Họ và tên : Phan Lệ Quỳnh

MSSV: 23720041

CÂU 1:

1. Toán tử định dạng chuỗi (f-string):

\* Đặc điểm:

\* Được giới thiệu từ Python 3.6.

\* Cú pháp trực quan, đặt chữ f trước dấu ngoặc kép.

\* Thay thế giá trị biến trực tiếp vào chuỗi bằng cách đặt tên biến trong cặp dấu ngoặc nhọn {}.

\* Có thể thực hiện các phép tính, gọi hàm ngay trong cặp dấu ngoặc nhọn.

\* Ưu điểm:

\* Đọc dễ hơn, viết nhanh hơn.

\* Tránh được lỗi cú pháp thường gặp.

2. Hàm định dạng chuỗi format():

\* Đặc điểm:

\* Có từ các phiên bản Python trước đó.

\* Cú pháp linh hoạt hơn, sử dụng cặp dấu ngoặc nhọn {} làm placeholder và truyền giá trị vào bằng phương thức format().

\* Có thể định dạng số, ngày tháng, ...

\* Ưu điểm:

\* Hỗ trợ nhiều kiểu định dạng hơn.

**5 ví dụ minh họa:**

\* In thông tin người dùng:

\* f-string: print(f"User {user\_id} has {num\_points} points.")

\* format(): print("User {} has {} points.".format(user\_id, num\_points))

\* Tính toán và in kết quả:

\* f-string: print(f"The sum of {a} and {b} is {a + b}.")

\* format(): print("The sum of {} and {} is {}.".format(a, b, a + b))

\* Định dạng số:

\* f-string: print(f"Pi is approximately {math.pi:.3f}.")

\* format(): print("Pi is approximately {:.3f}.".format(math.pi))

\* Định dạng ngày tháng:

\* f-string: import datetime; print(f"Today is {datetime.date.today():%B %d, %Y}.")

\* format(): import datetime; print("Today is {:%B %d, %Y}".format(datetime.date.today()))

\* Gọi hàm trong f-string:

\* f-string: def greet(name): return f"Hello, {name}!"; print(greet("Alice"))

\* format(): def greet(name): return "Hello, {}!".format(name); print(greet("Alice"))

CÂU 2:

\* Nhập dữ liệu: Chương trình yêu cầu người dùng nhập vào hai số nguyên là giới hạn dưới và giới hạn trên của đoạn.

\* Định nghĩa hàm: Hàm so\_ngau\_nhien() nhận hai tham số là so\_dau và so\_cuoi, sử dụng hàm randint() của module random để tạo số ngẫu nhiên và trả về kết quả.

\* Gọi hàm và in kết quả: Chương trình gọi hàm so\_ngau\_nhien() với các giá trị đã nhập và in ra số ngẫu nhiên thu được.

CÂU 3:

Sự khác biệt cơ bản:

\* Tính bất biến:

\* List: Có thể thay đổi sau khi được tạo. Bạn có thể thêm, xóa, sửa đổi các phần tử trong list.

\* Tuple: Không thể thay đổi sau khi được tạo. Một khi đã tạo, bạn không thể thêm, xóa hoặc sửa đổi các phần tử trong tuple.

\* Sử dụng dấu ngoặc:

\* List: Được bao quanh bởi dấu ngoặc vuông [].

\* Tuple: Được bao quanh bởi dấu ngoặc tròn ().

\* Hiệu suất:

\* Tuple: Thường có hiệu suất tốt hơn list, đặc biệt là khi làm việc với các tập dữ liệu lớn, vì tính bất biến của nó giúp cho việc tối ưu hóa bộ nhớ và truy xuất dữ liệu hiệu quả hơn.

\* Sử dụng làm khóa trong dictionary:

\* List: Không thể sử dụng làm khóa trong dictionary vì list là một đối tượng có thể thay đổi.

\* Tuple: Có thể sử dụng làm khóa trong dictionary vì tuple là một đối tượng bất biến.

CÂU 4:

1. Lưu trữ các giá trị không đổi:

\* Tọa độ: Một điểm trên mặt phẳng có thể được biểu diễn bằng một tuple (x, y). Vì tọa độ của một điểm thường không thay đổi nên tuple là lựa chọn phù hợp.

\* Màu sắc: Màu sắc thường được biểu diễn dưới dạng một tuple gồm các giá trị RGB (Red, Green, Blue).

\* Ngày tháng: Một ngày tháng có thể được biểu diễn bằng một tuple (năm, tháng, ngày).

2. 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 ánh xạ một tập hợp các giá trị không đổi với một giá trị khác.

\* Ví dụ: Một dictionary có thể lưu trữ thông tin về sinh viên với khóa là một tuple (tên, họ) và giá trị là điểm số.

3. Trả về nhiều giá trị từ một hàm:

\* Một hàm trong Python chỉ có thể trả về một giá trị. Tuy nhiên, bạn có thể trả về nhiều giá trị bằng cách đóng gói chúng trong một tuple.

\* Ví dụ: Một hàm tính toán diện tích và chu vi của một hình chữ nhật có thể trả về một tuple chứa cả diện tích và chu vi.

4. Đưa ra các tham số cho một hàm:

\* Một số hàm có thể nhận nhiều tham số đầu vào. Các tham số này có thể được truyền dưới dạng một tuple.

\* Ví dụ: Hàm max() trong Python có thể nhận một tuple các số và trả về số lớn nhất.

5. Sử dụng trong các cấu trúc dữ liệu khác:

\* Tuple có thể được sử dụng như một phần tử của list hoặc các cấu trúc dữ liệu khác.