

파이썬 프로그래밍 입문

반복문

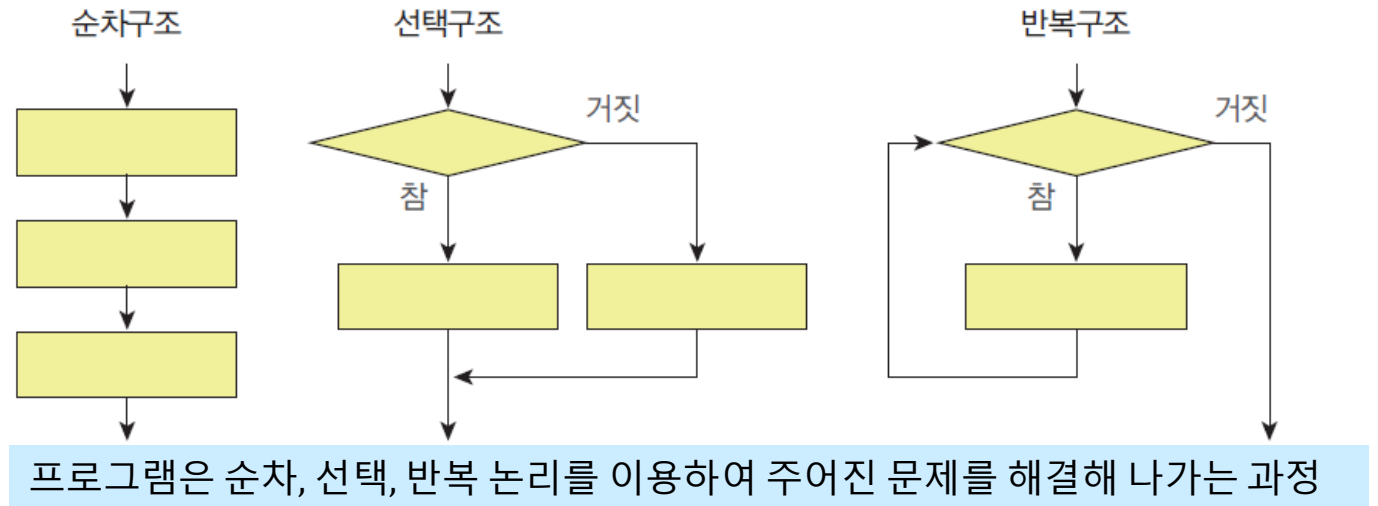
최 윤 정

cris.lecture@gmail.com

오늘은

□ 반복문 대해 알아봅니다.

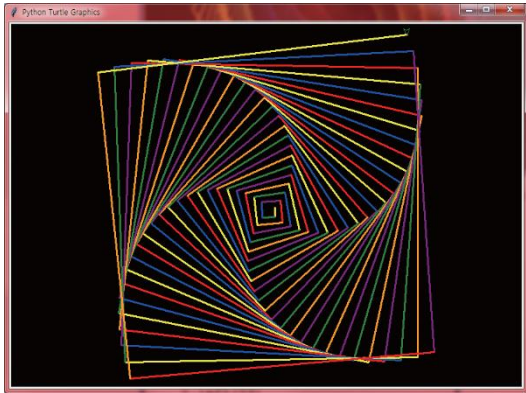
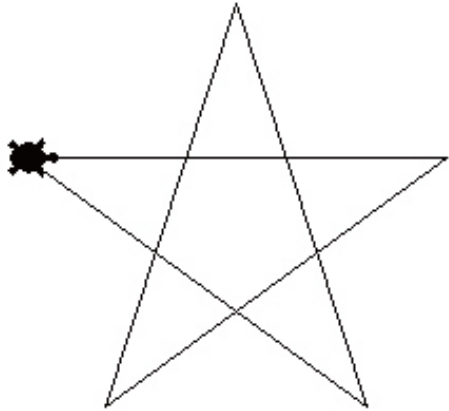
- 같은 라인이나 블록을 여러 번 반복하기
- for 와 while 을 사용하여
 - 지정한 횟수만큼
 - 조건만큼
 - 무한루프
- 중첩구조 : 반복문안에 또 반복문



□ 실습

- 구구단
- 로그인하기-횟수제한
- 사각형과 삼각형 그리기

실습내용 미리보기



```
PS D:\python코드> & C:/Users/user/App
가로길이: 5
세로길이: 10
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■
■■■■■

PS D:\python코드> & C:/Users/user/App
가로길이: 10
세로길이: 7
■■■■■■■■
■■■■■■■■
■■■■■■■■
■■■■■■■■
■■■■■■■■
■■■■■■■■
■■■■■■■■
```

```
===
0 :
1 : 0
2 : 0 1
3 : 0 1 2
4 : 0 1 2 3
5 : 0 1 2 3 4
6 : 0 1 2 3 4 5
7 : 0 1 2 3 4 5 6
8 : 0 1 2 3 4 5 6 7
9 : 0 1 2 3 4 5 6 7 8
```

```
0 :
1 : ▲
2 : ▲▲
3 : ▲▲▲
4 : ▲▲▲▲
5 : ▲▲▲▲▲
6 : ▲▲▲▲▲▲
7 : ▲▲▲▲▲▲▲
8 : ▲▲▲▲▲▲▲▲
9 : ▲▲▲▲▲▲▲▲▲
>>>
```

반복이란?

- 반복(iteration)은 동일한 문장을 여러 번 반복시키는 구조
- 컴퓨터는 인간과 다르게 반복적인 작업을 실수 없이 빠르게 할 수 있다.
- 가장 큰 장점!!



왜 반복이 중요한가?

- 전광판에 '방문을 환영합니다!'를 5번 출력한다고 하자.



`print("방문을 환영합니다!")` 를 여러 번 쓰면 되겠군!

왜 반복이 중요한가?

□ 5번 째야!!

```
print("방문을 환영합니다!")  
print("방문을 환영합니다!")  
print("방문을 환영합니다!")  
print("방문을 환영합니다!")  
print("방문을 환영합니다!")
```

복사해서 붙여넣기!!

만약 1000번 반복해야 한다면?

```
print("방문을 환영합니다!")  
print("방문을 환영합니다!")  
print("방문을 환영합니다!")  
print("방문을 환영합니다!")  
print("방문을 환영합니다!")  
...  
...  
print("방문을 환영합니다!")
```

1000번 복사해서 붙여넣는다구??
이건 좀...



만약 1000번 반복해야 한다면?

□ 반복 구조를 사용한다!

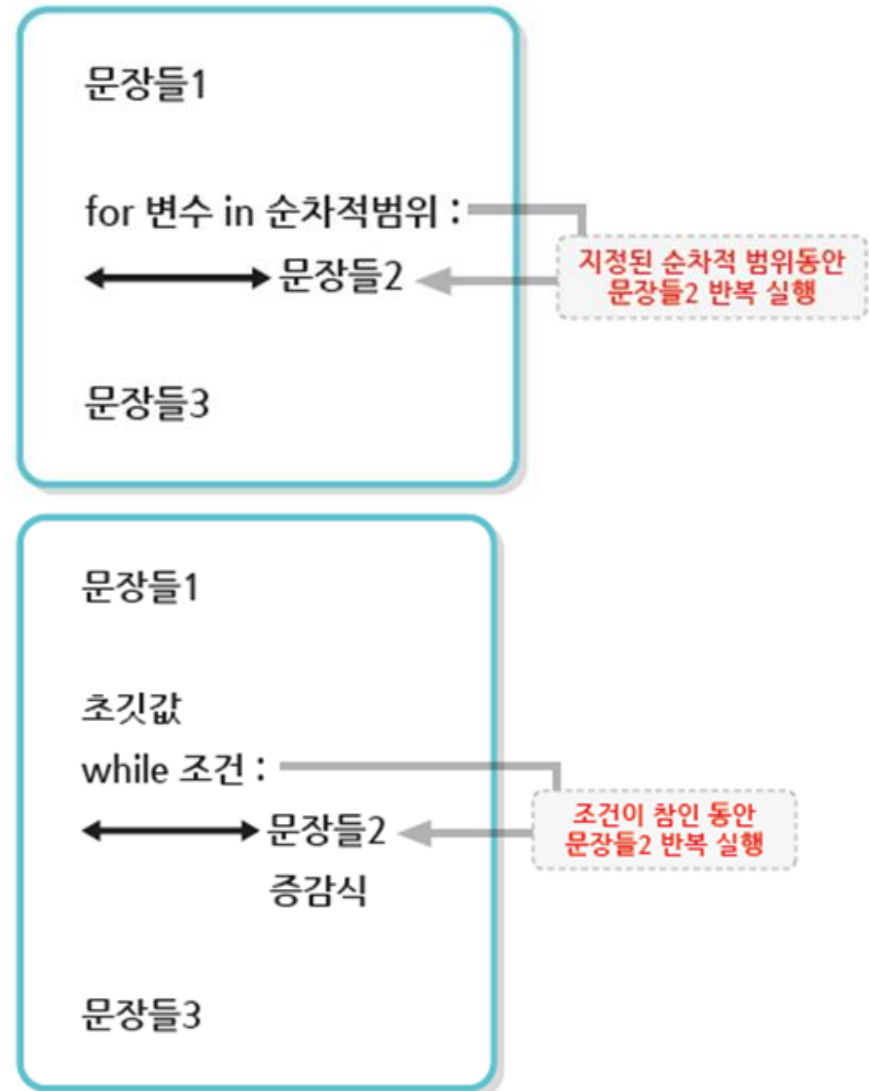
- 횟수로 제한하는 for문
- 조건으로 제한하는 while문

#i=0 변수는 초기화해두는 습관^^

```
for i in range(1000):
```

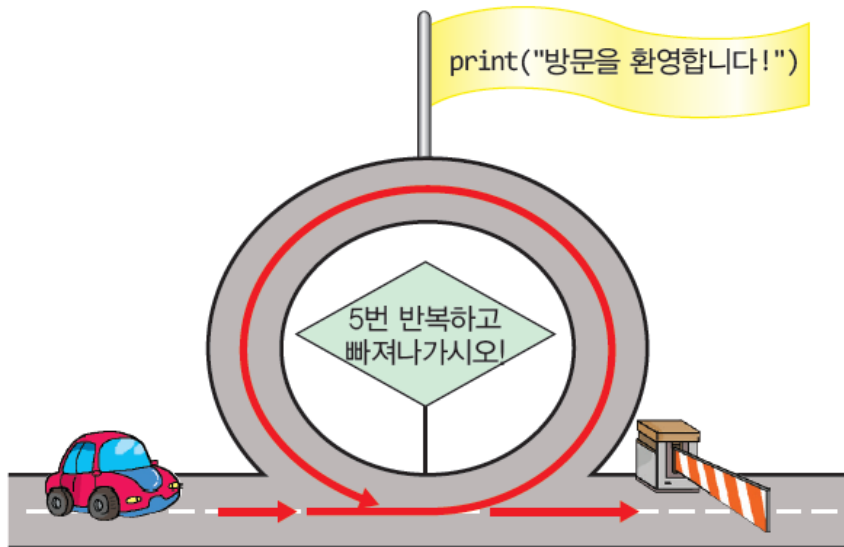
```
    print("방문을 환영합니다!")
```

0부터 999까지 1000번 반복시키는 구조



for 문

- 횟수로 반복논리를 표현하는 반복문으로 가장 많이 사용된다.
- range()함수를 이용하여 반복횟수를 지정한다.
- for-each : range() 대신 데이터 개수만큼 동작하게 하는 방법도 있다.



for 문

```
for 변수 in range( 종료 값 ) :  
    문장
```

0에서 (종료 값-1)까지의 숫자를 반환한다.

반복되는 문장으로 들여쓰기 하여야 한다.

for 문의 사용 예

#변수 i가 0부터 5미만의 수, 0,1,2,3,4까지 증가하면서 5회 반복한다.

```
for i in range(5):  
    print("방문을 환영합니다!")
```

#range 대신 리스트 표현을 이용하여 [] 에 범위(개수)를 줄 수도 있다.

#변수 i가 []안의 데이터인 1,2,3,4,5를 순서대로 방문하면서 5회 반복한다.

#[10,20,30,40,50]이어도 5회 반복한다.

```
for i in [1, 2, 3, 4, 5] :  
    print("방문을 환영합니다.")
```

방문을 환영합니다!
방문을 환영합니다!
방문을 환영합니다!
방문을 환영합니다!
방문을 환영합니다!

for 문 : range() 함수

□ range()를 이용하여 증감과 step을 표현한다.

- start값은 0, step값은 1로 설정되어 있어 생략가능하다. 별도로 명시할 경우에는 설정값이 사용된다.

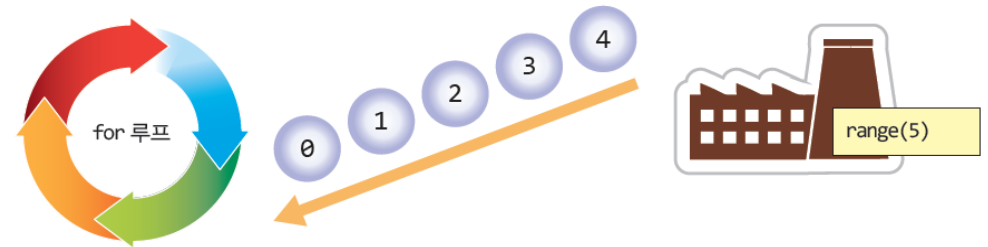
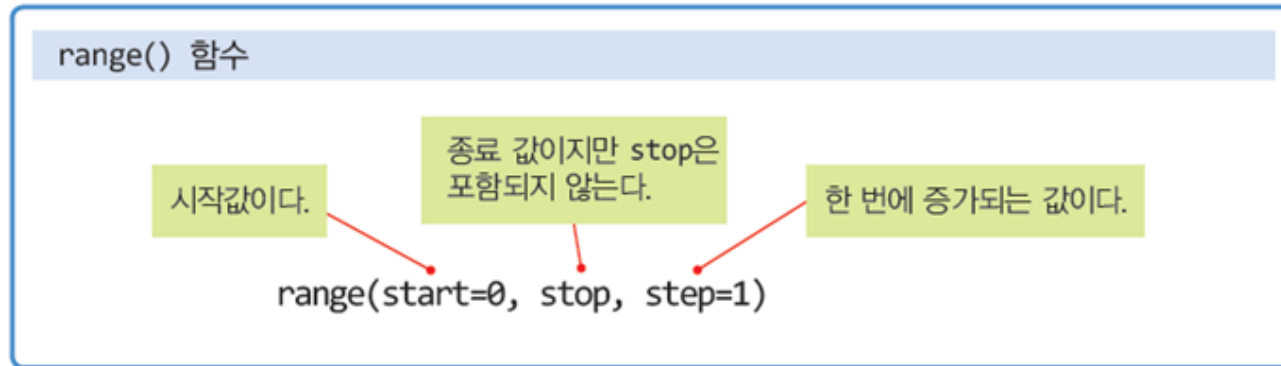


표 5-1 range() 함수의 개념사용

함수	설명
<code>range(x)</code>	0부터 x-1까지 정수의 순차적 범위
<code>range(x,y)</code>	x부터 y-1까지 정수의 순차적 범위
<code>range(x,y,z)</code>	x부터 y-1까지 z씩 증가하는 정수의 순차적 범위
<code>range(x,y,-z)</code>	x부터 y+1까지 z씩 감소하는 정수의 순차적 범위

for 문 : range() 함수

- 만약 1부터 시작하여서 10까지 홀수만 출력하고 싶다면?

1 3 5 7 9

```
for i in range(1, 10, 2):  
    print(i, end=" ")  
# print( i ) 한 줄에 하나씩 출력합니다.  
# end=""를 넣어 줄바꾸기를 하지 않도록 합니다
```

- 구구단 출력하기

출력할 단 : 3

3*1=3

3*2=6

....

3*9=27

```
dan=int(input("출력할 단 : ")) :  
for i in range(1, 10) :  
    print(f" {dan}*{i}={dan*i}", dan*i)  
#for i in [1, 2, 3, 4, 5,6,7,8,9] 도 가능
```

잠깐 Quiz

```
for i in range(1, 6, 1) :  
    print("i=", i)
```

```
i= 1  
i= 2  
i= 3  
i= 4  
i= 5
```

```
for i in range(5, 0, -1) :  
    print("i=", i)
```

```
i= 5  
i= 4  
i= 3  
i= 2  
i= 1
```

```
for i in range(1, 10, 2) :  
    print("i=", i)
```

```
i= 1  
i= 3  
i= 5  
i= 7  
i= 9
```

```
for i in range(10, 1, -2) :  
    print("i=", i)
```

```
i= 10  
i= 8  
i= 6  
i= 4  
i= 2
```

□ 출력할 때마다 엔터(한줄내리기)를 넣지않으려면? end 변수를 특정문자로 지정한다.

```
for i in range(1, 10, 2) :  
    print(i, end=" ")
```

```
1 3 5 7 9
```

```
for i in range(1, 10) :  
    print(i, end=" ")
```

```
1 2 3 4 5 6 7 8 9
```

```
for i in range(10, -1) :  
    print(i, end=" ")
```

```
출력결과없음. 왜일까?
```

```
for i in [1,10,100,1000] :  
    print(i, end="**")
```

잠깐 Quiz

```
for i in range(1, 6, 1) :  
    print("i=", i)
```

```
i= 1  
i= 2  
i= 3  
i= 4  
i= 5
```

```
for i in range(5, 0, -1) :  
    print("i=", i)
```

```
i= 5  
i= 4  
i= 3  
i= 2  
i= 1
```

```
for i in range(1, 10, 2) :  
    print("i=", i)
```

```
i= 1  
i= 3  
i= 5  
i= 7  
i= 9
```

```
for i in range(10, 1, -2) :  
    print("i=", i)
```

```
i= 10  
i= 8  
i= 6  
i= 4  
i= 2
```

□ 출력할 때마다 엔터(한줄내리기)를 넣지않으려면? end 변수를 특정문자로 지정한다.

```
for i in range(1, 10, 2) :  
    print(i, end=" ")
```

```
1 3 5 7 9
```

```
for i in range(1, 10) :  
    print(i, end=" ")
```

```
1 2 3 4 5 6 7 8 9
```

```
for i in range(10, -1) :  
    print(i, end=" ")
```

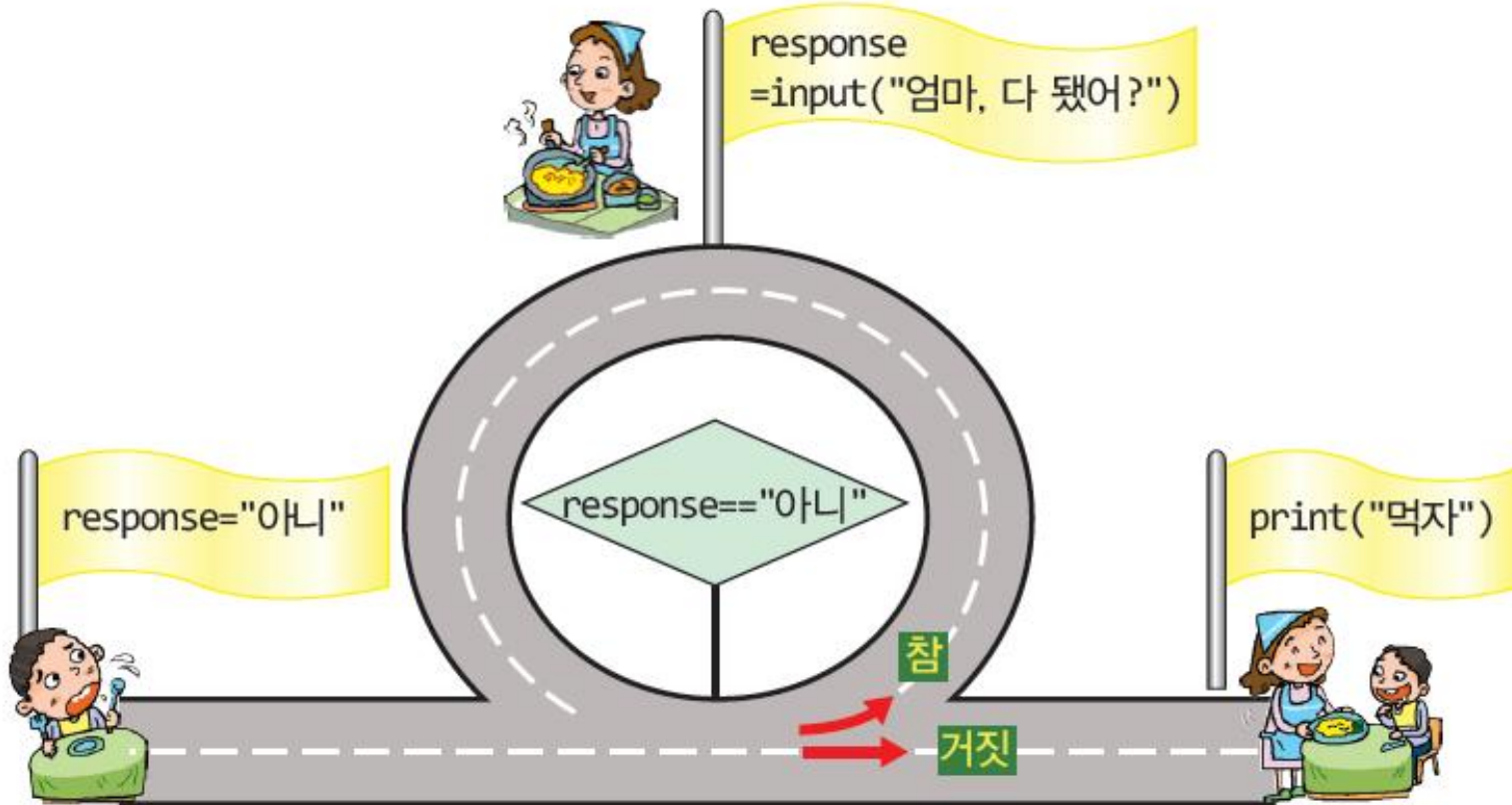
출력결과없음. 왜일까?

```
for i in [1,10,100,1000] :  
    print(i, end="**")
```

```
1*10*100*1000
```

while문 : 조건 제어 반복

- 단순 횟수가 아니라 어떤 조건이 만족되는 동안 반복하는 구조



while문 : 조건 제어 반복



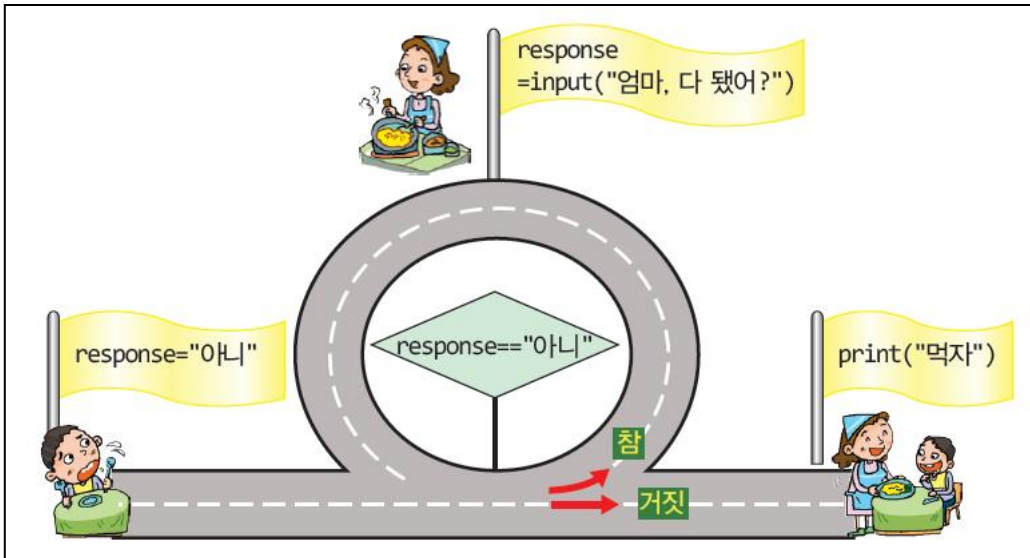
while 루프

반복을 하는 조건이다. 조건이 참이면 반복을 계속한다.

while **조건** :

반복 문장

반복되는 문장이다.



```
response = "아니"
```

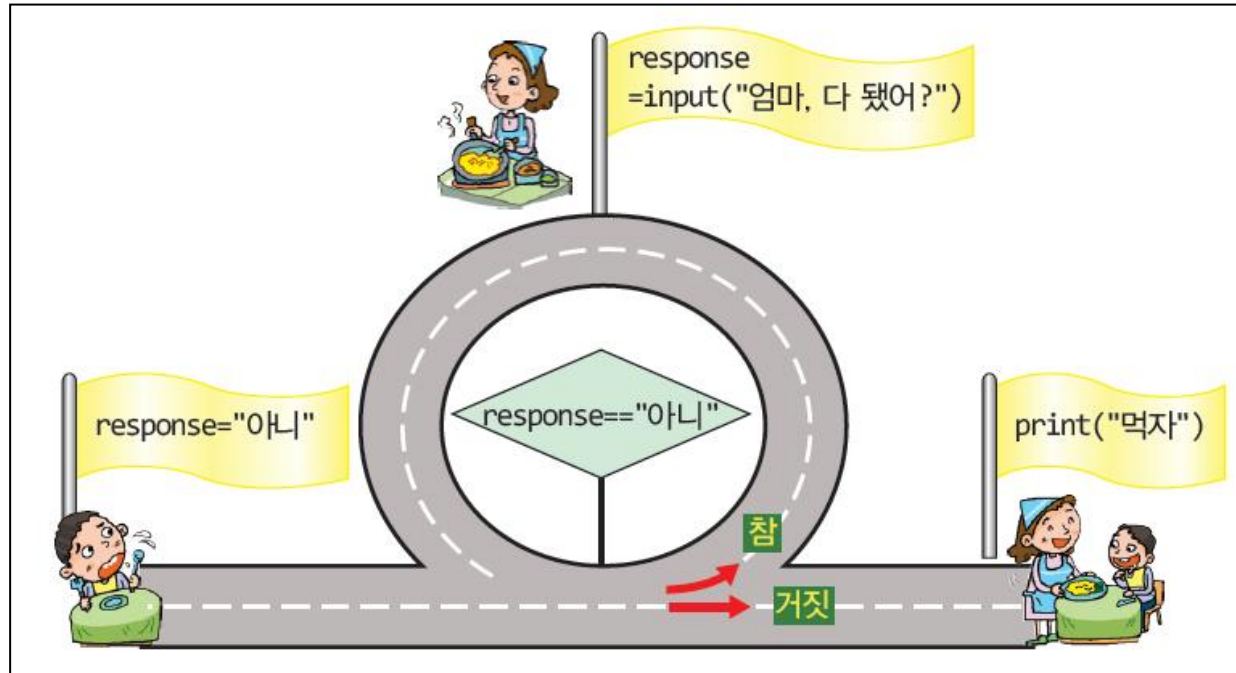
#response가 "아니"가 아니면, 반복문은 종료된다.

```
while response != "아니":
```

```
    response = input("엄마, 다됐어?");
```

```
print("먹자")
```

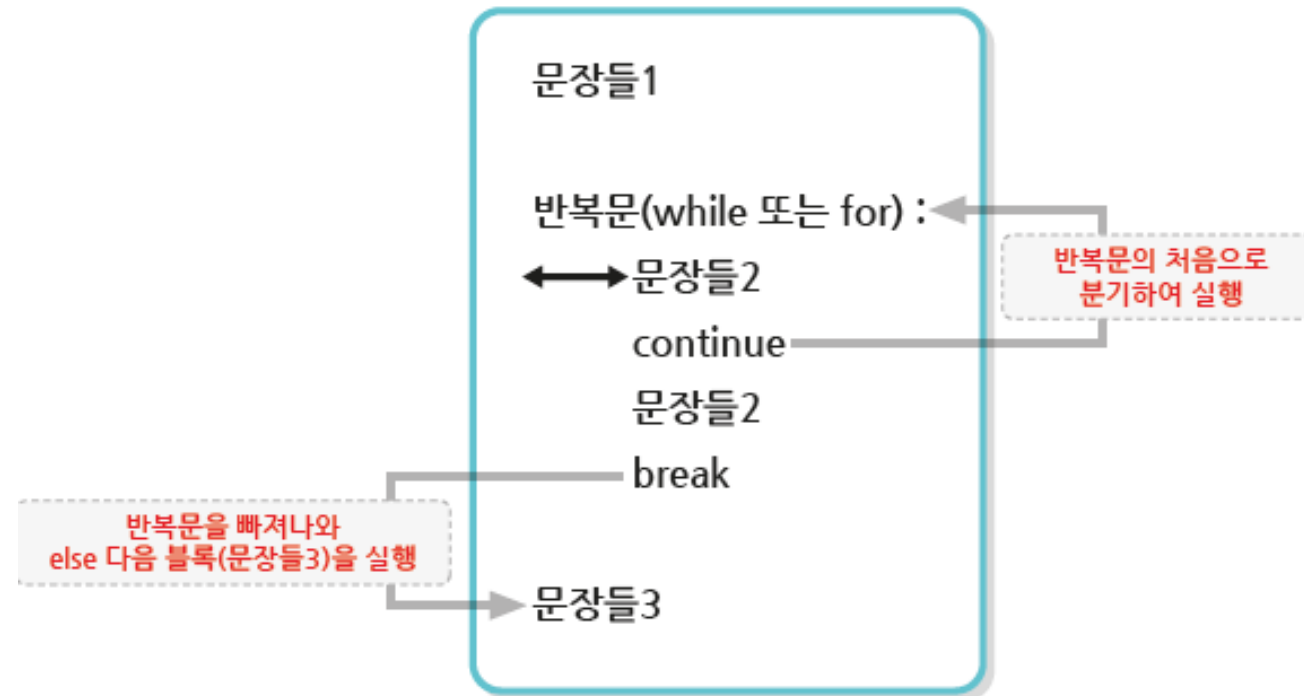

while문 : 조건 제어 반복



```
response = "아니"
while response == "아니":
    response = input("엄마, 다됐어?");
print("먹자")
```

while문 : 조건 제어 반복

- **continue문** : 제어를 반복문의 처음으로 이동시켜 다음 반복을 수행
- **break문** : 제어를 반복문 밖으로 탈출시킨다. 반복문을 종료하고 다음 문장들을 수행
- 일반적으로 continue문과 break문은 특정 조건이 만족하였을 경우 사용되기 때문에 if문과 같이 사용되는 것이 일반적이다!



Lab : 암호비교

- 사용자가 입력한 암호가 맞을 때까지 반복하자.



암호를 입력하시오: idontknow

암호를 입력하시오: 12345678

암호를 입력하시오: password

암호를 입력하시오: pythonisfun

로그인 성공

```
passwd = ""  
myPasswd = "pythonisfun"  
while passwd != myPasswd :  
    passwd = input("암호를 입력하시오: ")
```

#무한루프 스타일도 가능!

```
while True:  
    passwd = input("암호를 입력하시오: ")  
    if passwd == myPasswd : break
```

```
print("로그인 성공")
```

Lab : 조건으로 누적합과 평균 구하기

- 정수를 계속 입력받아서 누적합을 구한다.
- 단, 정수가 음수이면 누적하지 않으며, 정수의 합이 1000이상이면 총합과 평균을 구하여 출력한다.
- 평균은 반올림하여 소수점 두자리까지 구한다.

```
숫자1:1
숫자2:2
숫자3:3
숫자4:-1
숫자4:4
숫자5:1000
입력된 수 5개
총합 : 1010, 평균 : 202.00
```

```
#반복회수가 지정되지 않았으므로 while의 무한루프 스타일을 사용한다.
total=0 #사용할 변수 초기화
count=1
while True :
    #정수를 num에 입력받기
    #num이 음수가 아니면 누적하고
    #count +1증가시킨다.
    #누적합이 1000이상이면 종료한다.

#총합과 평균을 출력한다.
```

Solution

```
#정수를 계속 입력받아서 누적합 구하기
#단, 음수는 합산하지 않으며 누적합이 1000이상이 되면 총합과 평균을 구하여 출력한다.

total=0
count=0

while True :
    num = int(input("숫자"+str(count+1) + ":"))    #보기 좋은 출력을 위해
    if num<0 : continue        #num이 음수라면 반복문의 처음으로 돌아간다.
    total +=num                #total에 num을 누적한다.
    count+=1
    if(total>=1000) : break    #total>=1000이면 반복문을 종료한다.

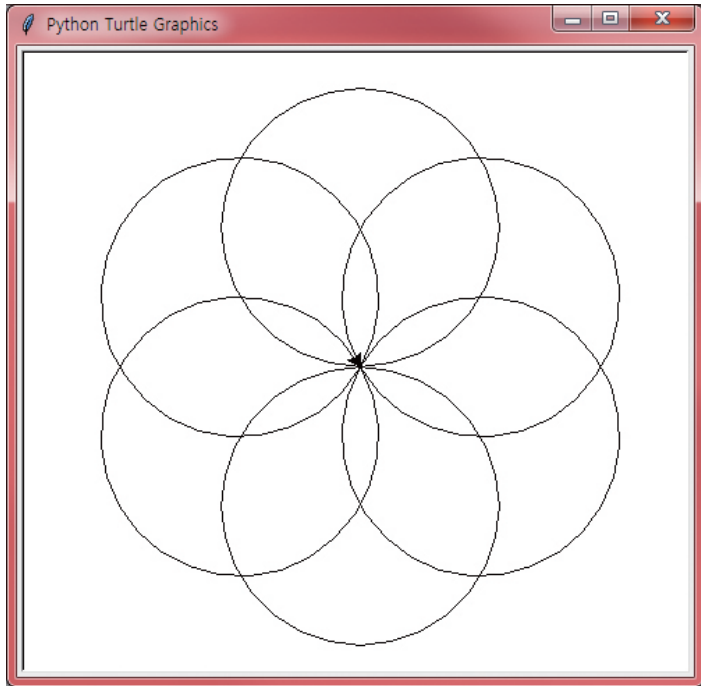
#####
print(f"입력된 수 {count}개")
print(f"총합 : {total}, 평균 : {total/count :.2f}")
```

Lab : 가벼운 실습

1. Turtle : 원그리기, 삼각형, 사각형..
2. 1부터 10까지 누적합 구하기
3. 1부터 n까지 누적곱 구하기 : $n!$
4. 제한회수가 있는 로그인 시스템
5. 사각형, 직각삼각형, 정삼각형 그리기
6. 369 게임 등
7. Supp: 거북이로 도형그리기

Lab#1 : 거북이로 원/삼각형/사각형 그리기

- 6개의 원 그리기
- 60도씩 방향을 회전하면서 원을 그린다.



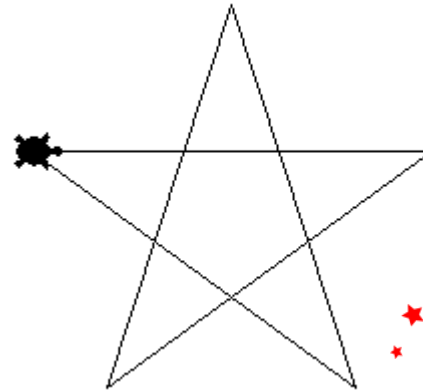
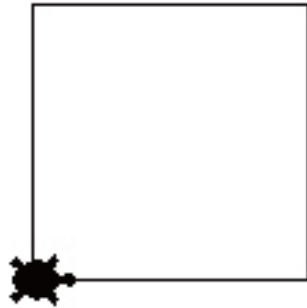
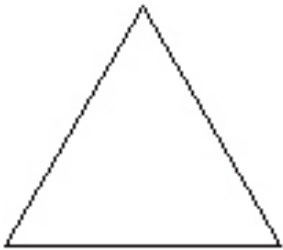
```
import turtle
t = turtle.Turtle()
n = 6
for count in range(n):
    t.circle(100)
    t.left(360/n)
```

input() #엔터를 누르면 창이 닫힌다.
#들여쓰기와 내어쓰기 주의하세요!

Lab#1 : 거북이로 원/삼각형/사각형 그리기

□ 이전에 그려왔던 다각형을 반복문을 이용하여 그려봅시다.

- 정삼각형과 정사각형
- 별그리기 : 거북이를 200 픽셀만큼 전진시키고 오른쪽으로 144도 회전하기를 5번 반복!



Solution

```
import turtle
t = turtle.Turtle()
t.shape("turtle")
```

정삼각형 그리기

```
for i in range(3):
    t.forward(100)
    t.left(360/3)
```

이동하기

```
t.penup()
t.goto(200, 0)
t.pendown()
```

정사각형 그리기

```
for i in range(4):
    t.forward(100)
    t.left(360/4)
```

이동하기

```
t.penup()
t.goto(200, 0)
t.pendown()
```

별그리기. 이번엔 while .

```
i = 0
while i < 5:
    t.forward(200)
    t.right(144)
    i = i + 1
```

#####

#정다각형 그리기

#그리고 싶은 다각형의 변 수를
#입력받아 그린다면?

import turtle

t = turtle.Turtle()

n =int(input("정다각형의 변의 개수는 : "))

#for

```
for i in range(n):
    t.forward(100)
    t.left(360/n)
```

#while도 가능

```
i=0
while i < n :
    t.forward(100)
    t.left(360/n)
    i+=1
```

Lab#2 : 1부터 10까지 누적합을 계산해보자

- 1부터 10까지의 합을 계산하는 문제를 for 와 while 루프로 작성해 보자.

1부터 10까지의 합계는 55 입니다

#도전 : 덧셈 과정을 표시해보세요

1부터 10까지의 합계는

$1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 =$
55입니다!



1

1



1+2

= 3



1+2+3

= 6



1+2+3+4

= 10

Solution : for 와 while 문으로

```
#while문
num = 1
result = 0
while count <= 10 :
    result += num
    num +=1

print( " 합계는 " , result)

#####
#for문
num = 1
result = 0
for num in range (1,11,1) :
    result += num

print("합계는", result)
```

```
#계산과정 보이기
num = 1
result = 0
print(f"1부터 10까지의 합계는 ")

for num in range (1,11,1) :
    result += num
    if (num<10) : print(num, end=" + ")
    else : print(num, " = ")
print(f"{result}입니다")
```

Lab#3: 팩토리얼 계산하기



- for문을 이용하여서 팩토리얼을 계산해보자.
- 팩토리얼 $n!$ 은 1부터 n 까지의 정수를 모두 곱한 것을 의미한다.
- 즉, $n! = 1 \times 2 \times 3 \times \dots \times (n-1) \times n$
- 곱하기의 반복

정수를 입력하시오: 10
10!은
3628800 입니다.

#도전 : 곱셈 과정을 표시해보세요

정수를 입력하시오: 10

10!은

$1 * 2 * 3 * 4 * 5 * 6 * 7 * 8 * 9 * 10 =$
3628800 입니다.

Solution

```
n = int(input("정수를 입력하시오: "))
```

```
fact = 1    #누적합, 누적곱을 구할 때는 항상 초기화!
```

```
print(n, "!은" )
```

```
for i in range(1, n+1):
```

```
    fact = fact * i    # fact *= i 로 써도 됩니다.
```

```
print(fact, "입니다.")
```

n! 누적곱 구하기

```
n = int(input("정수를 입력하시오: "))
```

```
fact = 1
```

```
print(f"{n}!은 ")    #계산 과정 보이기
```

```
for i in range(n, 0, -1):
```

```
    if (i==1) : print(i , end=" = ")
```

```
    else : print(i , end=" * ")
```

```
    fact *= i
```

```
print(f"{fact} 입니다.")
```

Lab#4: 제한회수가 있는 로그인 시스템

- id/password 정해진 횟수만큼 검사하기
- 아이디와 패스워드를 미리 설정해 두고, 사용자에게 입력을 받아 로그인한다.
- 단, 제한회수를 두어 검사한다

아이디: red

패스워드:12345

id 또는 password가 틀렸습니다. 다시 입력하세요 (1/5)

아이디: red

패스워드:pythonisfun

id 또는 password 가 틀렸습니다. 다시 입력하세요 (2/5)

아이디: blue

패스워드:pythonisfun!

id 또는 password 가 틀렸습니다. 다시 입력하세요 (3/5)

아이디: blue

패스워드:pythonisfun

로그인 성공!

로그인 시스템을 종료합니다.

Solution

```
myid = "blue"
mypasswd = "pythonisfun"
count = 5    #제한회수는 5회로 지정
i=1
while True :
    if (i > count) :
        print("기회를 모두 사용하셨습니다.")
        break
    id= input("아이디: ")
    passwd = input("패스워드:")
    if(myid==id and mypasswd == passwd) :
        print("로그인 성공!")
        break
    else :
        print("id 또는 password가 틀렸습니다. 다시 입력하세요 (%d/%d )" % (i, count) )
        i+=1

print("로그인 시스템을 종료합니다.")
```

무한반복문의 탈출조건

Lab#5 : 사각형, 직각삼각형, 정삼각형

□ 직선그리기 :

- ■ ■ ■ ■ ■ ■ ■ ■ ■

```
#반복문을 이용하여
for i in range(10):
    print("■ ", end=" ")
print()
```

```
#문자열 연결을 이용하여
print("■ "*10)
print()
```

□ 사각형 :

- 너비와 높이를 입력받는다.
- Hint : 너비만큼 반복하는 문장을 높이만큼 반복하도록 한다

□ 오른쪽 직각삼각형 :

- 높이를 입력받는다
- Hint : 사각형과 비슷한 구조이다. 단, 너비가 1부터 높이만큼 증가한다.

□ 정삼각형 :

- 높이를 입력받는다.
- Hint : 직각삼각형과 비슷한 구조이다. 단, 공백을 추가하는 반복문을 추가한다.

```
사각형 그리기
너비: 10
높이: 5
1 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
2 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
3 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
4 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
5 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■

-----
높이: 5
오른쪽 직각삼각형
1 : ★
2 : ★★
3 : ★★★
4 : ★★★★
5 : ★★★★★

-----
정삼각형
1 :      ★
2 :     ★★
3 :    ★★★
4 :   ★★★★
5 :  ★★★★★
PS D:\python코드> []
```


Solution : 사각형 + 오른쪽 직각삼각형 + 정삼각형

```
#너비와 높이를 입력받아서 "■ 나 □"을 이용하여
#사각형을 그려보자!
print("사각형 그리기")
width = int(input("너비: "))
height = int(input("높이: "))

for h in range(height):
    print(f"{h+1:2d} : ", end="") #line 번호 보이기. 생략가능
    for w in range(width):
        print("■ ", end="")
    print()

print("--"*10)
```

```
PS D:\python코드> 사각형그리기.py
사각형 그리기
너비: 10
높이: 5
1 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
2 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
3 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
4 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
5 : ■ ■ ■ ■ ■ ■ ■ ■ ■ ■
-----
```

```
for h in range(height):
    print(f"{h+1:2d} : ", end="") #라인번호 보이기. 생략가능
    print("■ "*width)
```

#높이를 입력받아서 오른쪽 직각삼각형과 정삼각형을 그려보자

```
height = int(input("높이: "))
```

```
print("오른쪽 직각삼각형")
```

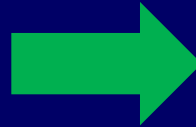
```
for h in range(height):  
    print(f"{h+1:2d} : ", end="")  
    for w in range(h+1):  
        print("★ ", end="")  
    print()
```

```
#####
```

```
print("정삼각형")
```

```
for h in range(height):  
    print(f"{h+1:2d} : ", end="")  
    for w in range(height-h+1):  
        print(" ", end="")  
    for w in range(h+1):  
        print("★ ", end="")  
    print()
```

```
print("끝!")
```



```
print("오른쪽 직각삼각형")  
for h in range(height):  
    print(f"{h+1:2d} : ", end="")  
    print("★ "*(h+1))
```

#오른쪽 직각삼각형 코드에
#빈칸을 출력하는 반복문을 추가한다.



```
print("정삼각형")  
for h in range(height):  
    print(f"{h+1:2d} : ", end="")  
    print(" "*(height-h+1), end="") #빈칸찍기  
    print("★ "*(h+1)) #별찍기
```

높이: 5

오른쪽 직각삼각형

```
1 : ★  
2 : ★ ★  
3 : ★ ★ ★  
4 : ★ ★ ★ ★  
5 : ★ ★ ★ ★ ★
```

정삼각형

```
1 :      ★  
2 :      ★ ★  
3 :      ★ ★ ★  
4 :      ★ ★ ★ ★  
5 :      ★ ★ ★ ★ ★
```

Lab #6 : 369 게임

- 1부터 100까지 카운트 하면서
- 십의자리가 3의 배수이면 “짝”, 일의자리가 3의 배수라면 “꿇”을 출력합니다.
- 생각해볼 것
 - 어떤 수의 십의자리를 구하는 방법은?
 - 어떤 수의 일의자리를 구하는 방법은?
 - 그 수가 3의 배수인지 알아보는 방법은?

```
1 :  
2 :  
3 : 꿇  
4 :  
5 :  
6 : 꿇  
7 :  
8 :  
9 : 꿇  
10 :  
11 :  
12 :  
13 : 꿇  
14 :  
15 :  
16 : 꿇  
17 :  
18 :  
19 : 꿇  
20 :  
21 :  
22 :  
23 : 꿇  
24 :  
25 :  
26 : 꿇  
27 :  
28 :  
29 : 꿇  
30 : 짝  
31 : 짝  
32 : 짝  
33 : 꿇
```

Solution : 여러가지 방법이 있습니다

```
# 3,6 9
a=""
b=""
for i in range(1,101,1) :
    십 = i // 10;
    일 = i % 10;
    if(십 != 0 and 십%3 ==0) : a = "짝"      # 십의 자리가 0이 아니고 3의 배수이면
    If(일 != 0 and 일%3 ==0) : b = "꿇"    # 일의 자리가 0이 아니고 3의 배수이면
    print(" %d : %s%s"%(i,a,b))
    a=""                                     #다음번 수를 위해 다시 초기화
    b=""
```

```

1 #나누기를 사용한 홀수짝수
2 for i in range(20) :
3     if(i % 2 == 0) : print(i,": 짝수")
4     else : print(i ,": 홀수")
5
6
7 #나누기를 사용한 369 게임
8 for i in range(1, 41) :
9     십 = i // 10
10    일 = i % 10
11    count =0
12    if(십 !=0 and 십 % 3 == 0) :
13        print(" 짹", end ="" )
14        count +=1
15
16    if(일 !=0 and 일 % 3 == 0) :
17        count+=1
18        if (count>=1) : print(" 짹!" )
19        else : print("\n 짹!")
20
21
22    if (count ==0) : print( i , end ="" )
23
24

```

```

0 : 짹수
1 : 홀수
2 : 짹수
3 : 홀수
4 : 짹수
5 : 홀수
6 : 짹수
7 : 홀수
8 : 짹수
9 : 홀수
10 : 짹수
11 : 홀수
12 : 짹수
13 : 홀수
14 : 짹수
15 : 홀수
16 : 짹수
17 : 홀수
18 : 짹수
19 : 홀수
1 2 짹!
4 5 짹!
7 8 짹!
10 11 12 짹!
14 15 짹!
17 18 짹!
20 21 22 짹!
24 25 짹!
27 28 짹!
  짹 짹 짹 짹!
  짹 짹 짹!
  짹 짹 짹!
40

```

이번 장 정리

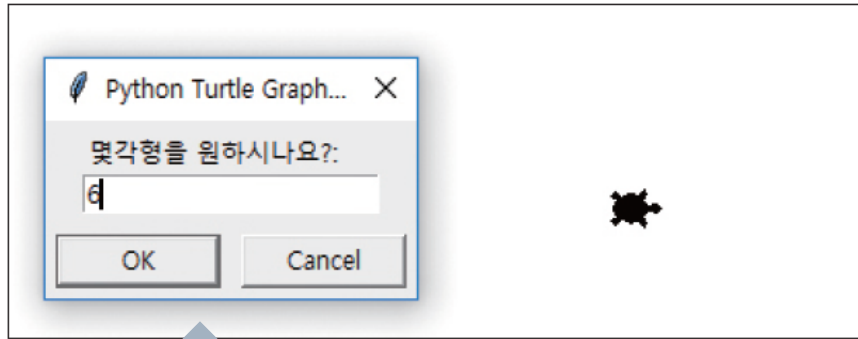
- 문장들을 반복 실행하려면 for나 while을 사용한다.
- 반복 실행되는 문장들을 들여쓰기 하여야 한다.
- for 문은 반복 회수를 정해져있을 때 유용하다.
- while 문은 반복 조건이 정해져 있을 때 유용하다.
- 반복문의 초입에서 조건식은 검사된다.

- 오늘의 실습활동
 - 반복문을 사용하는 프로그램 3개이상 만들어 올리기

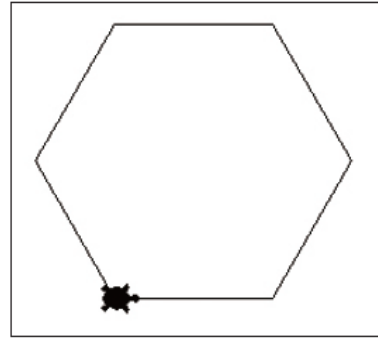
Supp. 거북이로 도형그리기

Lab: n-각형 그리기

- 사용자로부터 정수 n 을받아서 n -각형을 그리는 프로그램을 작성해봅시다.



`turtle.textinput("몇각형을 원하시나요? :")`



```
import turtle
t = turtle.Turtle()
t.shape("turtle")
```

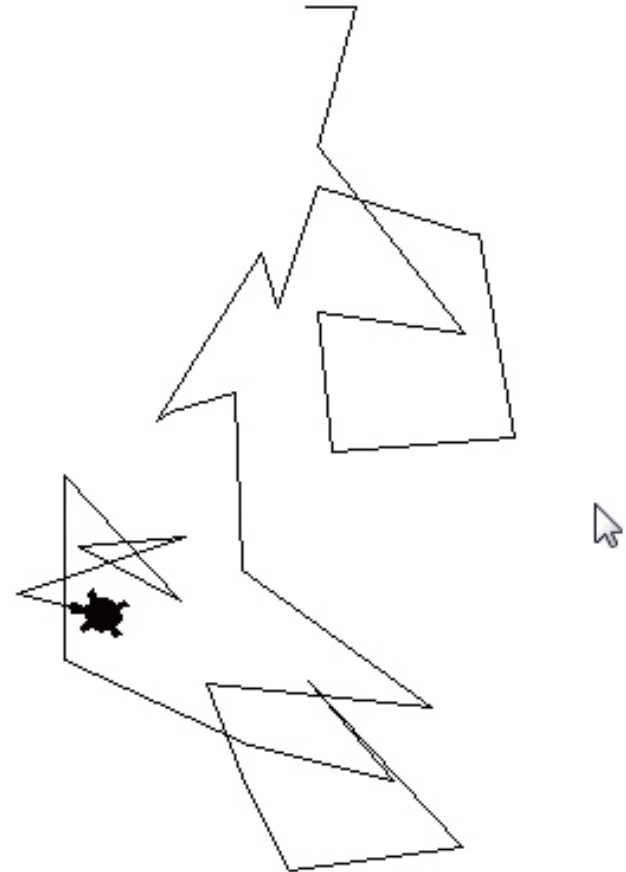
```
s = turtle.textinput("", "몇각형을 원하시나요?:")
n=int(s) #정수로 바꾸어 n에저장하고
```

```
for i in range(n):      # 반복!
    t.forward(100)
    t.left(360/n)

input()
```


Lab: 거북이를 랜덤하게 움직이게 하자

- 거북이가 술에 취한 것처럼 랜덤하게 움직이게 해보자.
- 이전에 사용한 Random 함수를 이용해서
 - 반복회수는 30번. 거리와 각도를 랜덤하게 얻는다.
 - 거리는 1~100
 - 각도는 -180~180
 - 방향은 오른쪽으로



알고리즘 : 순서와 방법 생각하기

30번 반복

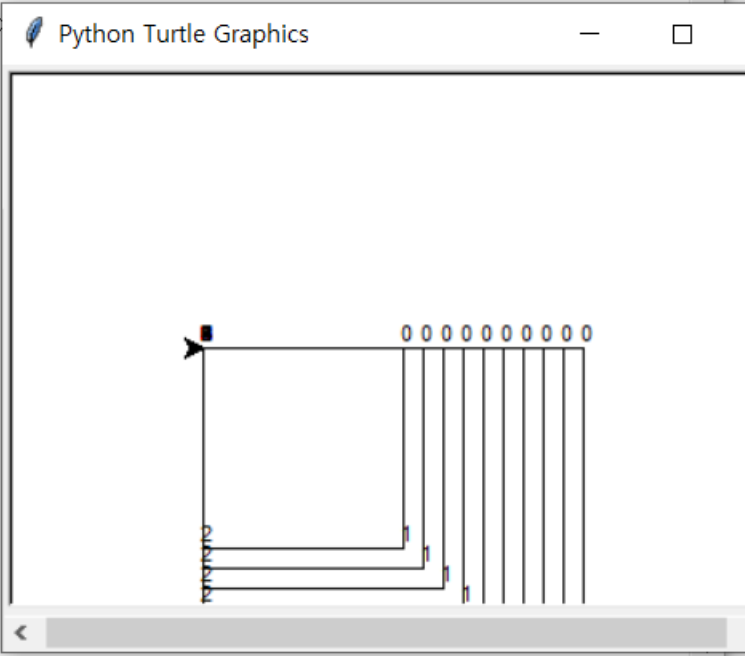
- * [1, 100] 사이의 난수를 발생하여 변수 length에 저장한다.
- * 거북이를 length만큼 움직인다.
- * [-180, 180] 사이의 난수를 발생하여 변수 angle에 저장한다.
- * 거북이를 angle만큼 회전시킨다.

```
① import turtle
② import random
③ t = turtle.Turtle()
④ t.shape("turtle")
⑤ count=30
⑥ for i in range(count):
⑦     length = random.randint(1, 100)
⑧     t.forward(length)
⑨     angle = random.randint(-180, 180)
⑩     t.right(angle)
```

Lab : 여러 개의 사각형그리기 - 2개의 while 루프

□ while 루프를 이용하여서 화면에 사각형을 그리는 코드를 작성해보자.

```
===== RESTART: D:\Wlec
===
k:0, i : 0,1,2,3,
k:1, i : 0,1,2,3,
k:2, i : 0,1,2,3,
k:3, i : 0,1,2,3,
k:4, i : 0,1,2,3,
k:5, i : 0,1,2,3,
k:6, i : 0,1,2,3,
k:7, i : 0,1,2,3,
k:8, i : 0,1,2,3,
k:9, i : 0,1,2,3,
!
>>>
```



Ln: 436 Col: 4

```
import turtle
t = turtle.Turtle()
i = 0
k = 0
count = 10
t.goto(-100, 0)
# 두개의 중첩된 반복문
while (k < count) :
    i = 0
    t.write(k)
    print(" k:%d, i : "%k, end = "")
    while (i < 4) :
        t.forward(100+ k*10) #길이조정
        t.right(90)
        t.write(i)
        print( i, end=",")
        i = i + 1 # i를 1씩 증가.
    k = k+1
    print("\n")
print("!")
```

Lab : 여러 개의 사각형그리기 - 2개의 for 루프

```
import turtle
t = turtle.Turtle()
i=0 #변수의 값을 0으로 초기화
k=0 #변수의 값을 0으로 초기화
count = 10 #count 변수값은 10으로 초기화
t.goto(-100, 0)
# 두개의 중첩된 반복문
```

```
while (k < count) :
    i=0 #내부반복문에서 i는 0,1,2,3 으로 증가하므로 rewind
    t.write(k)
    print(" k:%d, i : "%k, end = "")
    while (i < 4) :
        t.forward(100+ k*10) #길이조정
        t.right(90)
        t.write(i)
        print( i, end=",")
        i = i + 1 # i 를 1씩 증가
    k = k+1
    print("\n")
print("!")
```

```
import turtle
t= turtle.Turtle()
count = 10
t.goto(-100, 0)

for k in range(count) :
    t.write(k)
    print("k:%d, i : "%k, end = "")
    for i in range(4) :
        t.forward(100+ k*10)
        t.right(90)
        t.write(i)
        print( i, end=",")
    print("\n")
print("!")
```

수고하셨습니다.