## 1. Đệ quy trong Python là gì?

Đệ quy trong Python hay còn gọi là **recursion python**. Nói về toán học thì đệ quy là thuật toán giải quyết bài toán bằng cách gọi lại chính thuật toán đó, thao tác này sẽ thực hiện liên tục cho đến khi gặp điều kiện dừng.

Đệ quy được thể hiện rất tốt khi áp dụng với hàm trong Python. Hàm đệ quy là chương trình sẽ gọi lại chính hàm đó và ngưng gọi khi gặp điều kiện dừng. Nếu quay lại kiến thức về vòng lặp thì bản chất đây cũng là một loại vòng lặp đặc biệt phải không các bạn.

Chương trình đệ quy sẽ có điều kiện dừng, nếu không nó sẽ tạo ra một vòng đời đệ quy vô hạn, điều này giống như vòng lặp while trong Python là bạn đã được học.

Hãy làm một ví dụ đơn giản đó là tính giai thừa của một số.

Ví dụ sử dụng đệ quy để tính giai thừa của 4 thì sẽ là 1\*2\*3\*4 = 24.

**Bước 1**: yêu cầu người dùng nhập vào số 4

**Bước 2**: Sử dụng đệ quy để lặp tính tích từ 4 trở về 1. Gọi x là giá trị cho mỗi lần lặp thì ta có điều kiện dừng là x = 1.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | def calc\_factorial(x):      if x == 1:          return 1      else:          return (x \* calc\_factorial(x-1))    num = 4  print("Dãy fibo của ", num, "là ", calc\_factorial(num)) |

Bạn hãy để ý bên trong phần thân của hàm calc\_factorial nhé, điều kiện để dừng đệ quy là x == 1, ngược lại chương trình sẽ thực hiện lặp đệ quy bởi đoạn code return (x \* calc\_factorial(x-1)).

Quy trình hoạt động của nó như sau:

|  |  |
| --- | --- |
| 1  2  3  4  5 | calc\_factorial(4)              # Lần 1 gọi với số 4  4 \* calc\_factorial(3)          # Lần 2 gọi với số 3  4 \* 3 \* calc\_factorial(2)      # Lần 3 gọi với số 2  4 \* 3 \* 2 \* calc\_factorial(1)  # Lần 4 gọi với số 1  4 \* 3 \* 2 \* 1                  # Cuối cùng ta được chuỗi này => kết quả là 24 |

Lần gọi đên quy cuối cùng vì giá trị tham số x truyền vào là 1 nên sẽ không thực hiện đệ quy nữa, sau đó trả kết quả là **24**.

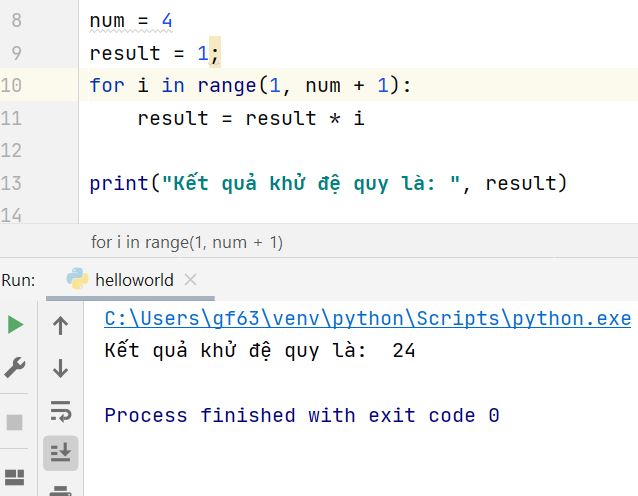
## 2. Khử đệ quy trong Python

Thực tế thì sử dụng đệ quy sẽ tốn rất nhiều tài nguyên của máy tính, bởi nó sẽ phải lưu trữ khá nhiều thông tin để tạo ra một biểu thức cuối cùng. Vì vậy người ta thưởng sử dụng khử đệ quy để chuyển đổi từ đệ quy thành vòng lặp.

Như bài toán tính Fibo trên ta có thể sử dụng trong vòng lặp rất dễ dàng và nhanh chóng.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | num = 4  result = 1;  for i in range(1, num + 1):      result = result \* i    print("Kết quả khử đệ quy là: ", result) |

Kết quả giống nhau:



## 3. Ưu điểm và nhược điểm của đệ quy Python

Sau đây là một vài ưu điểm và nhược điểm của đệ quy trong lập trình Python.

**Ưu điểm**:

* Các hàm đệ quy làm cho mã trông sạch sẽ
* Một tác vụ phức tạp có thể được chia thành các vấn đề phụ đơn giản hơn bằng cách sử dụng đệ quy.
* Tạo trình tự dễ dàng với đệ quy hơn là sử dụng một số lần lặp lồng nhau.

**Nhược điểm**:

* Đôi khi logic đằng sau đệ quy rất khó theo dõi.
* Chi phí gọi đệ quy rất tốn kém (không hiệu quả) vì chúng chiếm rất nhiều bộ nhớ và thời gian.
* Các hàm đệ quy khó gỡ lỗi.