# Phân biệt toán tử định dạng chuỗi và hàm định dạng chuỗi có sẵn trong gói thư viện chuẩn Python? Cho năm ví dụ minh họa tương ứng?

Toán tử định dạng chuỗi: Toán tử định dạng chuỗi (F-string) là một cách tiện lợi để tạo chuỗi với các giá trị được thay thế.

Định dạng f-string khá đơn giản khi sử dụng, tất cả những gì cần là một chữ f đứng trước một chuỗi thì tất cả các biểu thức (nằm trong cặp đóng mở ngoặc nhọn) sẽ được thực thi và chuyển đổi thành chuỗi và thay thế vào biểu thức đó.

Cú pháp: f"Chuỗi {biến}".

### VD1:

age = 35

print(f"Tuổi của bạn là: {age}")

* kết quả in ra: Tuổi của bạn là: 35

### VD2:

don\_gia = 25000

so\_luong = 3

print( f"Bạn mua {so\_luong} dia com het tong cong: {don\_gia \* so\_luong} VND")

🡺Kết quả in ra: Ban mua 3 dia com het tong cong: 75000 VND

Hàm định dạng chuỗi: Hàm định dạng chuỗi cho phép bạn tạo chuỗi với các giá trị thay thế dựa trên vị trí hoặc tên biến. Bản thân biến dạng chuỗi là một đối tượng str có một hàm nội tại là format() dùng để định dạng nội dung.

### VD3:

age = 35

cau\_noi = " Tuoi cua ban la: {}"

cau\_noi\_1 = cau\_noi.format(age)

print(cau\_noi\_1)

🡺kết quả in ra: Tuổi của bạn là: 35

### VD4:

name = "Alice"

age = 30

formatted\_string = "Tên tôi là {} và tôi {} tuổi.".format(name, age)

print(formatted\_string)

🡺kết quả in ra: Tên tôi là Alice và tôi 30 tuổi.

### VD5:

don\_gia = 25000

so\_luong = 3

tong\_tien = don\_gia\*so\_luong

mau\_thong\_bao = "Bạn mua {so\_luong} dia com het tong cong: {tong\_tien} VND"

thong\_bao = mau\_thong\_bao.format(so\_luong = so\_luong, tong\_tien = tong\_tien)

print(thong\_bao)

🡺Kết quả in ra: Bạn mua 3 dia com het tong cong: 75000 VND

# Viết chương trình xuất ra số ngẫu nhiên trong một đoạn bất kỳ bắt cho trước?

import random

random = random.randrange(0, 999, 1)

print(random)

# Khác biệt cơ bản giữa list và tuple?

Sự khác biệt cơ bản giữa list và tuple trong Python bao gồm:

## Tính biến đổi:

* List: Là mutable (có thể thay đổi), nghĩa là bạn có thể thay đổi, thêm, hoặc xóa các phần tử sau khi list đã được tạo.
* Tuple: Là immutable (không thể thay đổi), nghĩa là một khi tuple đã được tạo, bạn không thể thay đổi các phần tử bên trong nó.

## Hiệu suất (Performance):

* List: Do tính chất mutable, list có thể linh hoạt hơn, nhưng thường sẽ tốn nhiều bộ nhớ hơn và xử lý chậm hơn so với tuple.
* Tuple: Vì immutable, tuple thường nhanh hơn và sử dụng ít bộ nhớ hơn, đặc biệt là trong các trường hợp cần lưu trữ dữ liệu cố định mà không thay đổi.

## Cú pháp (Syntax):

* List: Được tạo bằng cách đặt các phần tử bên trong cặp dấu ngoặc vuông [].
* Tuple: Được tạo bằng cách đặt các phần tử bên trong cặp dấu ngoặc tròn ().

## Các thao tác:

* List: Có thể sử dụng các thao tác như thêm (`append`), xóa (`remove`, `pop`), hoặc thay đổi giá trị của phần tử.

1. Tuple: Không hỗ trợ các thao tác thay đổi, nhưng bạn có thể thực hiện các thao tác như lấy giá trị (indexing), duyệt qua các phần tử (looping), hoặc đếm số lượng phần tử.

## Tính ứng dụng:

* List: Thích hợp khi bạn cần một tập hợp dữ liệu có thể thay đổi trong suốt quá trình xử lý.
* Tuple: Thích hợp cho các trường hợp cần một tập hợp dữ liệu cố định, ví dụ như các cặp giá trị, các tham số hàm không đổi, hoặc làm khóa trong dictionary.

# Ứng dụng kiểu dữ liệu tuple trong thực tế?

* Ứng dụng trong một chương trình quản lý tọa độ địa lý, một điểm có thể được biểu diễn bằng một tuple chứa kinh độ và vĩ độ.
* Một ví dụ thực tế khác về việc sử dụng tuple là trong việc lưu trữ thông tin của một sinh viên trong một hệ thống quản lý sinh viên.

Giả sử hệ thống cần lưu trữ thông tin gồm tên sinh viên, mã số sinh viên, và năm sinh. Thông tin này có thể được lưu trữ dưới dạng một tuple, vì nó không cần thay đổi sau khi được tạo ra.

Sử dụng tuple giúp đảm bảo rằng thông tin này không bị thay đổi trong suốt quá trình xử lý. Khi cần truy xuất các giá trị, bạn có thể dễ dàng lấy chúng ra từ tuple.