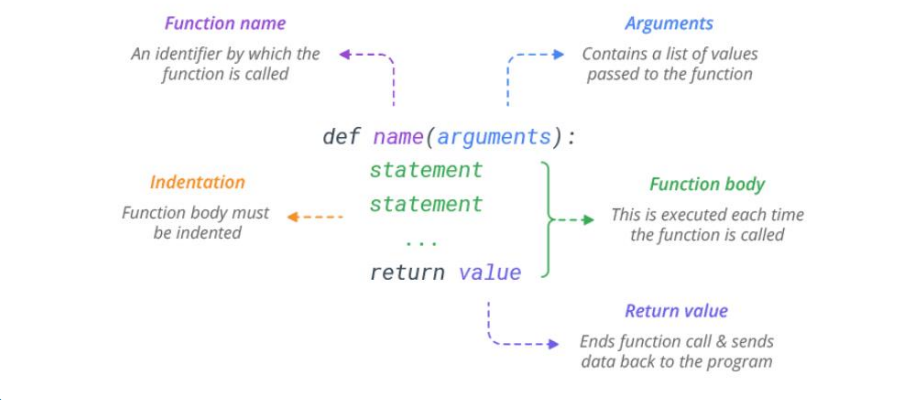


1. **Vòng lặp While, break, continue**

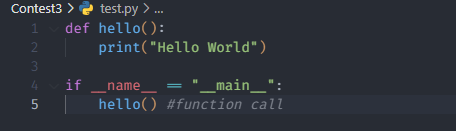




1. **Hàm trong Python**
2. **Xây dựng hàm**



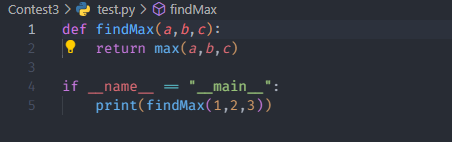
*Lời gọi hàm:*



*Tham số và đối số:*

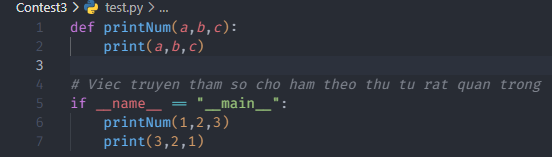


*Câu lệnh return:*



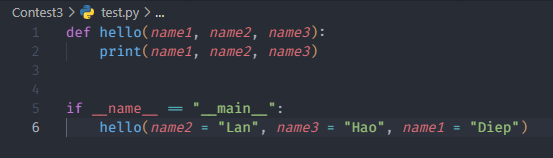


*Việc truyền tham số cho hàm theo thứ tự*



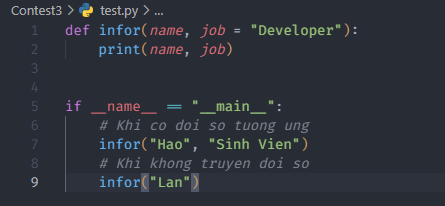


*Tuy nhiên khi ta sử dụng cú pháp parameter\_name = value thì thứ tự không còn quan trọng nữa*





*Ta có thể chỉ định đối số mặc định khi xây dựng 1 hàm. Giá trị mặc định này được sử dụng nếu hàm được gọi mà không có đối số tương ứng được truyền vào.*





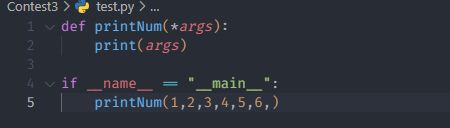
1. **Variable length argument (\*args and \*\*kwargs )**

*Variable length argument được sử dụng khi bạn không biết trước được số lượng phần tử của tham số*

*\*args: Khi tham số có thêm dấu \* ở trước, hàm sẽ tiến hành thu thập các đối số thành 1 tupple, sau đó ta có thể sử dụng args như 1 tupple*

*\*\*kwargs: Các đối số được thu thập thành 1 dictionary, trong đó đối số là key và giá trị tương ứng được gán cho value*

*Ví dụ 1:*





*Ví dụ 2:*





BÁO CÁO TIẾN DỘ HỌC TẬP (TUẦN 2)

Nguyễn Nguyên Hảo

1. ĐỘ PHỨC TẠP CỦA THUẬT TOÁN TRONG PYTHON (BIG O NOTATION)

Độ phức tạp về thời gian (time complexity) và độ phức tạp về không gian (space complexity) là hai yếu tốt quyết định một thuật toán có phù hợp để giải bài toán hay không

1. PHẠM VI CỦA BIẾN TRONG PYTHON

Không phải mọi biến đều có thể truy cập ở bất cứ đâu trong chương trình, phạm vi của biến mà tại đó có thể truy cập được gọi là scope, được xác định tại nơi mà biến khai báo

Python có 3 phạm vi:

+ Local scope

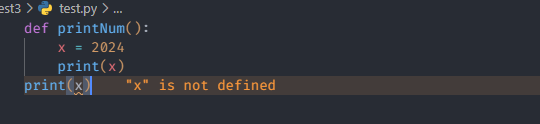
+ Global scope

+ Enclosing scope

1. Local Scope

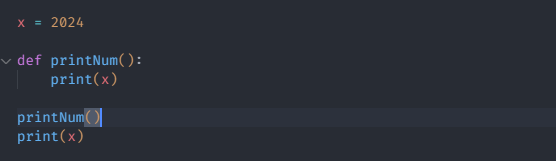
Một biến được khai báo trong phạm vi của 1 hàm được có phạm vi local scope. Khi đó biến có thể được truy cập từ vị trí nó được khai báo cho đến hết phạm vi của hàm, khi nào hàm còn thực thi thì biến đó còn tồn tại

+ Việc truy cập biến ngoài phạm vi của hàm sẽ gây ra lỗi:



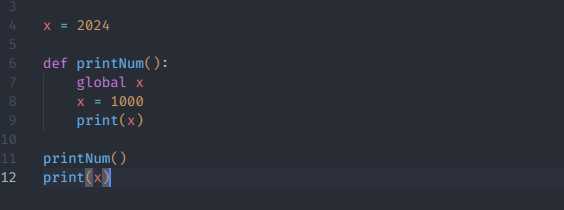
1. Global Scope

Biến được khai báo bên ngoài các hàm và có phạm vi từ đầu cho đến cuối file source code





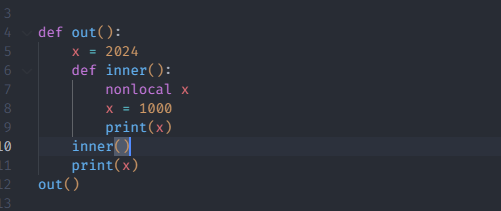
+ Để thay đổi giá trị của biến toàn cục trong hàm con ta sử dụng từ khóa global





1. Enclosing Scope

Trong nested function, khi khai báo hai biến có cùng tên ở hai hàm lồng nhau thì 2 biến này có phạm vi khác nhau



1. LIST TRONG PYTHON

List tương tự như cấu trúc mảng trong các ngôn ngũ khác nhưng có phần linh hoạt hơn

* Các tính chất của list:

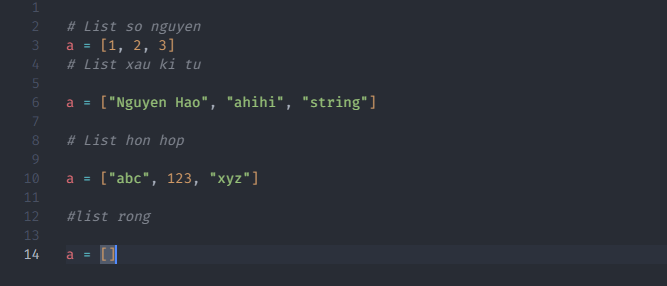
+ List are ordered: Các phần tử trong list có thứ tự

+ Accessed by index: Truy cập các phần tử trong list qua chỉ số

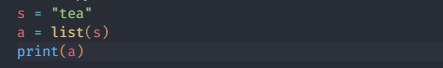
+ List can contain any sort of object: List có thể chứa tập hợp các kiểu dữ liệu như str, int, float thậm chí là 1 list khác

+ Lists are changeable: các phần tử trong list có thể thay đổi

1. Tạo list

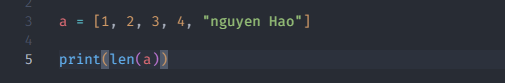


1. List constructor



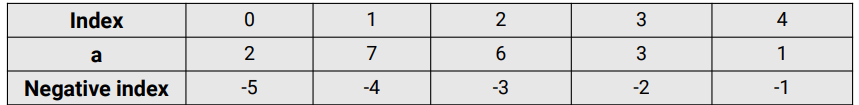


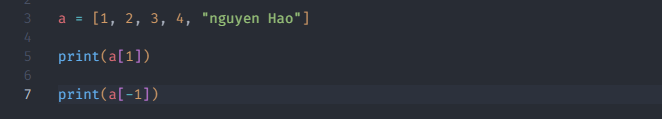
1. Hàm len()

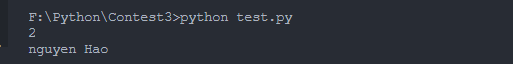




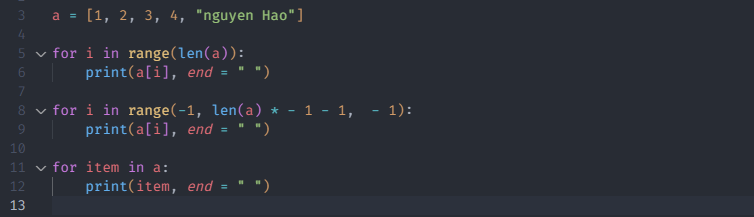
1. Truy cập các phần tử của list thông qua chỉ số





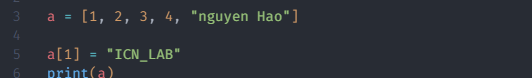


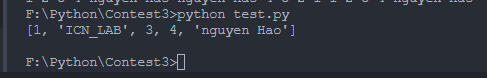
1. Duyêt list



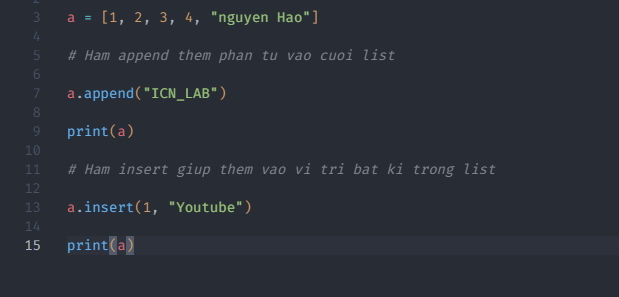


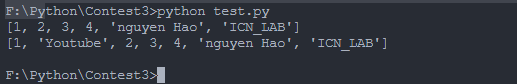
1. Thay đổi giá trị phần tử trong list



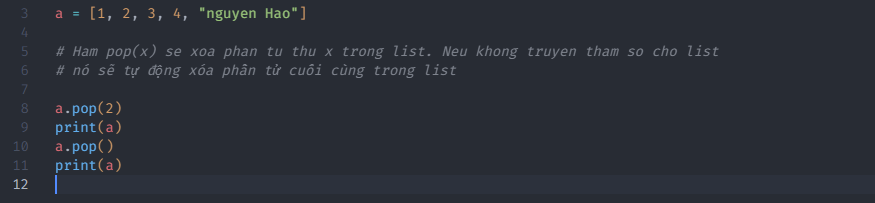


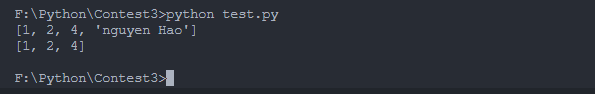
1. Thêm 1 phần tử vào list





1. Xóa 1 phần tử khỏi list





* Ngoài ra ta có thể sử dụng hàm del

