CHƯƠNG 5 – ITERATION (Vòng lặp)

1. Cập nhật biến (Updating variables)

- Định nghĩa: thay giá trị biến dựa trên chính giá trị hiện tại (thường dùng cho bộ đếm hoặc tổng).

```
x = 0
x = x + 1   # tăng x thêm 1
x = x - 2   # giảm x 2
```

- Giải thích:

Khi Python thực hiện x = x + 1 nó đọc giá trị hiện tại của x, tính x + 1, rồi gán kết quả lại cho x. Nếu x chưa tồn tại (chưa khởi tạo) thì sẽ báo lỗi NameError. Vì vậy luôn khởi tạo trước khi cập nhật.

2. while – vòng lặp điều kiện

- Định nghĩa: lặp một khối lệnh miễn là điều kiện còn đúng (True).

```
n = 5
while n > 0:
    print(n)
    n = n - 1
print("Blastoff!")
```

- Giải thích:

- Trước mỗi lần lặp, Python kiểm tra n > 0.
- Nếu True, thực hiện thân vòng lặp (in n và giảm n).
- Khi điều kiện False, thoát vòng lặp và chạy lệnh tiếp theo ngoài vòng lặp.
- Dùng while khi số lần lặp không biết trước và phụ thuộc vào điều kiện.

3. Vòng lặp vô hạn (Infinite loop) & break

- Định nghĩa: vòng lặp tiếp tục mãi nếu điều kiện luôn đúng; break dùng để thoát sớm.

```
while True:
    line = input("> ")
    if line == "done":
        break
    print(line)
```

- Giải thích:

while True tạo vòng lặp vô hạn – phải có break trong thân vòng để kết thúc khi điều kiện mong muốn xảy ra (ở ví dụ: khi người dùng nhập "done"). Cách này thuận tiện để kiểm tra điều kiện ở bất kỳ vị trí trong thân vòng, không nhất thiết ở đầu.

4. continue – bỏ phần còn lại của vòng hiện tại

```
while True:
    line = input("> ")
    if line == "done":
        break
    print(line)
```

- Giải thích:

Khi gặp continue, chương trình không chạy phần code sau nó trong lần lặp đó mà quay lại đầu vòng lặp để kiểm tra điều kiện và tiếp tục.

5. Accumulator, Counter – tổng và đếm (pattern phổ biến)

- Định nghĩa:
- Accumulator (tổng) dùng để cộng dồn giá trị.
- Counter dùng để đếm số lượng phần tử thỏa điều kiệ

```
total = 0         # accumulator
count = 0         # counter
for num in [4,5,7]:
    total += num
    count += 1
average = total / count
```

- Giải thích:

Khởi tạo total = 0, count = 0. Trong mỗi vòng, thêm giá trị vào total và tăng count. Cuối cùng dùng để tính trung bình. Đây là mẫu rất thường gặp khi xử lý dữ liệu.

6. Tìm max/min trong vòng lặp

```
largest = None
for v in [3, 10, 2, 8]:
    if largest is None or v > largest:
        largest = v
# largest = 10
```

- Giải thích:

Khởi largest = None để biết lần đầu chưa có giá trị. Sau đó so sánh từng phần tử, cập nhật khi phần tử lớn hơn giá trị lưu. Tương tự cho smallest với v < smallest.