Chapter. 03

파이썬 프로그래밍 언어

I 흐름 제어: 조건문과 반복문

FASTCAMPUS
ONLINE
금융공학/퀀트I

강사. 서찬웅

I 이번시간에 배울 내용

- 조건문 - if 문
- 반복문 - for 문
- continue, break
- range()함수
- 반복문 - while 문

I 조건문

어느 아내가 프로그래머 남편에게

아내 : 우유 하나 사와. 아, 달걀 있으면 6개 사와

남편은 우유를 6개 사왔다.

아내는 물었다.

아내 : 왜 우유를 6개나 사왔어!

남편: 달걀이 있길래 6개 사왔지

남편의 생각

IF 달걀 있는가?

우유 6개를 산다.

ELSE 달걀이 없다면 :

우유 하나 사온다.

I 조건문

- if문은 조건문에서 조건식이 True인지 False인지를 판단하여 문장을 실행하는데 조건식은 우리가 앞에서 배운 비교 연산자, 논리 연산자의 결과 값을 받아서 사용합니다.
  - ➢ if(만약) 조건이 참이라면, if 블록의 명령문을 실행
    else(아니면) else블록의 명령문을 실1행
  - ➢ 이 때 else 조건절은 생략이 가능

✓ 문법

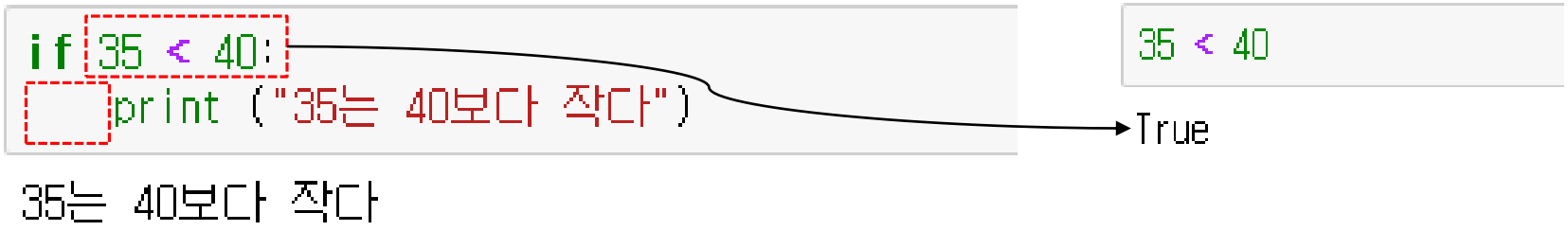
```
if 조건식1:
    statement1
elif 조건식2:
    statement2
else:
    statement3
```

조건식으로 끝난 문장은 : 를 붙여야 한다.

elif → else if의 줄인 말

I 조건문

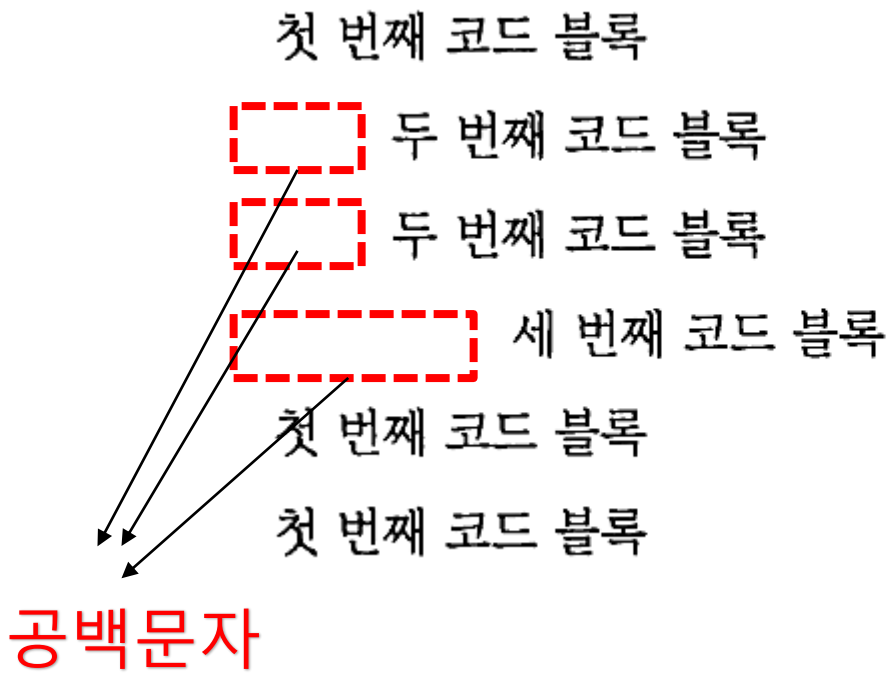
- if문의 예제를 보면서 자세히 알아보겠습니다.



- 35<40의 표현식만 본다면 오른쪽 그림처럼 True 입니다.
- True 값이기 때문에 if 구문의 아래 표현식 코드가 실행이 되었습니다.
- print 함수 앞에 공백을 우리는 앞에서 배운 들여쓰기(indent block)라고 부릅니다.
- 파이썬에서는 같은 깊이만큼(공백 또는 탭) 있는 표현식은 모두 같은 코드 블록에 있다고 인식합니다.

I 조건문

- 일반적으로 4 개의 공백 문자를 사용하지만 강제는 아닙니다.
- 이전 설명처럼 프로그래머가 원한다면 변경할 수 있습니다.
- 단 모든 들여쓰기를 일관되게 작성해야 합니다.
- 공백 2 개를 들여쓰기로 사용했다면 같은 코드 블록은 공백 2 개로 들여쓰기 되어 있어야 합니다.



I 조건문

- 아래 예제의 결과를 예상해 보세요.

```
speed = 120
if speed >= 110:
    print ("과속입니다.")
print ("안전운행")
```

- if 문은 위의 예제처럼 독립적으로 사용될 수 있지만, else 문과 함께 사용할 수 있습니다.
- else 아래 들여쓰기로 써진 코드는 if문이 거짓일 때(False) 실행이 됩니다.
- else문에도 : 가 붙어 있는걸 유의하세요. : 다음 줄은 들여쓰기가 시작되고 하위 코드들입니다.

```
speed = 100
if speed >= 110:
    print ("과속입니다.")
else:
    print ("정상속도입니다.")
print ("안전운행")
```

I 조건문

- else 구문을 할때 다시 if 조건을 줄 수 있습니다.
- 사람의 언어로 표현하면 '아니라면 그럼 다른 조건을 제시할께 " 정도 될거 같습니다.
- elif는 else if를 줄인 단어로 역시 조건문이 True일 때 실행할 코드는 : 다음 줄, 즉 들여쓰기가 시작되는 블록

```
speed = 120
if 130 <= speed:
    print ("130km 과속입니다.")
elif 120 <= speed:
    print ("120km 과속입니다.")
elif 110 <= speed:
    print ("110km 과속입니다.")
else:
    print ("정상속도입니다.")
print ("안전운행")
```

- else 아래 코드는 최종적으로 모든 조건이 False일 때 실행됩니다.
- 만약 else 위에 조건문들중 참이 실행이 되면 해당 코드블록만 실행되고 조건문은 종료가 됩니다.

  곧, else 표현식은 실행이 되지 않습니다.

I 조건문

- if 문을 사용해서 다음 문장을 코드로 표현하고 실행하세요.

- 문제

  Distance라는 변수에는 상수 값이 들어갈 예정입니다.

  이 distance가 3km이하로 작다면 “걸어가세요” 라고 말하고

  3km보다 멀고 10km보다 작다면 “버스타세요”라고 말해야 합니다.

  만약 10km이상 간다면 “택시 타세요”라고 답할수 있는 조건문을 만들어보세요.

  그리고 조건문과 상관없이 마지막에 “수고하셨습니다.”라고 출력하세요.

I 반복문 – for 문

- for 문을 반복문이라고 합니다. 가장 많이 사용하며, 형식은 아래와 같습니다.

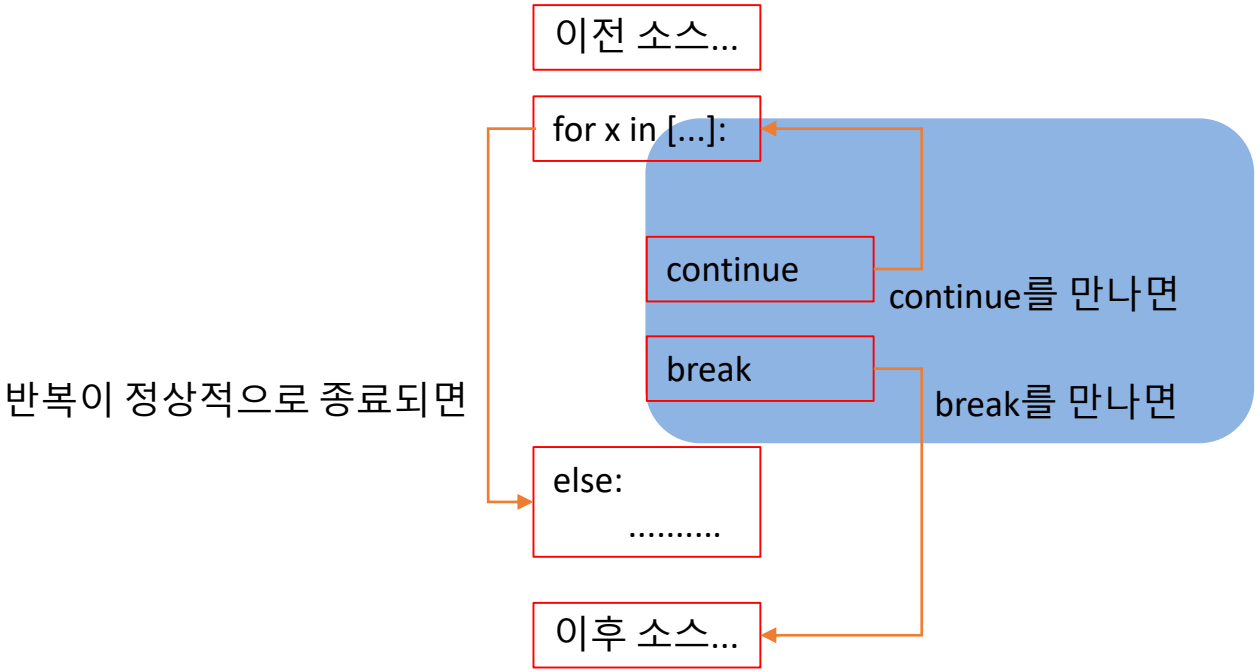
```
for <target> in <object> :
    <statement1>
else:
    <statement2>
```

조건문 뒤에는 : 를 잊지 말자

- <object>는 시퀀스형 데이터가 들어가야 합니다.
- <object>의 각 항목은 <target>에 치환되어 <statment1> 이나 <statement2>를 실행합니다.
- 반복의 횟수는 <object>의 크기가 즉 시퀀스의 크기가 됩니다.
- for 문에서 사용할 수 있는 객체는 아래와 같습니다.
  - 문자열, 리스트, 튜플, 사전, range() , 반복이 가능한 객체

I 반복문 – for 문 – continue, break

- for 문의 내부 블록에서 continue를 만나면 시작 부분으로 이동, break를 만나면 for문의 내부 블록을 종료합니다.
- break, continue 표현식은 if 문과 함께 표현합니다.



I range() 함수

- 내장 함수인 range() 함수는 숫자로 이루어진 값을 만듭니다.
- for 문과 같이 사용합니다. range() 함수는 한 개에서 세 개까지의 인수를 받을 수 있습니다.
- range(x)와 같이 인수가 주어져 있을 경우, 0부터 인자값 x보다 작은 수까지의 리스트를 1간격으로 생성
- range(x, y)인 경우 x부터 y보다 작은 값(y-1)으로 데이터를 시퀀스하게 만듭니다.
- range(x, y, s)인 경우 range(x, y)와 같지만 간격을 s로 하여 값을 생성을 합니다.

```
for number in range(3):
    print (number)
```

```
0
1
2
```

```
for number in range(1,3):
    print (number)
```

```
1
2
```

```
for number in range(1,10,2):
    print (number)
```

```
1
3
5
7
9
```

I 반복문 – for 문 – continue, break

- 아래 예제를 보면서 결과를 예측해 봅시다.
- if 조건을 추가하여 조건식이 참이면 break, 혹은 continue 합니다.
- for문에서 else문은 for 문 블록이 강제로 종료(break) 되지 않아야 실행이 됩니다.

```
for i in range(1,13):
    if i % 2 == 0 and i % 3 == 0:
        print (i)
        break
        print ('if statement')

else:
    print ('complete')
```

```
for i in range(1,13):
    if i % 2 == 0 and i % 3 == 0:
        print (i)
        continue
        print ('if statement')

else:
    print ('complete')
```

I 문제

- 아래 for문은 1부터 100까지 정수를 출력하는 코드입니다.
- 아래 조건을 추가하여 만들어 보세요.

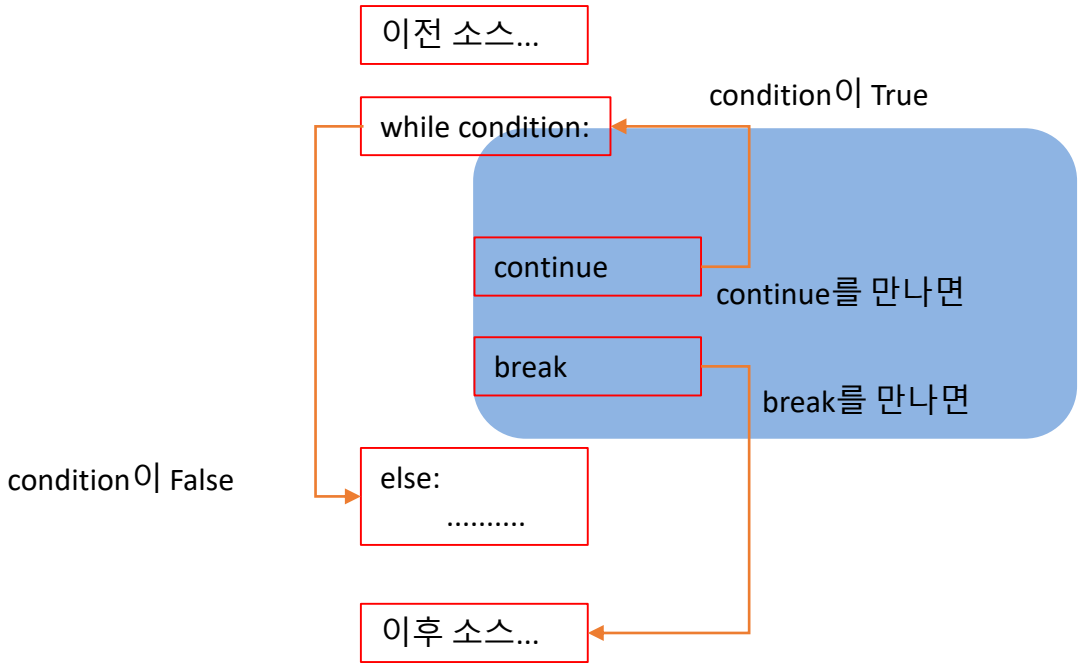
```
for x in range(1, 101):
    print (x)
```

- 조건 : 2의 배수를 출력해 주세요. 단 5의 배수 일때는 출력을 하지 않습니다.

I while문

- for 문보다 일반적인 반복문입니다.
- 시작 부분의 <condition>을 검사해서 결과가 True 이면 내부 블록이 반복적으로 실행됩니다.
- else 블록은 <condition>이 False일 때 while문을 빠져나올 때 실행됩니다.
- break로 빠져나올 때는 else 블록은 실행되지 않습니다.

```
while <condition>:
    <statement>
else:
    <statement>
```



I while문

```
In [125]: a = 0
     ...: while a < 10:
     ...:     a = a + 1
     ...:     if a < 3:
     ...:         print (a)
     ...:         continue
     ...:         print ('continue')
     ...:     if a > 10:
     ...:         break
     ...:
     ...: else:
     ...:     print ('else block')
     ...: print ('done')
1
2
else block
done
```

a에 +1 하여 다시 a에 대입

a가 3보다 작을때 실행 print (a)가 출력되고 continue를 만나 처음으로 이동

a가 10보다 크지 못하기 때문에 if문은 항상 False break문은 실행되지 못함

정상적으로 종료되었기 때문에 else 구문이 실행

while문과 같은 블록 위치를 같기 때문에 while문과 독립적으로 print('done') 실행

```
In [128]: a = 0
     ...: while a < 10:
     ...:     a = a + 1
     ...:     if a < 3:
     ...:         print (a)
     ...:         continue
     ...:         print ('continue')
     ...:     if a > 6:
     ...:         break
     ...:
     ...: else:
     ...:     print ('else block')
     ...: print ('done')
1
2
done
```

a가 6보다 클때 True가 되기 때문에 break문 실행 while 종료

정상적으로 종료가 되지 않았기 때문에 else는 실행되지 않는다.

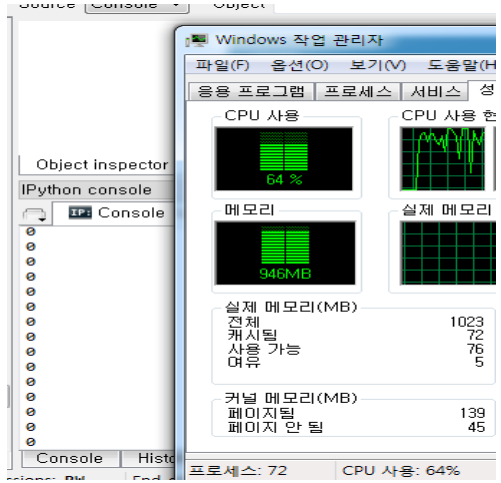
I while문

- 참고
  - effective Python 책에서는.... 아래와 같이 권고합니다.
    - 반복문 뒤에 else 블록을 사용하면 직관적이지 않고 혼동하기 쉬우니 사용하지 말 것.

- for 문과 while 문의 차이점
  - for문은 조건 및 집합의 순환을 위해서 반복 할 수 있습니다.
  - while 문은 하고 싶은 어떤 종류의 반복적인 구성도 할 수 있습니다.
  - 하지만 while문은 무한 루프를 일으킬 수 있기 때문에 보통은 for 문으로 많은 일을 하게 됩니다.

```
In [1]: i = 0

In [2]: while i < 5:
   ...:     print i
   ...:     i = i + 1
   ...:
0
1
2
3
4
```



I 실습

- 윤년(閏年)은 역법을 실제 태양년에 맞추기 위해 여분의 하루 또는 월(月)을 끼우는 해이다. 태양년은 정수의 하루로 나누어떨어지지 않고, 달의 공전주기와 지구의 공전주기는 다르기 때문에 태양력에서는 하루(윤일), 태음태양력에서는 한 달(윤달)을 적절한 시기에 끼워서 이를 보정한다. - Wikipedia
- 윤년의 규칙
  - 1. 연수가 4로 나누어 떨어지는 해는 윤년으로 한다.
  - (2004, 2008, 2012, 2016, 2020)
  - 2. 이 중에서 100으로 나누어 떨어지는 해는 평년으로 한다.
  - (1900, 2100, 2200, 2300)
  - 3. 그 중에서 400으로 나누어 떨어지는 해는 윤년으로 둔다.
  - 위의 3가지 규칙을 사용하여 1900년부터 2100년까지 윤년을 출력하시요.

I 정리

- if 문을 사용한 제어문에 대한 학습
- for 문과 while 문을 사용하여 반복문을 학습
- for 문과 while 문의 차이
- range() 함수를 사용한 데이터 생성
- continue, break 문을 통한 반복문 제어

감사합니다

FAST CAMPUS
ONLINE
서찬웅 강사.