**BÀI TẬP TỔNG HỢP**

**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?**

* **Toán tử định dạng chuỗi**

Chuỗi ký tự (chuỗi) là một loại dữ liệu rất hay được sử dụng trong [Python](https://topdev.vn/blog/python-la-gi/) , một từ, một đoạn văn bản đều là chuỗi kiểu. Chuỗi trong Python được đánh dấu bằng dấu nháy đơn ' hoặc dấu kép “, tuy nhiên nếu bắt đầu bằng dấu nào thì phải kết thúc bằng dấu đấy.

Cú pháp cũ và tương tự như trong C.

Ít linh hoạt hơn và có thể khó đọc với các định dạng phức tạp.

<chuỗi> **%**(giá trị thứ 1, giá trị thứ 2, .., giá trị thứ n – 1, giá trị thứ n)

**VD1: Định dạng số nguyên**

ten = "TN"

tuoi = 21

thongtin = " Tên tôi là %a và tôi %b tuổi." % (ten, tuoi)

print(thongtin)

**VD2: Định dạng số thực**

gia\_tri = 3.14159

chuoi = "Giá trị pi khoảng là %.2f." % gia\_tri

print(chuoi)\

**VD3: Định dạng chuỗi và số thực**

ten = "TN"

diem = 92.345

chuoi = "Sinh viên %s đạt điểm %.1f trong bài kiểm tra." % (ten, diem)

print(chuoi)

**VD4: Định dạng nhiều biến**

ten = "TN"

diem1 = 85

diem2 = 90

chuoi = "%s có điểm thi lần 1 là %d và điểm thi lần 2 là %d." % (ten, diem1, diem2)

print(chuoi)

**VD5: Định dạng ký tự đặc biệt**

chuoi = "Ký tự phần trăm là %%."

print(chuoi)

* **Hàm định dạng chuỗi (str.format() và f-strings):**

Cung cấp cách định dạng chuỗi hiện đại hơn và linh hoạt hơn.

str.format() cho phép sử dụng các chỉ định vị trí và định dạng linh hoạt.

f-strings cho phép nhúng các biểu thức trực tiếp vào chuỗi và dễ đọc hơn.

**VD1: Sử dụng str.format()**

ten = "tn"

tuoi = 21

chuoi = "Tên tôi là {} và tôi {} tuổi.".format(ten, tuoi)

print(chuoi)

**VD2: Định dạng số thực với str.format()**

gia\_tri = 3.14159

chuoi = "Giá trị pi khoảng là {:.2f}.".format(gia\_tri)

print(chuoi)

**VD3: Định dạng chuỗi và số thực với str.format()**

ten = "TN"

diem = 92.345

print("Sinh viên {0} đạt điểm {1:.1f} trong bài kiểm tra.", format(ten, diem))

**VD4: Sử dụng f-strings**

ten = "TN"

diem1 = 85

diem2 = 90

print(f"{ten} có điểm thi lần 1 là {diem1} và điểm thi lần 2 là {diem2}.")

**VD5: Định dạng số thực với f-strings**

gia\_tri = 3.14159

chuoi = f"Giá trị pi khoảng là {gia\_tri:.2f}."

print(chuoi)

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

a = random.randrange(1, 100, 5)

b = random.randrange(1, 100, 5)

tuso1 = math.sqrt(a) - math.sqrt(b)

mauso1 = 4 \* math.sqrt(a) - 4 \* math.sqrt(b)

tuso2 = math.sqrt(a) + (ab := math.sqrt(a \* b)\*\*0.25)

mauso2 = 4 \* math.sqrt(a) + 4 \* math.sqrt(b)

if mauso1 != 0 and mauso2 != 0:

ket\_qua = (tuso1 / mauso1) - (tuso2 / mauso2)

print(f"a = {a}, b = {b}")

print("Kết quả:", ket\_qua)

else:

print("Mẫu số bằng 0, không thể thực hiện phép tính.")

**Câu 3: Khác biệt cơ bản giữa list và tuple**

Lists là các đối tượng có thể thay đổi, có nghĩa là bạn có thể sửa đổi một đối tượng list sau khi nó được tạo.

Các tuple là các đối tượng bất biến, có nghĩa là bạn không thể sửa đổi một đối tượng tuple sau khi nó được tạo.

Dữ liệu thuộc kiểu list có thể thay đổi được, dữ liệu thuộc kiểu tuple không thể thay đổi được

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

Tuple cũng là một container được sử dụng rất nhiều trong các chương trình Python nên không gây thiệt hại gì cho List

Một Tuple bao gồm các yếu tố sau:

Được giới hạn bởi cặp ngoặc () , tất cả những gì nằm trong đó là những phần tử của Tuple.

Các phần tử của Tuple được phân tách bằng comma ( , ).

Tuple có khả năng chứa mọi giá trị, đối tượng trong Python.

Sử dụng dấu ngoặc đơn () và đặt giá trị bên trong cú pháp như sau:

(<giá trị thứ nhất>, <giá trị thứ hai>, .., <giá trị thứ n – 1>, <giá trị thứ n>)

**Ví dụ:**

tup = (1, 2, 3, 4)

tup

(1, 2, 3, 4)

>>> empty\_tup = ()  # khởi tạo tuple rỗng

>>> empty\_tup

()

>>> type(tup) # kiểu dữ liệu Tuple thuộc lớp tuple

<class 'tuple'>

**CÁCH TRUY CẬP CÁC GIÁ TRỊ TRONG TUPLES**

**Ví dụ:**

tupl1 = ('computersc', 'IT', 'CSE');

tupl2 = (1993, 2016);

tupl3 = (2, 4, 6, 8, 10, 12, 14, 16);

print ("tupl1[0]", tupl1[0])

print ("tupl3[2:4]", tupl3[2:4])

Đầu ra:

tupl1 [0] computerc

tupl3 [2: 4] (6, 8)