NỀN TẢNG CÔNG NGHỆ THÔNG TIN 2

CÁCH SỬ DỤNG VÒNG LẶP FOR, WHILE

LaTeX bởi Ngô Hoàng Tùng

Cấu trúc lặp FOR, WHILE

I. Mục tiêu

a. Cấu trúc lặp FOR, WHILE

b. Cách sử dụng vòng lặp FOR, WHILE

c. Cách sử dụng vòng lặp lồng nhau

d. Cách sử dụng vòng lặp với mảng 1 chiều

e. Cách sử dụng vòng lặp với mảng 2 chiều

II. Nội dung

a. Cấu trúc lặp FOR, WHILE

- Cấu trúc lặp FOR: Dùng để lặp một đoạn mã với số lần lặp xác định.
- Cấu trúc lặp WHILE: Dùng để lặp một đoạn mã khi điều kiện là đúng.

b. Cách sử dụng vòng lặp FOR, WHILE

- Vòng lặp FOR:
  Cú pháp:

```
for i in range(start, end, step):
    # Do something
```

  Hoặc:

```
for i in range(start, end, step):
    # Do something
```

  Giải thích:
  - `start`: giá trị bắt đầu (mặc định là 0)
  - `end`: giá trị kết thúc (không bao gồm giá trị này)
  - `step`: bước nhảy (mặc định là 1)
  - `range(end)`: lặp từ 0 đến end-1
  - `range(start, end)`: lặp từ start đến end-1
  - `range(start, end, step)`: lặp từ start đến end-1 với bước nhảy là step

  Ví dụ:

```
for i in range(5):
    print(i)  # In ra 0, 1, 2, 3, 4
```

  – range(5) tương đương với range(0, 5, 1) hoặc range (0, 5).

- Vòng lặp WHILE:
  Cú pháp:

```
while condition:
    # Do something
```

  Giải thích:
  - `condition`: điều kiện để tiếp tục lặp (nếu điều kiện là đúng thì lặp)

  Ví dụ 1:

```
i = 0
while i < 5:
    print(i)  # In ra 0, 1, 2, 3, 4
    i += 1
```

  – Vòng lặp while sẽ tiếp tục lặp cho đến khi điều kiện là sai.
  Ví dụ 2:

```
while True:
    print("Hello, World!")
```

  – Vòng lặp while sẽ lặp vô hạn vì điều kiện luôn đúng. Để dừng vòng lặp, bạn có thể sử dụng lệnh `break`.

```
while True:
    print("Hello, World!")
    break
```

c. Cách sử dụng vòng lặp lồng nhau

- Vòng lặp lồng nhau là khi bạn đặt một vòng lặp bên trong một vòng lặp khác. Điều này cho phép bạn lặp qua nhiều cấp độ dữ liệu.
- Cú pháp:

```
for i in range(n):
    for j in range(m):
        # Do something
```

  Hoặc:

```
while condition1:
    while condition2:
        # Do something
```

  Giải thích:
  - Vòng lặp ngoài sẽ lặp qua các giá trị từ 0 đến n-1.
  - Vòng lặp trong sẽ lặp qua các giá trị từ 0 đến m-1 cho mỗi giá trị của vòng lặp ngoài.

  Ví dụ 1:

```
for i in range(3):
    for j in range(2):
        print(f"i: {i}, j: {j}")
```

– Kết quả sẽ là:
i: 0, j: 0
i: 0, j: 1
i: 1, j: 0
i: 1, j: 1
i: 2, j: 0
i: 2, j: 1
Ví dụ 2:

```
i = 0
while i < 3:
    j = 0
    while j < 2:
        print(f"i: {i}, j: {j}")
        j += 1
    i += 1
```

– Kết quả như ví dụ 1.

d. Cách sử dụng vòng lặp với mảng 1 chiều

- Khi bạn có một mảng 1 chiều, bạn có thể sử dụng vòng lặp lồng nhau để truy cập từng phần tử của mảng.
- Cú pháp:

```
for i in range(len(array)):
    # Do something with array[i]
```

Hoặc:

```
while condition:
    # Do something with array[i]
```

Giải thích:

– Sử dụng vòng lặp for hoặc while để lặp qua các chỉ số của mảng.
– Truy cập từng phần tử của mảng bằng cách sử dụng chỉ số.

Ví dụ:

```
array = [1, 2, 3, 4, 5]
for i in range(len(array)):
    print(array[i], end=' ')
```

– Kết quả sẽ là: 1 2 3 4 5
Hoặc:

```
array = [1, 2, 3, 4, 5]
i = 0
while i < len(array):
    print(array[i], end=' ')
    i += 1
```

– Kết quả cũng sẽ là: 1 2 3 4 5

e. Cách sử dụng vòng lặp với mảng 2 chiều

- Khi bạn có một mảng 2 chiều, bạn có thể sử dụng vòng lặp lồng nhau để truy cập từng phần tử của mảng.

- Cú pháp:

```
for i in range(len(array)):
    for j in range(len(array[i])):
        # Do something with array[i][j]
```

Hoặc:

```
while condition1:
    while condition2:
        # Do something with array[i][j]
```

Giải thích:

  - Sử dụng vòng lặp for hoặc while để lặp qua các chỉ số của mảng 2 chiều.
  - Truy cập từng phần tử của mảng bằng cách sử dụng chỉ số hàng và cột.

Ví dụ:

```
array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
for i in range(len(array)):
    for j in range(len(array[i])):
        print(array[i][j], end=' ')
```

– Kết quả sẽ là: 1 2 3 4 5 6 7 8 9

Hoặc:

```
array = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
i = 0
while i < len(array):
    j = 0
    while j < len(array[i]):
        print(array[i][j], end=' ')
        j += 1
    i += 1
```

– Kết quả cũng sẽ là: 1 2 3 4 5 6 7 8 9

III. Bài tập

Bài 1:

Cho mảng 1 chiều gồm n số nguyên dương, hãy tính tổng các số trong mảng.

Input:

Dòng đầu chứa số nguyên dương duy nhất n ($0 < n < 1000$) là số lượng phần tử của mảng.

Dòng thứ hai chứa n số nguyên là các phần tử của mảng $a_0, a_1, \ldots, a_{n-1}$ ($-10^9 \leq a_i \leq 10^9$).

Output:

In ra một số nguyên là tổng các phần tử của mảng.

Input	Output
5 9 -5 20 -10 5	19

Bài 2:

Cho n và m đều là số nguyên dương. Hãy xuất 1 ma trận $a_{i,j}$ có kích thước n x m với $a_{i,j}$ tăng dần từ 1 tới n x m và theo đường ZigZag.

Input:

Dòng đầu chứa hai số nguyên dương n, m ($1 \leq n, m \leq 1000$) là kích thước của ma trận.

Output:

In ra ma trận $a_{i,j}$ theo định dạng như ví dụ dưới đây.

Input	Output
3 4	1 2 3 4 8 7 6 5 9 10 11 12

Bài 3:

Cho n là số nguyên dương. Hãy xuất ra dãy số nguyên tố bé hơn hoặc bằng n.

Input:

Dòng đầu chứa số nguyên dương duy nhất n ($1 \leq n \leq 100$) là số nguyên cần kiểm tra.

Output:

In ra các số nguyên tố bé hơn hoặc bằng n.

Input	Output
10	2 3 5 7

Bài 4:

Cho n, m là số nguyên dương và 1 ma trận n x m. Hãy sắp xếp các phần tử trong ma trận theo thứ tự tăng dần theo cả cột và dòng.

Input:

Dòng đầu chứa hai số nguyên dương n, m ($1 \leq n, m \leq 100$) là kích thước của ma trận.

Dòng thứ hai chứa n x m số nguyên là các phần tử của ma trận.

Output:

In ra các phần tử của ma trận đã sắp xếp theo thứ tự tăng dần.

Input	Output
3 3 5 1 7 9 11 3 2 6 4	1 4 6 2 5 7 3 9 11

HẾT