알고리즘 사고와 반복

08강

반복 필요성

- '터틀 그래픽을 이용한 도형 그리기' 과정 중 반복된 코드부분 발생
- 반복된 코드 부분들은 반복문 형태로 표현 가능
- 프로그램 코드를 간결하게 작성하고 쉽게 파악 가능
- 반복문 = 동일한 문장이나 부분을 지정된 횟수나 조건에 따라 여러 번 반복하는 구조

```
>>> import turtle
>>> turtle.shape("turtle")
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>> turtle.right(90)
>>> turtle.forward(100)
>>>
```

반복

반복

반복

반복

```
>>> import turtle
>>> turtle.shape("turtle")
>>> for i in range(4) :
        turtle.right(90)
        turtle.forward(100)

>>>
```

반복문

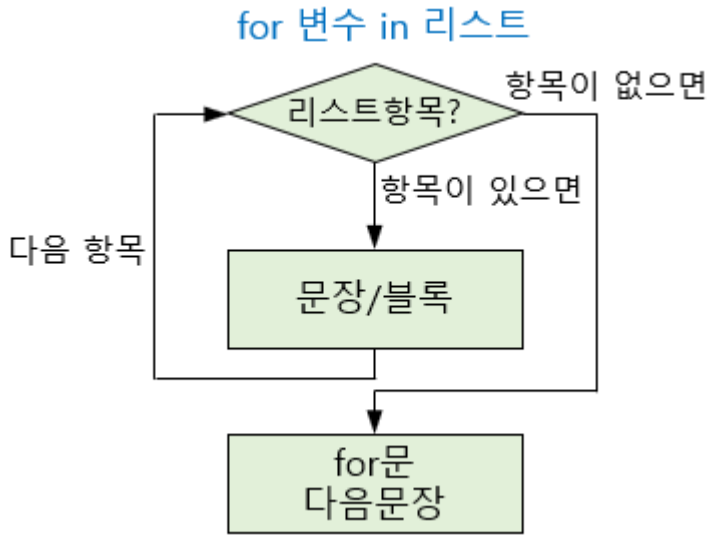
TIP

루프(loop)

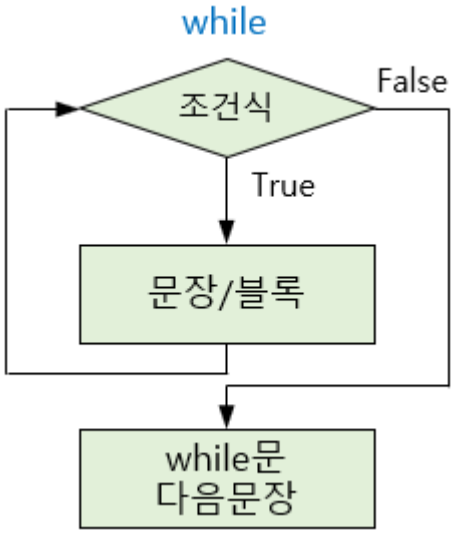
반복 구조는 실행의 흐름이 위에서 아래로, 아래에서 위로 회전하듯이 실행되며, 실행되는 과정이 마치 회전되는 경우와 비슷하다 하여 루프(loop)라고도 한다.

파이썬 프로그램 내 반복 (for 문 / while 문)

- for 문
  - 지정된 횟수만큼 반복하는 횟수 제어 반복
  - 정확한 반복 횟수를 미리 아는 경우에 사용 가능



- while 문
  - 지정 조건을 만족할 때 계속 반복되는 조건 제어 반복
  - 조건을 만족하면 (조건이 참인 경우) 계속 반복
  - 조건을 만족하지 않으면(조건이 거짓인 경우) 반복을 종료



횟수만큼 반복 (리스트 이용)

- 리스트(list)
  - 순서를 갖는 여러 가지 값들을 담을 수 있는 자료형
  - [ ] 안에 리스트 원소를 순서대로 나열

```
list1 = [1, 2, 3, 4, 5]
list2 = ['a', 'b', 'c']
list3 = [ 1, 'a', "abc", [1, 2, 3]]
```

- 리스트 이용 for 문
  - 리스트를 이용하여 지정된 횟수만큼 반복

```
for 변수 in 리스트:
    문장(또는 블록)
```

  - 리스트 원소들의 순서대로 반복하면서 문장(또는 블록) 반복 수행

```
for i in [1, 2, 3, 4, 5]:
    print('파이썬')
```

  - 변수 i의 값을 반복되는 부분에서 사용하여 출력 가능

```
for i in [1, 2, 3, 4, 5]:
    print('파이썬', i)
```

실행결과

```
파이썬 1
파이썬 2
파이썬 3
파이썬 4
파이썬 5
```

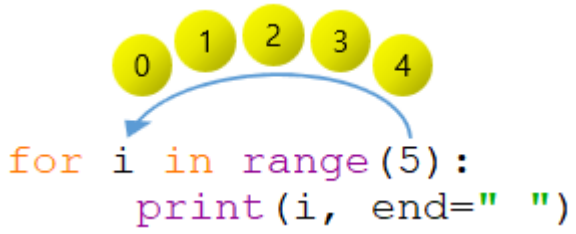
횟수만큼 반복 (range() 함수 이용)

- Range() 함수
  - 정수들을 생성하는 함수 `range(start=0, stop, step=1)`
  - start에서 시작하여 (stop - step)까지 step 간격으로 정수들 생성
  - 만약 start와 step이 생략되어 호출되면 start는 0, step은 1로 간주됨
  - (예) range(5)는 start와 step은 생략된 것이며, range(0, 5, 1)와 같음. 0부터 시작하여 (stop(5) - step(1)) 까지인 [0, 1, 2, 3, 4] 생성
  - (예) 1부터 5까지 생성할 경우 range(1, 6, 1) 또는 range(1, 6)으로 호출
- range() 함수를 이용한 for 문
  - range() 함수를 이용한 for 문 형태

```
for 변수 in range():
    문장(또는 블록)
```

  - range() 함수의 결과에 따라 순서대로 문장(또는 블록)을 반복 수행
  - range(5)는 range(0, 5, 1)로 호출한 것과 같고, 함수의 결과는 (stop - 1)까지 0, 1, 2, 3, 4가 생성되어 5회 반복됨

```
for i in range(5):
    print(i, end=" ")
```



  - print() 함수의 end=" "에 의해 출력 결과가 줄이 바뀌지 않고 한 줄에 출력됨

for 문 실습

- 리스트/range()를 이용하여 1부터 5까지 정수의 합계를 구하시오.

```
s=0
for i in [1, 2, 3, 4, 5]:
    s = s + i
print('s :', s)
```

```
s=0
for i in [1, 2, 3, 4, 5]:
    s = s + i
    print('i :', i, ', s :', s)
print('s :', s)
```

```
s=0
for i in [5, 4, 3, 2, 1]:
    s = s + i
print('s :', s)
```

```
s=0
for i in range(1, 6):
    s = s + i
    print('i :', i, ', s :', s)
print('s :', s)
```

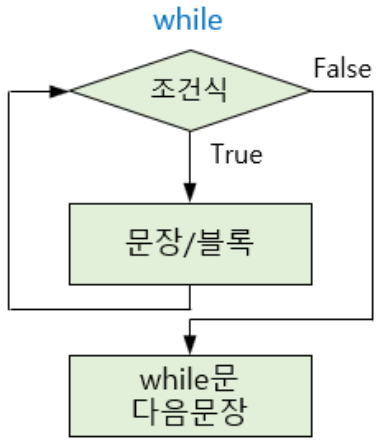
```
# range(1, 6, 1):
# range(5)
# range(6)
```

```
s=0
for i in range(1, 11, 2):
# for i in range(9, -1, -2):
    s = s + i
    print('i :', i, ', s :', s)
print('s :', s)
```

조건에 따른 반복 (while 문)

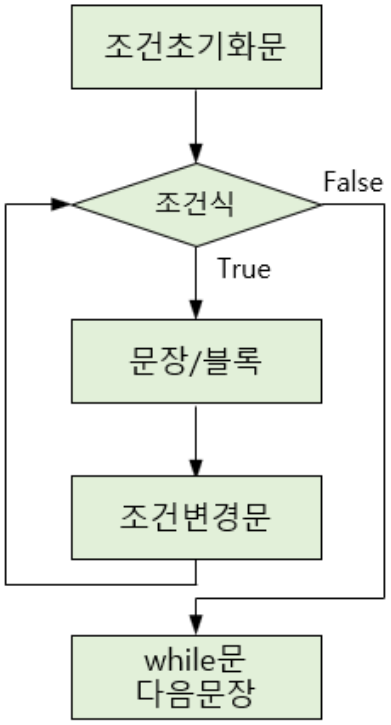
- 조건식의 값이 참인 경우 문장(또는 블록)을 반복 수행
- 조건식의 값이 거짓이면 반복을 종료하고 루프에서 빠져나감
- 초기 조건 값, 조건 비교, 조건 값 변경
  - while 문에 진입하기 전 초기 조건 값 설정
  - while 문에 진입하면서 조건 비교
  - while 루프 내에서 다음 번 반복을 위한 조건 값 변경

```
while 조건식:
    문장(또는 블록)
```

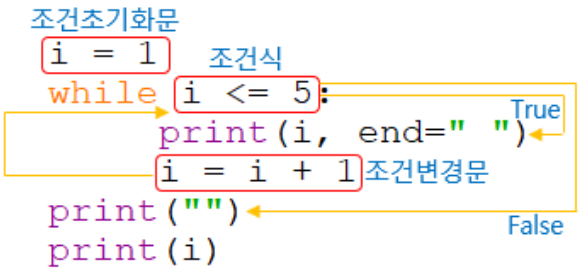


- **루프 제어 변수**
  - 초기 변수의 값 설정
  - 설정된 변수 값을 이용하여 조건 비교
  - 변수의 값을 변경한 후 다시 조건식 비교
  - 변수를 통해 반복문을 제어함
  - 반복문의 조건 비교 변수를 루프 제어 변수라 함

```
조건초기화문
while 조건식:
    문장(또는 블록)
    조건변경문
```



- 1부터 5보다 작거나 같을 때까지 반복하면서 값 출력




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

while문 실습

- 1부터 시작하여 루프 제어변수 n값이 10보다 작거나 같을 때까지 반복하면서 숫자가 짝수인 경우의 합계를 구함 (단, 루프 제어 변수n은 1씩 증가함)

```
n = 1
s  = 0
while n <= 10:
    if n%2 == 0:
        s = s + n
        print('n :', n, ', s :', s)
    n = n + 1
print('s :', s)
```

```
# while n >= 10:

# print('n :', n, ', s :', s)
```

```
n = 2
s  = 0
while n <= 10:
    s = s + n
    print('n :', n, ', s :', s)
    n = n + 2
print('s :', s)
```

- while문을 이용하여 9부터 1까지 반복하면서 홀수의 합을 구함 (단, 루프 제어 변수n은 -2씩 변화함)

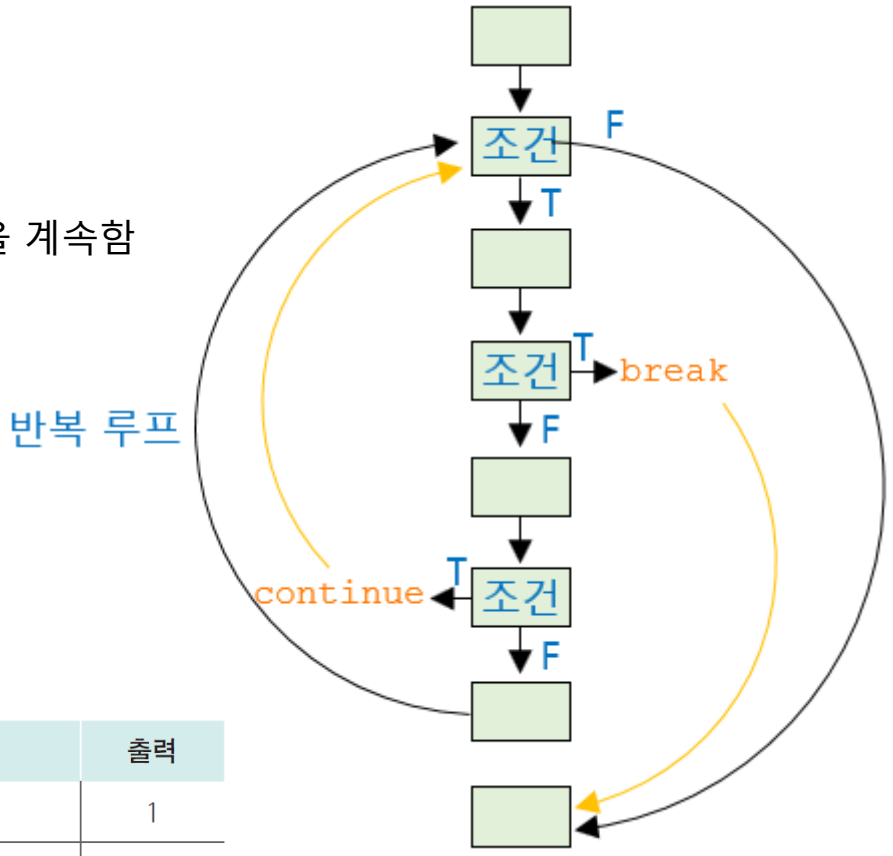
```
n = 9
s  = 0
while n >= 1:
    s = s + n
    print('n :', n, ', s :', s)
    n = n - 2
print('s :', s)
```

루프(Loop) 제어 (루프 탈출과 계속 반복)

- break 문
  - 반복문의 루프를 빠져 나옴
- continue 문
  - continue문 아래에 있는 문장들은 건너뛰고 다음 반복을 계속함

```
for i in [1, 2, 3, 4, 5]:
    if i == 4:
        break
    if i % 2 == 0:
        continue
    print(i, end=" ")
print("")
print(i)
```

for 반복

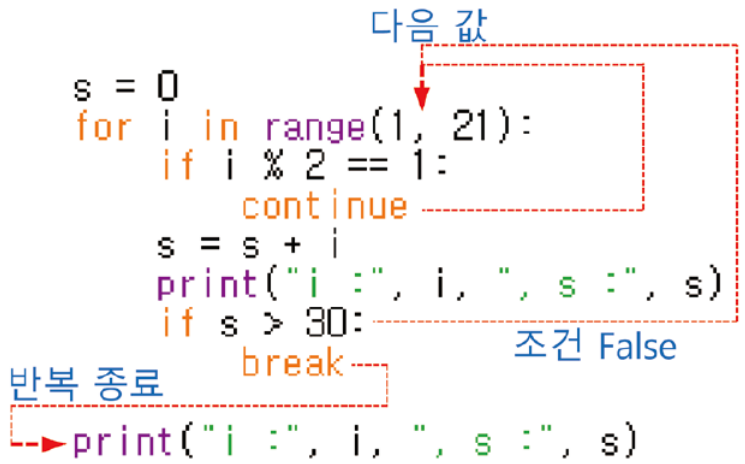


반복순서	변수 i 값	if i == 4	if i % 2 == 0	출력
반복 1	1	False	False	1
반복 2	2	False	True (continue 실행)	
반복 3	3	False	False	3
반복 4	4	True (break 실행)		
for 문 종료	4			4

루프 제어문 실습

- for 문을 이용하여 1부터 20까지 반복하면서 홀수인 경우 continue를 사용하여 반복을 계속 진행하고, 짝수인 경우만 합계를 구함. 단, 합계가 30을 넘을 경우, 반복을 종료함. 반복을 진행하면서 반복 진행 횟수와 합계를 계속 표시하고, 반복이 종료된 후에도 표시함.

```
s = 0
for i in range(1, 21):
    if i%2 == 1:
        continue
    s = s + i
    print('i :', i, ', s :', s)
    if s > 30:
        break
print('i :', i, ', s :', s)
```



- for 문을 이용하여 1부터 20까지 반복하면서 짝수인 경우 continue를 사용하여 반복을 계속 진행하고, 홀수인 경우만 합계를 구함. 단, 합계가 30을 넘을 경우, 반복을 종료함. 반복을 진행하면서 반복 진행 횟수와 합계를 계속 표시하고, 반복이 종료된 후에도 표시함.

```
if i%2 == 0:
```

EOD