Bài 6: CÂU LỆNH LẶP WHILE

1. Cú pháp chung:

while <điều_kiện>:

<khối_lệnh>

➢ **Giải thích cú pháp:**

while: từ khóa bắt đầu vòng lặp.

<điều_kiện>: một biểu thức logic (kiểu True/False), biểu thức quan hệ, biểu thức số học, hằng hoặc biến kiểu xâu.

<khối_lệnh>: các câu lệnh lùi vào (thụt đầu dòng) sẽ được thực hiện lặp lại.

➢ **Cách thức thực hiện:**

1. Kiểm tra điều kiện.
2. Nếu điều kiện đúng (hoặc khác 0 hoặc rỗng) → thực hiện khối lệnh.
3. Sau khi chạy xong khối lệnh → quay lại kiểm tra điều kiện lần nữa.
4. Lặp lại cho đến khi điều kiện sai → thoát vòng lặp.

👉 Đây là vòng lặp lặp với số lần chưa biết trước, chỉ dừng khi điều kiện không còn đúng.

➢ **Ví dụ minh hoạ:**

* Ví dụ 1: In ra các số từ 1 đến 5

```
i = 1
while i <= 5:
    print(i, end=" ")
    i += 1
```

=>> 1 2 3 4 5

* Ví dụ 2: Tính tổng các số từ 1 đến n

```
n = int(input("Nhập n: "))
tong = 0
i = 1
while i <= n:
    tong += i
    i += 1
print("Tổng từ 1 đến", n, "là:", tong)
```

=>> Nhập n: 5

Tổng từ 1 đến 5 là: 15

* Ví dụ 3: Nhập mật khẩu đúng mới thoát

```
mat_khau_dung = "python123"
mat_khau = ""

while mat_khau != mat_khau_dung:
    mat_khau = input("Nhập mật khẩu: ")
print("Đăng nhập thành công!")
```

- **Chú ý:**

- Luôn cần có câu lệnh thay đổi giá trị biến điều kiện (vd: i += 1) để vòng lặp không chạy vô hạn.
- Nếu điều kiện ban đầu sai → vòng lặp không thực hiện lần nào.
- Để thoạt khỏi vòng lặp vô hạn lần, ta bấm tổ hợp phím Ctrl + C.

2. Câu lệnh break và câu lệnh continue.

❖ **Câu lệnh break**

- Giải thích:

break dùng để thoát hẳn khỏi vòng lặp (kể cả khi điều kiện vẫn còn đúng).
Sau khi gặp break, chương trình sẽ dừng vòng lặp ngay lập tức và chạy tiếp các lệnh phía sau vòng lặp.

- Ví dụ : Tìm số đầu tiên chia hết cho 7

```
i = 1
while i <= 20:
    if i % 7 == 0:
        print("Số đầu tiên chia hết cho 7 là:", i)
        break
    i += 1
```

=>> Số đầu tiên chia hết cho 7 là: 7 (Ở đây, vòng lặp while dừng ngay khi tìm thấy số 7 nhờ break)

❖ **Câu lệnh continue**

- **Giải thích:**

continue dùng để bỏ qua phần còn lại của vòng lặp hiện tại, và chuyển sang lần lặp tiếp theo. Điều kiện của vòng lặp vẫn được kiểm tra như bình thường.

- **Ví dụ:** In ra các số từ 1 đến 10, bỏ qua số 5

```
i = 1
while i <= 10:
    if i = = 5:
        i += 1
        continue   # bỏ qua phần in số 5
    print(i, end=" ")
    i += 1
```

=>> 1 2 3 4 6 7 8 9 10

(Ở đây, khi i = 5, lệnh continue được thực hiện → bỏ qua print(i))

- **So sánh:**
  - break: dừng hẳn vòng lặp.
  - continue: bỏ qua lần lặp hiện tại, tiếp tục vòng lặp mới.
- **Dùng** break, continue trong vòng lặp for

Ví dụ:

1) In các số từ 1 → 10, bỏ qua số 5

```
for i in range(1, 11):
    if i == 5:
        continue
    print(i, end=" ")
```

=>> 1 2 3 4 6 7 8 9 10

2) Tìm số đầu tiên chia hết cho 5 trong dãy 1 → 10

```
for i in range(1, 11):
    if i % 5 == 0:
        print("Số đầu tiên chia hết cho 5 là:", i)
        break
```

3) In các số từ 1 → 10, trong đó:

- Bỏ qua số 5 (continue)
- Dừng vòng lặp khi gặp số 8 (break)

```
for i in range(1, 11):
    if i == 5:
        continue   # bỏ qua in số 5
    if i == 8:
        break      # dừng hẳn vòng lặp
    print(i, end=" ")
```

———— BÀI TẬP ----------

Bài 1: In ra các số từ 1 đến 10.

Code:

```
i = 1
while i <= 10:
    print(i, end=" ")
    i += 1
```

Kết quả minh họa:

Kết quả: 1 2 3 4 5 6 7 8 9 10

Bài 2: Nhập vào số nguyên dương n. Tính tổng các chữ số của n.

Code:

```
n = int(input("Nhập số n: "))
tong = 0
while n > 0:
    tong += n % 10
    n //= 10
print("Tổng các chữ số:", tong)
```

Kết quả minh họa:
Ví dụ: n=1234 → Kết quả: 10

Bài 3: Nhập vào một số nguyên dương, nếu sai yêu cầu nhập lại.

Code:

```
n = -1
while n <= 0:
    n = int(input("Nhập số nguyên dương: "))
print("Bạn đã nhập:", n)
```

Bài 4: Nhập vào các số, dừng khi nhập số âm.

Code:

```
while True:
    n = int(input("Nhập số (nhập số âm để dừng): "))
    if n < 0:
        break
    print("Bạn nhập:", n)
```

Bài 5: In ra các số từ 1 đến 10, bỏ qua số 5.

Code:

```
i = 0
while i < 10:
    i += 1
    if i == 5:
        continue
    print(i, end=" ")
```

Kết quả minh họa:
Kết quả: 1 2 3 4 6 7 8 9 10

Bài 6: Tính giai thừa của số n.

Code:

```
n = int(input("Nhập n: "))
gt = 1
i = 1
```

```
while i <= n:
    gt *= i
    i += 1
print(f"{n}! = {gt}")
```

Kết quả minh họa:
Ví dụ: n=5 → 120

Bài 7: Kiểm tra số nguyên tố.

Code:

```
n = int(input("Nhập số n: "))
if n < 2:
    print("Không phải số nguyên tố")
else:
    i = 2
    while i * i <= n and n % i != 0:
        i += 1
    if i * i > n:
        print("Là số nguyên tố")
    else:
        print("Không phải số nguyên tố")
```

Bài 8: In ra dãy Fibonacci có n số hạng.

Code:

```
n = int(input("Nhập n: "))
a, b = 0, 1
i = 0
while i < n:
    print(a, end=" ")
    a, b = b, a + b
    i += 1
```

Kết quả minh họa:
Ví dụ: n=8 → 0 1 1 2 3 5 8 13

Bài 9: Tìm chữ số lớn nhất của n.

Code:

```
n = int(input("Nhập n: "))
max_digit = 0
while n > 0:
    digit = n % 10
    if digit > max_digit:
        max_digit = digit
```

```
    n //= 10
print("Chữ số lớn nhất:", max_digit)
```

Kết quả minh họa:
Ví dụ: n=9831 → 9

Bài 10: Nhập vào số nguyên n, in ra số đảo ngược của n.

Code:

```
n = int(input("Nhập n: "))
dao = 0
while n > 0:
    dao = dao * 10 + n % 10
    n //= 10
print("Số đảo ngược:", dao)
```

Kết quả minh họa:
Ví dụ: n=1234 → 4321