

Chương 5: Iteration.

- $x = x + 1 \rightarrow$ lấy giá trị cũ của x , cộng 1 rồi gán lại cho x .
- Nếu x chưa được tạo, Python báo lỗi vì phải tính về phải trước khi gán.
- Luôn khởi tạo biến trước khi cập nhật:

```
x = 0  
x = x + 1
```
- Tăng 1 gọi là increment, giảm 1 gọi là decrement.

Updating variables

- Vòng lặp vô hạn xảy ra khi điều kiện **while** luôn True.
- Dừng khi ta không biết trước lúc nào phải dừng.
- Có thể thoát vòng lặp bằng câu lệnh **break** bên trong thân vòng lặp.
- Ví dụ:

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

- Mục đích: Cho phép kiểm tra điều kiện dừng ở bất kỳ vị trí nào trong vòng lặp, không chỉ ở đầu.

"Infinite loops" and break

- **While** dùng lặp lại một công việc nhiều lần.
- Mỗi lần lặp gọi là iteration.
- Cần có biến lặp để thoát, tránh vòng lặp vô hạn.
- Ví dụ:

```
n = 5  
while n > 0:  
    print(n)  
    n = n - 1  
    print("Blastoff!")
```
- Quy trình: Kiểm tra \rightarrow chạy thân \rightarrow quay lại.

The while statement

- **continue** dùng để bỏ qua phần còn lại của vòng lặp và nhảy sang lượt lặp tiếp theo.
- Hữu ích khi muốn lọc dữ liệu trong lúc lặp.
- Ví dụ: bỏ qua các dòng bắt đầu bằng #, chỉ in các dòng khác.

```
while True:  
    line = input("> ")  
    if line[0] == '#':  
        continue  
    if line == "done":  
        break  
    print(line)  
    print("Done!")
```

Finishing iterations with continue

- Dùng **bisection**: kiểm tra giữa chương trình \rightarrow thu hẹp lỗi nhanh

Debugging

- **for** dùng khi biết trước danh sách cần lặp.
- Lặp qua từng phần tử: số, chữ, dòng, danh sách...
- Khác **while**: không kiểm tra điều kiện \rightarrow chỉ chạy hết các mục.
- Ví dụ:

```
friends = ["Joseph", "Glenn", "Sally"]  
for friend in friends:  
    print("Happy New Year:", friend)  
    print("Done!")
```

Definite loops using for

- Đếm (Counting): biến đếm tăng mỗi vòng.
- Tổng (Summing): biến accumulator tích lũy tổng
- Max / Min: lưu giá trị lớn nhất / nhỏ nhất hiện tại.

Loop patterns

- **Accumulator**: biến cộng dồn.
- **Counter**: biến đếm (bắt đầu từ 0).
- **Decrement**: giảm giá trị.
- **Initialize**: khởi tạo giá trị ban đầu.
- **Increment**: tăng giá trị.
- **Infinite loop**: vòng lặp không dừng.

Glossary