**Bài làm**

**1. Phân biệt toán tử định dạng chuỗi và hàm định dạng chuỗi trong Python**

Trong Python, có hai cách phổ biến để định dạng chuỗi: sử dụng toán tử định dạng chuỗi (`%`) và các hàm định dạng chuỗi như str.format() hoặc f-string (f"").

- Toán tử định dạng chuỗi (%): Đây là cách định dạng chuỗi cổ điển trong Python. Nó hoạt động giống như toán tử định dạng trong ngôn ngữ C.

Cú pháp:

Python

"Chuỗi có định dạng %s" % (giá trị)

***Ví dụ:***

*# Ví dụ 1*

name = "Alice"

print("Hello, %s!" % name)

*# Ví dụ 2*

age = 25

print("You are %d years old." % age)

*# Ví dụ 3*

price = 19.99

print("The price is $%.2f" % price)

*# Ví dụ 4*

data = (3, 4)

print("Coordinates: x=%d, y=%d" % data)

*# Ví dụ 5*

percentage = 85.5

print("You scored %.1f%% on the test." % percentage)

- Hàm định dạng chuỗi:

- str.format(): Cung cấp nhiều tính năng hơn cho việc định dạng chuỗi. Các tham số có thể được đặt theo vị trí hoặc tên.

Cú pháp:

"Chuỗi có định dạng {}".format(giá trị)

***Ví dụ:***

*# Ví dụ 1*

name = "Khoa"

print("Hello, {}!".format(name))

*# Ví dụ 2*

age = 30

print("You are {} years old.".format(age))

*# Ví dụ 3*

price = 15.75

print("The price is ${:.2f}".format(price))

*# Ví dụ 4*

data = {"x": 10, "y": 20}

print("Coordinates: x={x}, y={y}".format(\*\*data))

*# Ví dụ 5*

language = "Python"

print("{0} is fun, {0} is powerful.".format(language))

- F-string (f""): Đây là phương pháp định dạng chuỗi mới nhất trong Python (giới thiệu từ Python 3.6). Nó cho phép nhúng trực tiếp các biểu thức Python bên trong chuỗi.

Cú pháp:

f"Chuỗi có định dạng {biểu\_thức}"

***Ví dụ:***

*# Ví dụ 1*

name = "Eve"

print(f"Hello, {name}!")

*# Ví dụ 2*

age = 28

print(f"You are {age} years old.")

*# Ví dụ 3*

price = 12.50

print(f"The price is ${price:.2f}")

*# Ví dụ 4*

data = (7, 9)

print(f"Coordinates: x={data[0]}, y={data[1]}")

*# Ví dụ 5*

grade = 95.6

print(f"You scored {grade:.1f}% on the test.")

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

Để xuất ra số ngẫu nhiên trong một đoạn cho trước, bạn có thể sử dụng hàm randint() từ thư viện random.

Ví dụ:

import random

# Đoạn cho trước

min\_value = 10

max\_value = 50

# Sinh số ngẫu nhiên trong đoạn [min\_value, max\_value]

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

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

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

- List:

- Có thể thay đổi (mutable): Các phần tử trong list có thể được thay đổi sau khi tạo.

- Cú pháp: Sử dụng dấu ngoặc vuông [].

- Hiệu suất: Thường chậm hơn tuple do có thể thay đổi.

- Sử dụng: Thích hợp khi cần danh sách có thể thay đổi (thêm, xóa, sửa).

- Tuple:

• Không thể thay đổi (immutable): Các phần tử trong tuple không thể thay đổi sau khi tạo.

• Cú pháp: Sử dụng dấu ngoặc tròn ().

• Hiệu suất: Thường nhanh hơn list do không thể thay đổi.

• Sử dụng: Thích hợp khi cần một danh sách cố định, đảm bảo tính bất biến.

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

Tuple được sử dụng phổ biến trong các trường hợp sau:

• Truyền nhiều giá trị: Dùng tuple để trả về nhiều giá trị từ một hàm.

• Khoá trong từ điển: Tuple có thể dùng làm khoá trong dictionary vì chúng không thay đổi.

• Nhóm dữ liệu: Tuple được dùng để nhóm các giá trị khác nhau thành một đơn vị không thay đổi (ví dụ: tọa độ, thời gian).

***Ví dụ:***

# Trả về nhiều giá trị từ hàm

def get\_position():

return (10, 20)

x, y = get\_position()

# Tuple làm khoá trong từ điển

location = {(40.7128, -74.0060): "New York", (34.0522, -118.2437): "Los Angeles"}

# Nhóm dữ liệu

point = (4, 5) # Tọa độ điểm

date\_of\_birth = (1990, 8, 25) # Ngày sinh (năm, tháng, ngày)

-Hết-