제어문

1. 조건문

▶ if

• if 다음의 조건식이 참(True)이면 들여쓰기 한 문장을 실행한다.

• if 문장 끝에는 콜론(:)을 입력

▶ if ~ else

• 조건식이 참일 경우 실행할 문장과 거짓일 경우 실행할 문장이 다를 경우 if구문에 else구문을 추가한다.

• else구문은 if문의 조건식이 False일 경우 실행하는 블록을 정의한다.

• else는 단독으로 사용될 수 없다.

▶ if ~ elif ~ else

• 여러 개의 조건식을 사용하려면 elif구문을 이용한다.

• elif는 단독으로 사용할 수 없다.

2. 반복문

▶ for ~ in 반복문

• 나열 가능한 자료에서 자료를 모두 소비할 때까지 처리한다.

```
for 변수 in 나열 가능한 자료 :
        변수의 값을 처리할 문장 1
        변수의 값을 처리할 문장 2
```

변수 : 단일 변수 또는 여러 개의 변수가 올 수 있다.

자료 : 리스트, 튜플, 딕셔너리, 셋 등이 될 수 있다.

▶ range(start, stop, step)

• for 문장의 items 객체 위치에 range() 함수를 이용하여 반복문을 실행할 수 있다.

```
ex) for I in range(0, 10) :
        print(I, end='₩t')
=> 0 1 2 3 4 5 6 7 8 9
```

• range() 함수를 이용하면 인덱스 위주의 반복을 실행할 수 있다.

• range() 함수의 start가 생략되면 0부터 시작.

• step은 얼마씩 증가시킨 값을 갖게 할 것인지를 결정.

▶ **while**

- 조건문이 참일 동안 계속 실행한다.

```
while 조건문 :
        조건이 참일 동안 반복 실행할 구문
```

- break를 만나면 break를 포함하는 반복문을 완전히 탈출한다.
- continue는 반복문 내에서 continue 이후의 문장을 건너뛴다.

3. 중첩 루프

```
for row in rows:
        바깥 반복문을 처리할 문장
        for data in row:
                안쪽 반복문을 처리할 문장
```

- 중첩 루프는 2차원 이상의 구조인 데이터를 다룰 때 사용한다.
- 2차원 리스트 인덱싱 : 변수명[행][열]
- 3차원 리스트 인덱싱 : 변수명[면][행][열]

4. 중첩 루프 탈출

- break문을 사용하면 조건에 따라 반복문을 종료한다.
- continue문을 사용하면 현재 반복문 블록을 실행하지 않고 다음 반복으로 이동한다.
- 예외처리를 이용하여 중첩 루프를 탈출한다.