13. 파이썬 - 반복문(for, while)

Made By. 김규일

- 보통 반복문은 for, while 문을 사용한다.
- for 문의 경우 횟수에 따라 반복을 결정하고 while의 경우 조건에 따라 반복을 결정한다.
- 따라서 보통 몇 회 반복, 리스트의 처음부터 끝까지 등 횟수에 관련된 반복은 for문을 사용하고 반복하다 이런 경우 정지, 이런 경우까지 반복 등 조건에 관련된 반복은 while 문을 사용한다.
- 무슨 반복문을 사용하는 것은 정답이 없으니 상황에 맞게 반복문을 선택해서 사용하자

for 문

- for문은 보통 in과 range 함수로 사용한다.
- in의 경우 보통 튜플이나 리스트 등에서 처음부터 끝까지 반복을 수행할 때 사용한다.
- 아래는 in을 사용하는 for 문의 구조와 예시이다.

```
for 변수 in 리스트(또는 튜플, 문자열):
    수행할 문장1
    수행할 문장2
    ...

>>> test_list = ['one', 'two', 'three']
>>> for i in test_list:
...     print(i)
...
one
two
three
```

- 위의 예시처럼 반복문은 리스트의 처음부터 끝까지 반복을 한다.
- 또한 반복문이 진행될 때 리스트의 각 값들은 변수 i에 저장된다.
- 즉, i가 리스트의 처음 값부터 끝 값까지 변화해야 반복문이 종료된다.
- 위의 방식을 활용하여 여러 개의 값을 가진 자료형에도 사용할 수 있다.

```
>>> a = [(1,2), (3,4), (5,6)]
>>> for (first, last) in a:
...     print(first + last)
...
3
7
11
```

- 위의 예시에서는 각각 first, last 변수가 튜플의 왼쪽과 오른쪽 값을 가리킨다.
- 이를 통해 튜플, 맵 등 다양한 자료형에 for, in 조합을 활용할 수 있다.

- 반복문에는 break와 continue가 존재한다.
- break는 언제든지 반복문을 끝낼 수 있는 예약어이고 continue는 뒤의 코드를 무시한채 다음 반복으로 넘어가는 예약어이다.

```
# marks2.py
marks = [90, 25, 67, 45, 80]

number = 0
for mark in marks:
    number = number +1
    if mark < 60:
        continue
    print("%d번 학생 축하합니다. 합격입니다. " % number)

C:\doit>python marks2.py
1번 학생 축하합니다. 합격입니다.
3번 학생 축하합니다. 합격입니다.
5번 학생 축하합니다. 합격입니다.
```

- 위의 코드는 continue의 예시를 보여준다.
- 60점 아래는 if 문에 걸리게 되고 continue를 통해 다음 코드를 실행하는 것이 아닌 다음 반복문으로 넘어간다.
- 따라서 아래와 같은 출력을 보인다.
- break 또한 비슷하게 if문과 함께 사용되며 바로 모든 남은 반복을 멈추고 반복문을 탈출한다.

- range 함수는 횟수를 결정한다.

```
>>> add = 0
>>> for i in range(1, 11):
...     add = add + i
...
>>> print(add)
55
```

- 보통 위의 형태로 사용한다.
- i는 range 함수를 통해 0부터 10이 된다.
- 여기서 중요한 점은 range(1, 11)일 때 i가 1~11이 아닌 1~11-1 이라는 점이다.
- range() 의 오른쪽 값은 반복 횟수에 포함이 안되고 -1 된 만큼의 값까지 반복한다.
- 아래는 range() 를 이용한 구구단의 예시이다.

```
>>> for i in range(2,10):        # 1번 for문
...     for j in range(1, 10):   # 2번 for문
...         print(i*j, end=" ")
...     print('')
...
2 4 6 8 10 12 14 16 18
3 6 9 12 15 18 21 24 27
4 8 12 16 20 24 28 32 36
5 10 15 20 25 30 35 40 45
6 12 18 24 30 36 42 48 54
7 14 21 28 35 42 49 56 63
8 16 24 32 40 48 56 64 72
9 18 27 36 45 54 63 72 81
```

리스트 컴프리헨션(List Comprehension)

- Python에는 간단하게 리스트 안에 for문을 삽입할 수 있는 기능을 제공한다.
- 다음의 예시를 보자

```
>>> a = [1,2,3,4]
>>> result = []
>>> for num in a:
...     result.append(num*3)
...
>>> print(result)
[3, 6, 9, 12]
```

- 위의 코드는 반복문을 통해 리스트에 3의 배수를 입력하는 코드이다.

- 이를 리스트 컴프리헨션을 이용하여 한줄의 코드로 사용할 수 있다.

```
>>> a = [1,2,3,4]
>>> result = [num * 3 for num in a]
>>> print(result)
[3, 6, 9, 12]


>>> a = [1,2,3,4]
>>> result = [num * 3 for num in a if num % 2 == 0]
>>> print(result)
[6, 12]
# [표현식 for 항목 in 반복가능객체 if 조건문]

[표현식 for 항목1 in 반복가능객체1 if 조건문1
        for 항목2 in 반복가능객체2 if 조건문2
        ...
        for 항목n in 반복가능객체n if 조건문n]

>>> result = [x*y for x in range(2,10)
...               for y in range(1,10)]
>>> print(result)
[2, 4, 6, 8, 10, 12, 14, 16, 18, 3, 6, 9, 12, 15, 18, 21, 24, 27, 4, 8, 12, 16,
20, 24, 28, 32, 36, 5, 10, 15, 20, 25, 30, 35, 40, 45, 6, 12, 18, 24, 30, 36, 42
, 48, 54, 7, 14, 21, 28, 35, 42, 49, 56, 63, 8, 16, 24, 32, 40, 48, 56, 64, 72,
9, 18, 27, 36, 45, 54, 63, 72, 81]
```

- 리스트 컴프리헨션은 자주 보이는 형태이니 익혀두면 좋다.

while

- for 문과 다르게 while은 조건을 통해 반복을 진행한다.
- 기본 형태는 다음과 같다.

```
while <조건문>:
    <수행할 문장1>
    <수행할 문장2>
    <수행할 문장3>
    ...
```

- 조건문의 조건이 참일 경우 반복을 하고 거짓이 될 경우 반복문을 탈출한다.
- 다음의 예시 코드를 보자

```
>>> treeHit = 0
>>> while treeHit < 10:
...     treeHit = treeHit +1
...     print("나무를 %d번 찍었습니다." % treeHit)
...     if treeHit == 10:
...         print("나무 넘어갑니다.")
...
나무를 1번 찍었습니다.
나무를 2번 찍었습니다.
나무를 3번 찍었습니다.
나무를 4번 찍었습니다.
나무를 5번 찍었습니다.
나무를 6번 찍었습니다.
나무를 7번 찍었습니다.
나무를 8번 찍었습니다.
나무를 9번 찍었습니다.
나무를 10번 찍었습니다.
나무 넘어갑니다.
```

- 위의 코드는 treeHit가 10이 아니라면, 즉 treeHit < 10이 참이라면 반복한다.
- 따라서 계속 반복을 하면서 threeHit = treeHit + 1 이 계속 진행된다.
- 결국 treeHit가 10이 되는 순간 반복을 하지 않고 반복문을 멈추게 된다.
- 여기서 중요한 점은 반복문의 마지막 시작은 treeHit가 9일 때이다. 10이 되는 순간 treeHit < 10은 거짓이 되기 때문이다.

- while문의 조건문 또한 앞서 배운 if 문의 조건문처럼 불 타입으로 끝나는 연산들의 조합으로 조건문을 만들 수 있다.

- while 문 같은 경우 메뉴 만들기와 같은 곳에 활용 가능하다.

```
>>> number = 0
>>> while number != 4:
...     print(prompt)
...     number = int(input())
...

1. Add
2. Del
3. List
4. Quit

Enter number:
1

1. Add
```

```
2. Del
3. List
4. Quit
```

- while 또한 break, continue 모두 사용 가능하다.
- 아래는 break의 예시이다.

```
>>> coffee = 10
>>> money = 300
>>> while money:
...     print("돈을 받았으니 커피를 줍니다.")
...     coffee = coffee -1
...     print("남은 커피의 양은 %d개입니다." % coffee)
...     if coffee == 0:
...         print("커피가 다 떨어졌습니다. 판매를 중지합니다.")
...         break
...
```

- while의 경우 조건문이 항상 참이되는 경우라면 무한루프에 빠질 수 있으니 주의하자
- 아래는 강제로 무한루프를 만든 while 문의 예시이다.

```
>>> while True:
...     print("Ctrl+C를 눌러야 while문을 빠져나갈 수 있습니다.")
...
Ctrl+C를 눌러야 while문을 빠져나갈 수 있습니다.
Ctrl+C를 눌러야 while문을 빠져나갈 수 있습니다.
Ctrl+C를 눌러야 while문을 빠져나갈 수 있습니다.
....

# Ctrl+C는 터미널 창의 명령 취소
```

- 누군가의 반복문이 이해가 안가거나 반복문을 어떻게 설계할지 모르겠다면 변화하는 값을 하나씩 써 가면서 표나 그림을 그려보면 쉽게 이해할 수 있다.