Câu 1: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?

Tiêu chí	Toán tử Định dạng Chuỗi	Hàm Định dạng Chuỗi
Mô tả	Phương pháp định dạng chuỗi cũ, kế thừa từ ngôn ngữ C.	Phương pháp định dạng chuỗi hiện đại, được giới thiệu từ Python 2.7.
Cú pháp cơ bản	"chuỗi định dạng" % giá trị	"chuỗi định dạng {}".format(gía trị)
Tính linh hoạt	Hạn chế, chỉ hỗ trợ các mã định dạng cơ bản	Hỗ trợ nhiều tùy chọn định dạng hơn, có thể sử dụng chỉ số và tên tham số.
Sử dụng biểu thức	Không hỗ trợ biểu thức phức tạp.	Hỗ trợ biểu thức phức tạp trong định dạng.
Khả năng đọc mã	Đơn giản nhưng có thể khó đọc với nhiều giá trị hoặc định dạng phức tạp.	Dễ đọc và bảo trì hơn, đặc biệt với nhiều giá trị hoặc định dạng phức tạp.
Khả năng mở rộng	Khó mở rộng cho các định dạng phức tạp hoặc thay đổi định dạng.	Dễ mở rộng, dễ dàng thay đổi định dạng và thêm giá trị.
Khả năng tương thích	Tương thích với các phiên bản Python cũ.	Hỗ trợ tốt trong các phiên bản Python mới hơn, từ Python 2.7 trở đi.
Cú pháp đa giá trị	"chuỗi định dạng" % (giá trị1, giá trị2, ...)	"chuỗi định dạng {0}, {1}, ...".format(gía trị1, giá trị2, ...)

Vd1

```
name = "Alice"
print("Tên: {}".format(name))
```

vd2

```
age = 30
print("Tuổi: {}".format(age))
```

vd3

```
name = "Bob"
age = 25
print("Tên: {}, Tuổi: {}".format(name, age))
vd4
name = "Charlie"
height = 1.75
print("Tên: {0}, Chiều cao: {1:.1f} m".format(name, height))
vd5
pi = 3.14159
print("Giá trị của pi là: %.2f" % pi)  # Giá trị của pi là: 3.14
```

Câu 2: Viết chương trình xuất ra số ngẫu nhiên trong một đoạn bất kỳ bất cho trước?

Bài 12 file python

Câu 3: Khác biệt cơ bản giữa list và tuple trong Python

List và **tuple** là hai kiểu dữ liệu được sử dụng rất phổ biến trong Python để lưu trữ các phần tử. Tuy nhiên, chúng có những đặc điểm khác biệt quan trọng.

1. Tính bất biến (Immutability):
   List: Có thể thay đổi sau khi được tạo. Bạn có thể thêm, xóa hoặc sửa đổi các phần tử trong list.
   Tuple: Không thể thay đổi sau khi được tạo. Các phần tử của tuple được cố định và không thể được sửa đổi.
2. Sử dụng dấu ngoặc:
   List: Được bao quanh bởi dấu ngoặc vuông [].
   Tuple: Được bao quanh bởi dấu ngoặc tròn ().
3. Hiệu suất:
   Tuple: Thường nhanh hơn list một chút vì tính chất bất biến của nó. Khi Python biết rằng một đối tượng sẽ không thay đổi, nó có thể thực hiện một số tối ưu hóa.
   List: Có thể chậm hơn một chút so với tuple do tính năng có thể thay đổi của nó.
4. Sử dụng:
   List: Thường được sử dụng khi bạn cần một cấu trúc dữ liệu có thể thay đổi linh hoạt. Ví dụ: lưu trữ danh sách các sản phẩm, danh sách các người dùng.
   Tuple: Thường được sử dụng khi bạn cần một cấu trúc dữ liệu bất biến, chẳng hạn như:
   - Key trong dictionary: Key của dictionary phải là bất biến, và tuple thường được sử dụng cho mục đích này.

- Trả về nhiều giá trị từ một hàm: Hàm có thể trả về một tuple chứa nhiều giá trị.
- Định nghĩa các constant: Các giá trị không đổi trong chương trình thường được định nghĩa dưới dạng tuple.

Câu 4:Ứng dụng của kiểu dữ liệu tuple trong thực tế

Tuple trong Python là một cấu trúc dữ liệu bất biến, có thứ tự, thường được sử dụng để nhóm các giá trị có liên quan với nhau.

một số ví dụ điển hình về việc sử dụng tuple:

1. Đại diện cho các điểm tọa độ:
   - Mỗi điểm tọa độ trên mặt phẳng có thể được biểu diễn bằng một tuple gồm hai phần tử: tọa độ x và tọa độ y.
   - Ví dụ: point = (3, 5) đại diện cho điểm có tọa độ x là 3 và tọa độ y là 5.
2. Lưu trữ thông tin về người dùng:
   - Một tuple có thể chứa thông tin cơ bản về một người dùng như: tên, tuổi, email.
   - Ví dụ: user = ("Nguyen Van A", 25, "vanan@example.com")
3. Trả về nhiều giá trị từ một hàm:
   - Khi một hàm cần trả về nhiều giá trị, ta có thể đóng gói các giá trị đó vào một tuple và trả về.
   - Ví dụ:

Python
```
def find_min_max(numbers):
    return min(numbers), max(numbers)
```
4. Làm khóa trong dictionary:
   - Vì tuple là bất biến nên chúng có thể được sử dụng làm khóa trong dictionary. Điều này rất hữu ích khi bạn muốn sử dụng một cặp hoặc bộ nhiều giá trị làm khóa.
   - Ví dụ:

Python
```
my_dict = {(1, 2): "value1", (3, 4): "value2"}
```
5. Định nghĩa các constant:
   - Tuple thường được sử dụng để định nghĩa các hằng số (constant) trong chương trình.
   - Ví dụ:

Python
```
DAYS_IN_WEEK = (7,)
COLORS = ("red", "green", "blue")
```
6. Lưu trữ dữ liệu cấu trúc:

- Tuple có thể được sử dụng để lưu trữ dữ liệu có cấu trúc, chẳng hạn như kích thước của một hình ảnh (width, height), thông tin về một sản phẩm (tên, giá, mô tả).

7. Truyền nhiều đối số cho một hàm:
   - Có thể truyền một tuple làm đối số cho một hàm, tương tự như việc truyền nhiều đối số riêng biệt.
   - Tính bất biến: Giúp bảo vệ dữ liệu không bị thay đổi vô tình, tăng tính ổn định của chương trình.
   - Hiệu suất: Tuple thường nhanh hơn list một chút, đặc biệt khi sử dụng làm khóa trong dictionary.
   - Dễ đọc: Cấu trúc rõ ràng, dễ hiểu.
   - Ứng dụng đa dạng: Có thể sử dụng trong nhiều ngữ cảnh khác nhau.