**CÂU HỎI TỔNG HỢP**

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?

2. Viết chương trình xuất ra số ngẫu nhiên trong một đoạn bất kỳ bất cho

trước?

3. Khác biệt cơ bản giữa list và tuple?

4. Ứng dụng kiểu dữ liệu tuple trong thực tế?

Bài làm

**Câu 1. Phân biệt toán tử định dạng chuỗi và hàm định dạng chuỗi có sẵn trong gói thư viện chuẩn Python**

*Toán tử định dạng chuỗi (%) vs Hàm định dạng chuỗi (format())*

**Toán tử %:**

* Cú pháp: %s để thay thế bằng chuỗi, %d để thay thế bằng số nguyên, %f để thay thế bằng số thực, ...
* Cách thức: Dùng dấu % để đánh dấu vị trí cần thay thế và đặt các giá trị cần thay thế sau dấu %.

Ví dụ:

ten = "Zill"

tuoi = 25

print("Tên tôi là %s và tôi %d tuổi." % (ten, tuoi))

**Hàm format():**

* Cú pháp: {index} hoặc {name} để thay thế, đặt các giá trị cần thay thế trong hàm format().
* Cách thức: Dùng cặp dấu ngoặc nhọn để đánh dấu vị trí cần thay thế, bên trong có thể là chỉ số hoặc tên biến.

Ví dụ:

ten = "Bob"

diem = 8.5

print("Tên tôi là {} và điểm trung bình là {:.2f}".format(ten, diem))

***Năm ví dụ minh họa:***

**1. In thông tin người dùng:**

* Toán tử %:

ho = "Nguyen"

ten = "Van A"

ns = 2000

print("Họ và tên: %s %s. Năm sinh: %d" % (ho, ten, ns))

* Hàm format():

ho = "Nguyen"

ten = "Van A"

ns = 2000

print(f"Họ và tên: {ho} {ten}. Năm sinh: {ns}") # f-string (cách viết gọn của format())

**2. Tính toán diện tích hình tròn:**

* Toán tử %:

ban\_kinh = 5

pi = 3.14

print("Diện tích hình tròn bán kính %.2f là: %.2f" % (ban\_kinh, pi \* ban\_kinh\*\*2))

* Hàm format():

ban\_kinh = 5

pi = 3.14

print(f"Diện tích hình tròn bán kính {ban\_kinh:.2f} là: {pi \* ban\_kinh\*\*2:.2f}")

**3. In ngày tháng năm:**

* Toán tử %:

ngay = 15

thang = 4

nam = 2023

print("Hôm nay là ngày %d tháng %d năm %d" % (ngay, thang, nam))

* Hàm format():

ngay = 15

thang = 4

nam = 2023

print(f"Hôm nay là ngày {ngay:02d} tháng {thang:02d} năm {nam}") # Định dạng số có 2 chữ số

**4. In thông tin sản phẩm:**

* \*Toán tử %:

ten\_sp = "Táo"

gia = 15000

so\_luong = 2

print("Sản phẩm: %s. Giá: %d VNĐ/kg. Số lượng: %d kg" % (ten\_sp, gia, so\_luong))

* Hàm format():

ten\_sp = "Táo"

gia = 15000

so\_luong = 2

print(f"Sản phẩm: {ten\_sp}. Giá: {gia:,} VNĐ/kg. Số lượng: {so\_luong} kg") # Định dạng số với dấu phẩy

**5. In kết quả phép tính:**

* Toán tử %:

a = 10

b = 3

print("Tổng của %d và %d là: %d" % (a, b, a + b))

* Hàm format():

a = 10

b = 3

print(f"Tổng của {a} và {b} là: {a + b}")

**Kết luận:**

* Cả hai cách đều dùng để định dạng chuỗi.
* Hàm format() linh hoạt hơn:
  + Cho phép định dạng phức tạp hơn.
  + Sử dụng f-string giúp code gọn gàng hơn.
* Toán tử %:
  + Dễ hiểu đối với người mới bắt đầu.
  + Thường được sử dụng trong các phiên bản Python cũ hơn.

**Câu 2. Viết chương trình xuất ra số ngẫu nhiên trong một đoạn bất kỳ bất cho**

**trước?**

import random

*# Nhập giá trị thấp nhất và cao nhất từ người dùng*

try:

min\_value = int(input("Nhập giá trị thấp nhất của đoạn: "))

max\_value = int(input("Nhập giá trị cao nhất của đoạn: "))

*# Kiểm tra xem giá trị thấp nhất có nhỏ hơn giá trị cao nhất không*

if min\_value > max\_value:

print("Giá trị thấp nhất phải nhỏ hơn giá trị cao nhất.")

else:

*# Sinh số ngẫu nhiên trong khoảng từ min\_value đến max\_value*

random\_number = random.randint(min\_value, max\_value)

print(f"Số ngẫu nhiên trong đoạn [{min\_value}, {max\_value}] là: {random\_number}")

except ValueError:

print("Vui lòng nhập một số nguyên hợp lệ.")

**Câu 3. Khác biệt cơ bản giữa list và tuple?**

**List (Danh sách)**

* Tính chất: Có thể thay đổi (mutable)
* Sử dụng:
  + Lưu trữ một tập hợp các phần tử có thể thay đổi theo thời gian.
  + Thường được sử dụng khi bạn cần thêm, xóa hoặc sửa đổi các phần tử trong danh sách.
* Cú pháp: Dùng dấu ngoặc vuông [] để tạo một list.
  + Ví dụ: my\_list = [1, 2, 3, "apple", "banana"]

**Tuple (Bộ)**

* Tính chất: Không thể thay đổi (immutable)
* Sử dụng:
  + Lưu trữ một tập hợp các phần tử không thay đổi.
  + Thường được sử dụng để bảo vệ dữ liệu khỏi bị thay đổi vô tình hoặc khi bạn cần một cấu trúc dữ liệu không đổi.
  + Tuple cũng có thể được sử dụng làm key cho dictionary.
* Cú pháp: Dùng dấu ngoặc tròn () để tạo một tuple.
  + Ví dụ: my\_tuple = (1, 2, 3, "apple", "banana")

*Bảng so sánh:*

|  |  |  |
| --- | --- | --- |
| *Tính chất* | *List* | *Tuple* |
| Có thể thay đổi | Có | *Không* |
| Sử dụng dấu ngoặc | [] | () |
| Sử dụng làm key cho dictionary | *Không* | *Có* |
| Tốc độ truy cập | Tương đối chậm hơn tuple | |Nhanh hơn list |

|

Ví dụ minh họa:

# List

fruits = ["apple", "banana", "cherry"]

fruits.append("orange") # Thêm phần tử vào list

print(fruits) # Output: ['apple', 'banana', 'cherry', 'orange']

# Tuple

coordinates = (3, 4)

# coordinates[0] = 5 # Sẽ gây ra lỗi vì tuple không thể thay đổi

print(coordinates) # Output: (3, 4)

*Khi nào nên sử dụng list và tuple?*

* Sử dụng list: Khi bạn cần một cấu trúc dữ liệu linh hoạt, có thể thay đổi kích thước và nội dung.
* Sử dụng tuple: Khi bạn cần một cấu trúc dữ liệu không đổi, để bảo vệ dữ liệu hoặc khi bạn cần sử dụng nó làm key cho dictionary.

**Câu 4. Ứng dụng kiểu dữ liệu tuple trong thực tế?**

**1. Lưu trữ dữ liệu không đổi:**

* Tọa độ: Một cặp số (x, y) đại diện cho một điểm trên mặt phẳng. Vì tọa độ thường không thay đổi, tuple là một lựa chọn hoàn hảo.
* Màu sắc: Một bộ ba số (R, G, B) biểu diễn một màu sắc.
* Ngày tháng năm: Một bộ ba số (ngày, tháng, năm) biểu diễn một ngày cụ thể.

**2. Làm khóa cho dictionary:**

* Vì tuple là bất biến, chúng có thể được sử dụng làm khóa trong dictionary. Điều này rất hữu ích khi bạn muốn ánh xạ một cặp giá trị hoặc nhiều giá trị hơn vào một giá trị khác.
* Ví dụ: Một dictionary lưu trữ thông tin sinh viên, với khóa là tuple (mã số sinh viên, họ tên).

students = {

(123, "Nguyễn Văn A"): {"tuổi": 20, "lớp": "CNTT"},

(456, "Trần Thị B"): {"tuổi": 19, "lớp": "KTQT"}

}

**3. Trả về nhiều giá trị từ hàm:**

* Một hàm có thể trả về nhiều giá trị bằng cách đóng gói chúng trong một tuple.
* Ví dụ: Một hàm tính diện tích và chu vi của hình chữ nhật.

def tinh\_dien\_tich\_chu\_vi(chieu\_dai, chieu\_rong):

dien\_tich = chieu\_dai \* chieu\_rong

chu\_vi = 2 \* (chieu\_dai + chieu\_rong)

return dien\_tich, chu\_vi

**4. Đưa ra các hằng số:**

* Tuple được sử dụng để định nghĩa các hằng số mà không thể bị thay đổi trong quá trình thực thi chương trình.
* Ví dụ: Các ngày trong tuần, các tháng trong năm.

days\_of\_week = ("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")

**5. Tối ưu hóa hiệu suất:**

* Tuple thường nhanh hơn list trong việc truy cập các phần tử, đặc biệt là khi tuple có kích thước lớn.
* Tuple cũng tiêu tốn ít bộ nhớ hơn list.

**6. Các ứng dụng khác:**

* Định dạng dữ liệu: Tuple có thể được sử dụng để định dạng dữ liệu đầu vào hoặc đầu ra.
* Truyền tham số cho hàm: Tuple có thể được sử dụng để truyền một số lượng không xác định các tham số cho một hàm.