Cau 1:  
Toán tử % được sử dụng để định dạng chuỗi, giống như cách thức hoạt động trong ngôn ngữ C.

Vd:a=18

Tuoi=”toi nam nay %d tuoi” %a

Print(Tuoi)

Output: toi nam nay 18 tuoi

Vd2: pi = 3.14159265359

Bi="Giá trị của Pi là khoảng %.3f." % pi

print(Bi)

Output: "Giá trị của Pi là khoảng 3.141."

Vd3: name = "John"

ten = "Tên tôi là %s." % name

print(Ten)

# Output: "Tên tôi là John."

Vd4:

name = "Phong" age = 19

Tentuoi= "Tôi tên là %s và tôi %d tuổi." % (name, age)

print(Tentuoi)

# Output: "Tôi tên là Phongvà tôi 19 tuổi."

Vd5:

percentage = 75

formatted\_string = "Bạn đã hoàn thành %d%% công việc." % percentage print(formatted\_string)

# Output: "Bạn đã hoàn thành 75% công việc."

2. Hàm định dạng chuỗi (str.format())

Hàm str.format() là một cách linh hoạt và mạnh mẽ hơn để định dạng chuỗi.

Vd:a=18

Tuoi=”toi nam nay {} tuoi” format(a)

Print(Tuoi)

Output: “toi nam nay 18 tuoi”

# Ví dụ 2: Định dạng số thực

pi = 3.14159265359

formatted\_string = "Giá trị của Pi là khoảng {:.2f}.".format(pi)

print(formatted\_string)

# Output: "Giá trị của Pi là khoảng 3.14."

# Ví dụ 3: Định dạng chuỗi

name = "John"

formatted\_string = "Tên tôi là {}.".format(name)

print(formatted\_string)

# Output: "Tên tôi là John."

# Ví dụ 4: Định dạng nhiều giá trị

name = "Alice" age = 30

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

print(formatted\_string)

# Output: "Tôi tên là Alice và tôi 30 tuổi."

# Ví dụ 5: Định dạng chỉ số phần trăm

percentage = 75

formatted\_string = "Bạn đã hoàn thành {}% công việc.".format(percentage)

print(formatted\_string)

# Output: "Bạn đã hoàn thành 75% công việc."

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

trước?

start = int(input("Nhập số bắt đầu của đoạn: "))

end = int(input("Nhập số kết thúc của đoạn: "))

random\_number = random.randint(start, end)

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

Cau 2:

Python

import random

# Sinh một số ngẫu nhiên từ 1 đến 100

so\_ngau\_nhien = random.randint(1, 100)

print("Số ngẫu nhiên là:", so\_ngau\_nhien)

Cau 3:

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

1. Tính khả biến:

* List: Là kiểu dữ liệu có thể thay đổi (mutable). Bạn có thể thêm, xóa hoặc sửa đổi các phần tử trong một list sau khi nó được tạo.
* Tuple: Là kiểu dữ liệu bất biến (immutable). Một khi tuple được tạo, bạn không thể thay đổi các phần tử của nó.

2. Sử dụng dấu ngoặc:

* List: Được bao quanh bởi dấu ngoặc vuông [].
* Tuple: Được bao quanh bởi dấu ngoặc tròn ().

3. Sử dụng:

* List: Thường được sử dụng khi bạn cần một cấu trúc dữ liệu có thể thay đổi linh hoạt, ví dụ như lưu trữ một danh sách các số, chuỗi hoặc các đối tượng khác.
* Tuple: Thường được sử dụng để lưu trữ các dữ liệu không thay đổi, như các hằng số, các cặp khóa-giá trị trong một dictionary, hoặc khi bạn muốn đảm bảo rằng dữ liệu không bị thay đổi vô tình.

Cau 4:

Ứng dụng kiểu dữ liệu tuple trong thực tế?

-Lưu trữ dữ liệu không đổi

Vd: coordinates = (21.0285, 105.8542) # Tọa độ Hà Nội

-Làm khóa trong từ điển

Vd: coordinates = {(1, 2): "point A", (3, 4): "point B"}

-Trả về nhiều giá trị từ 1 hàm

Vd: def find\_min\_max(numbers):

return (min(numbers), max(numbers))

-Tạo các cấu trúc dữ liệu phức tạp

Vd: points = [(1, 2), (3, 4), (5, 6)]

-Sử dụng làm giá trị mặc định

Vd: default\_settings = ("English", "Metric", "24h")

language, units, time\_format = default\_settings

print(language)

print(units)

print(time\_format)

-Cải thiện hiệu suất

Vd: large\_tuple = (1, 2, 3) \* 1000000

-Phân tích cú pháp và giải nén dữ liệu

Vd: dimensions = (1920, 1080)

width, height = dimensions

print(f"Width: {width}, Height: {height}")