반복문 응용:

중첩 반복문

조건문+반복문

기타 제어문

(break / continue)

중첩 반복문

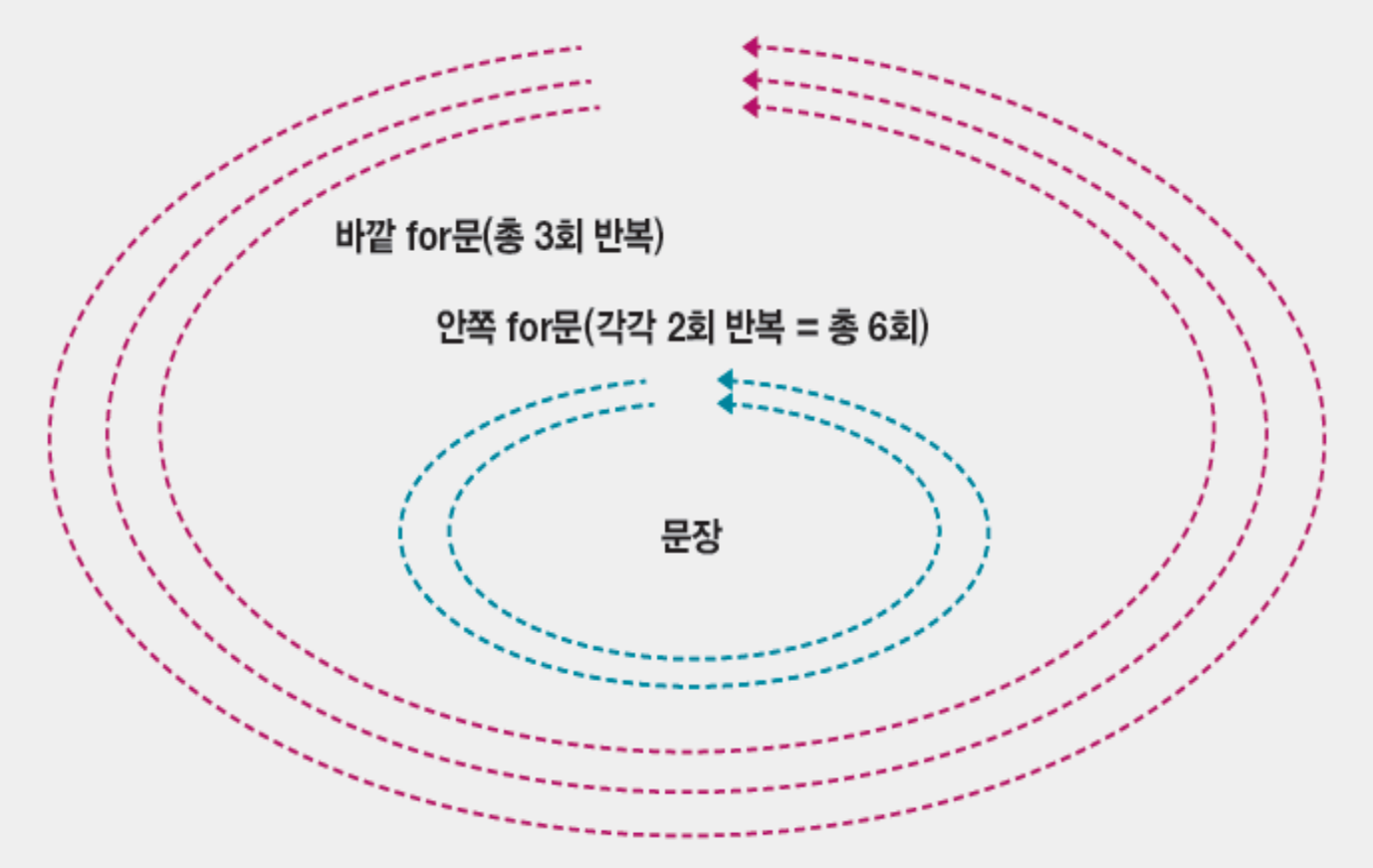
- 조건문과 마찬가지로 반복문 또한 중첩 가능
  - 하나의 **for**문 혹은 하나의 **while**문으로 해결되지 않는 문제 발생 시 중첩된 반복문 사용

for item1 in sequence1:
  True_statements1
  for item2 in sequence2:
  True_statements2
	sequence1 내의 item1 각각에 대해서 True_statements1과 다음 for문을 반복적으로 수행
	sequence2 내의 item2 각각에 대해서 True_statements2를 반복적으로 수행

while 조건1:
  True_statements1
  while 조건2:
  True_statements2
	조건1을 만족할 경우 True_statements1과 다음 while문을 반복적으로 수행
	조건2를 만족할 경우 True_statements2를 반복적으로 수행

중첩 for문

- 실행 횟수 = 바깥 for문 반복 횟수 × 안쪽 for문 반복 횟수



중첩 for문 예제1

```
for i in range (0, 3, 1) :
    for k in range(0, 2, 1) :
            print("파이썬은 꿀잼입니다. ^^ (i값: %d, k값: %d)" % ( i, k ))
```

출력 결과

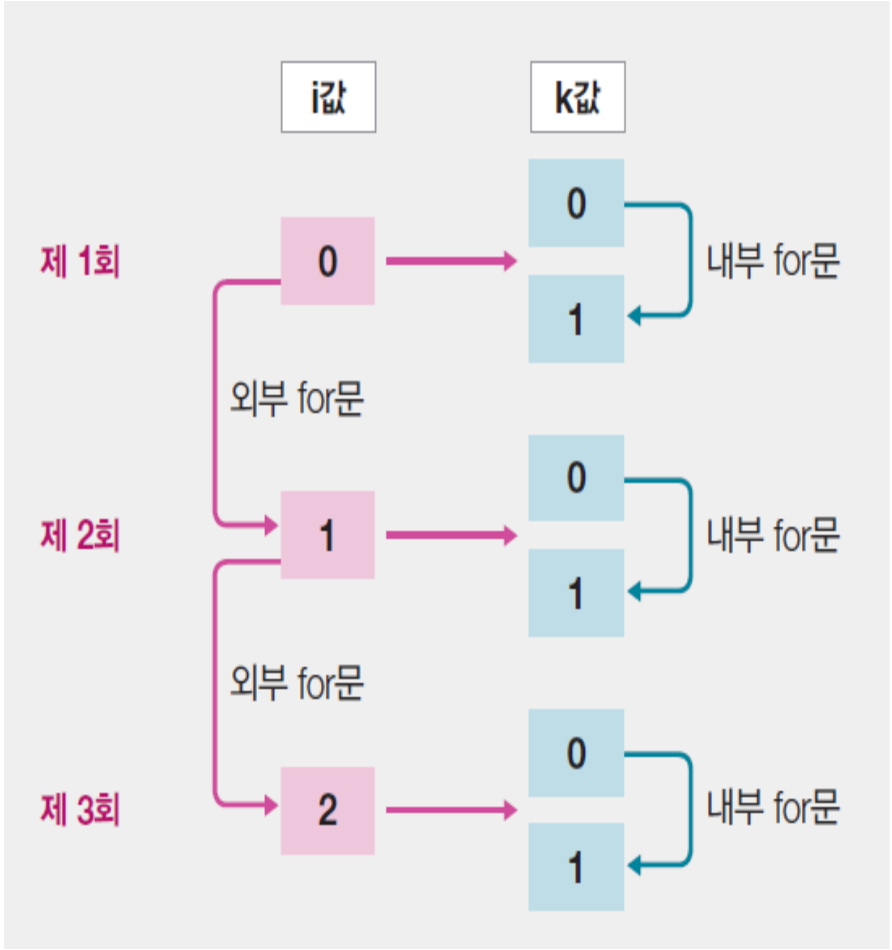
```
파이썬은 꿀잼입니다. ^^ (i값: 0, k값: 0)
파이썬은 꿀잼입니다. ^^ (i값: 0, k값: 1)
파이썬은 꿀잼입니다. ^^ (i값: 1, k값: 0)
파이썬은 꿀잼입니다. ^^ (i값: 1, k값: 1)
파이썬은 꿀잼입니다. ^^ (i값: 2, k값: 0)
파이썬은 꿀잼입니다. ^^ (i값: 2, k값: 1)
```

중첩 for문 예제1 *(cont.)*

(1) 외부 for문 1회: i에 0을 대입
    내부 for문 1회: k에 0을 대입한 후에 print() 수행
    내부 for문 2회: k에 1을 대입한 후에 print() 수행

(2) 외부 for문 2회: i에 1을 대입
    내부 for문 1회: k에 0을 대입한 후에 print() 수행
    내부 for문 2회: k에 1을 대입한 후에 print() 수행

(2) 외부 for문 3회: i에 2를 대입
    내부 for문 1회: k에 0을 대입한 후에 print() 수행
    내부 for문 2회: k에 1을 대입한 후에 print() 수행



중첩 for문 예제2

- 구구단 2단부터 9단까지 출력


```
for i in range(2, 10, 1) :
    for k in range(1, 10, 1) :
        print(" %d X %d = %2d" % (i, k, i*k))
    print("")
```

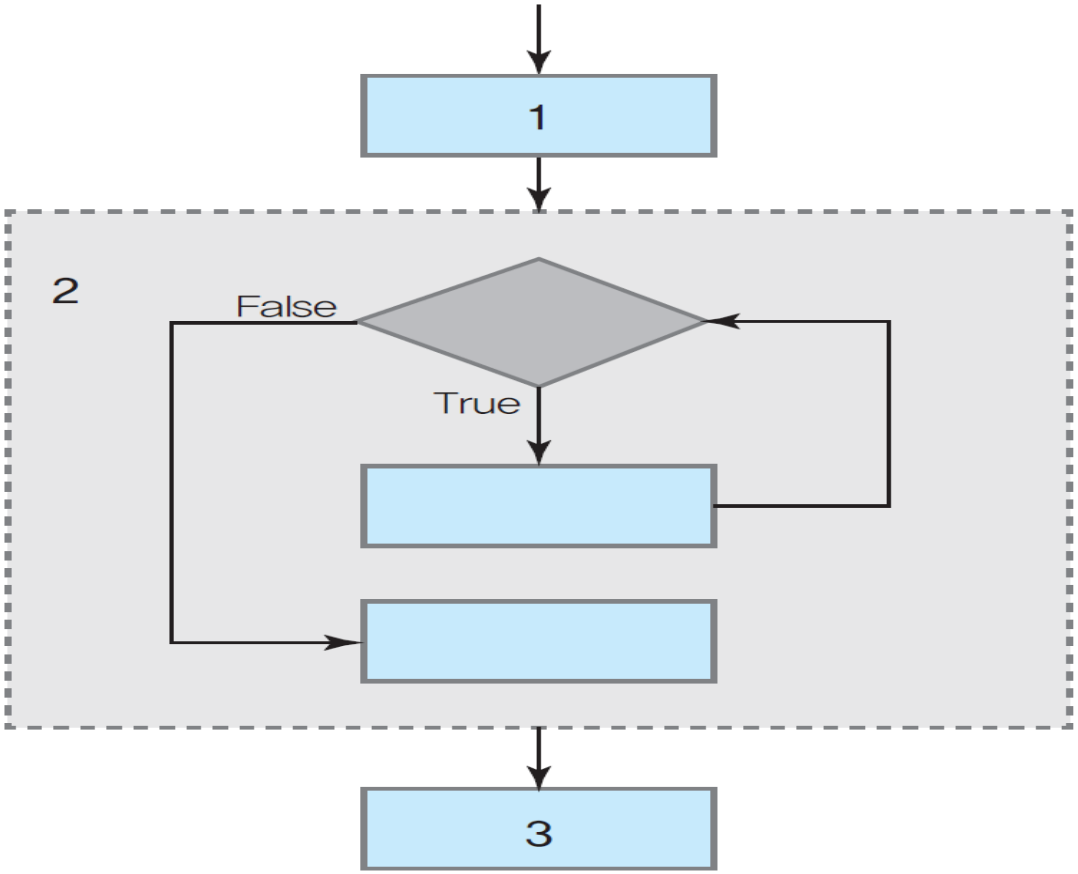
중첩 for문 예제2 (cont.)

- 프로그램 수정
  - 기존 구구단 결과는 세로로 출력되므로 결과를 보려면 스크롤을 움직여야 함
  - 오른쪽 여백에 구구단 결과를 보여주기 위해 가로 먼저 출력


```
for i in range(1, 10) :
    guguLine=""
    for k in range(2, 10) :
        guguLine = guguLine + str("%2dX%2d=%2d" % (k, i, k*i))
    print(guguLine)
```

알고리즘의 구조: 조건 구조 + 반복 구조

- 조건 구조와 반복 구조를 포함하는 순차 구조



while문 + if문 / for문 + if문

- *30 이하의 정수를 입력하면 입력한 정수보다 큰 수부터 30까지의 모든 짝수를 출력하는 프로그램*

```
n = int(input('Enter the even number (1~30) : '))

while n < 30:
      n = n + 1
      if n % 2 == 0:
         print(n)
```

- *10부터 20 사이의 홀수를 출력하는 프로그램*

```
for i in range(10, 21, 1):
    if i%2 != 0:
       print(i, "is a odd number")
```

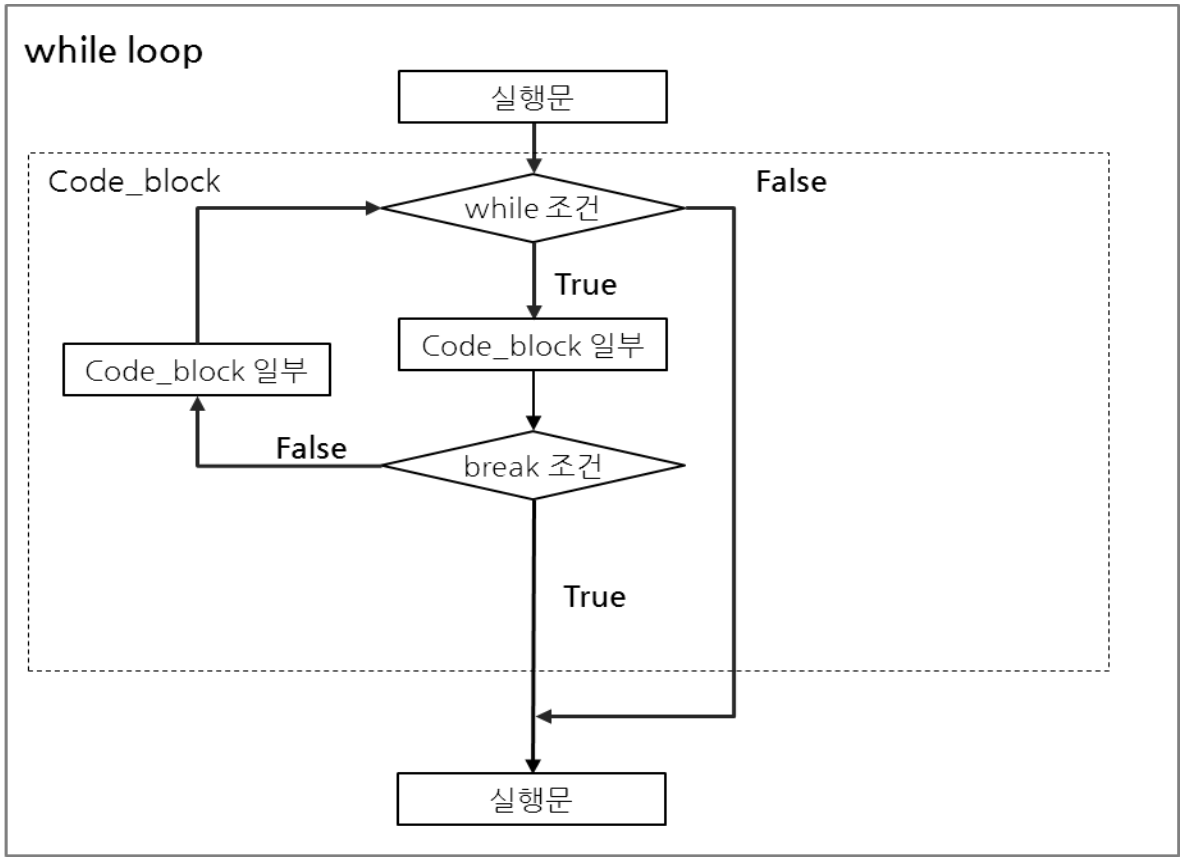
기타 제어문: break / continue

- break
  - 사용자가 원하는 시점에서 강제적으로 Code_block의 반복을 강제로 중단

- continue
  - 무조건 블록의 남은 부분을 건너뛰고 반복문의 처음으로 돌아감

while문의 break 사용

while 조건:
  Code_block
  (break) 조건:
  break
	조건에 만족하는 동안 Code_block을 반복적으로 수행하고,
	break 조건에 만족하면 반복을 중단하고 while문을 빠져나감



while문의 무한루프와 break 사용

- 무한 루프
  - while문의 조건식이 항상 참(True)이여서 반복문이 끝나지 않는 경우
  - *cf) 무한 루프를 중지하려면 Ctrl + C 를 누름*
  - *예제)*

사용자가 입력한 정수 n 이 홀수인지 짝수인지를 판별하는 프로그램을 만들고자 한다.
단, 0값을 입력하면, 프로그램을 종료한다.

```
while(True) :
    n = int(input('enter the number : '))
    if n == 0 :
        print('EXIT')
        break
    elif n%2 == 0 :
        print(n, 'is even number')
    else :
        print(n, 'is odd number')
```

for문의 break 사용 예제

- *1~100까지 더하되, 누적 합계(hap)가 1000 이상이 되는 시작 지점을 구하는 프로그램*

```
hap = 0

for i in range(1,101) :
    hap += i

    if hap >= 1000 :
        break

print("1~100의 합에서 최초로 1000이 넘는 위치 : %d" % i)
```

for문의 continue 사용 예제

- *0~9까지 짝수를 출력하는 프로그램*

```
for i in range(10):
    if i % 2 == 1:
        continue
    print(i)
```

- *1~100까지의 합을 구하되 1 +2 +4 +5 +7 +8 +10 +...과 같이 3의 배수를 제외하고 더하는 아래의 프로그램을 continue를 사용하여 구현하시오.*

```
hap = 0

for i in range(1,101) :
    if i % 3 != 0 :
        hap += i

print("1~100의 합계(3의 배수 제외): %d" % hap)
```