



단국대학교  
SW중심대학

# 창의적 사고와 코딩

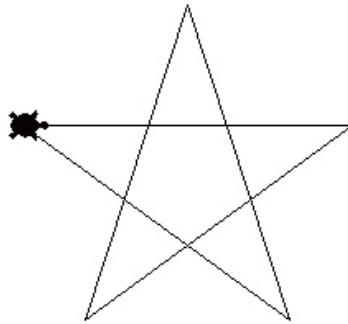
## Chap 06. 반복문

박 소 현

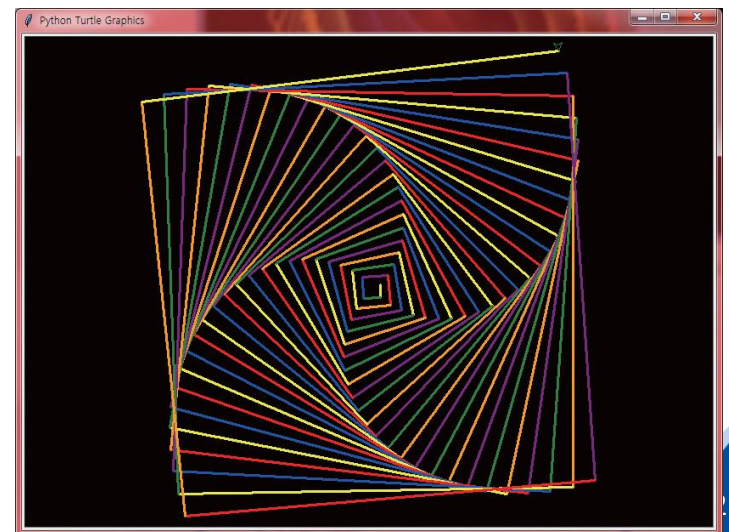
[sohyunpark@dankook.ac.kr](mailto:sohyunpark@dankook.ac.kr)



(1) 반복을 이용해서 별을 그려보자



(2) 반복을 이용하여 스파이럴을 그려보자.



- 6.1 반복의 개념과 중요성

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



## 왜 반복이 중요한가?

- 하나의 예로 화면에 회사에 중요한 손님이 오셔서 대형 전광판에 '방문을 환영합니다!'를 5번 출력한다고 하자.

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



# 왜 반복이 중요한가?



```
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 붙여넣기 ?



# 만약 1000번 반복해야 한다면?



- 반복 구조를 사용하여야 한다.

```
for i in range(1000):  
    print("방문을 환영합니다!")
```

1000번 반복시키는 구조

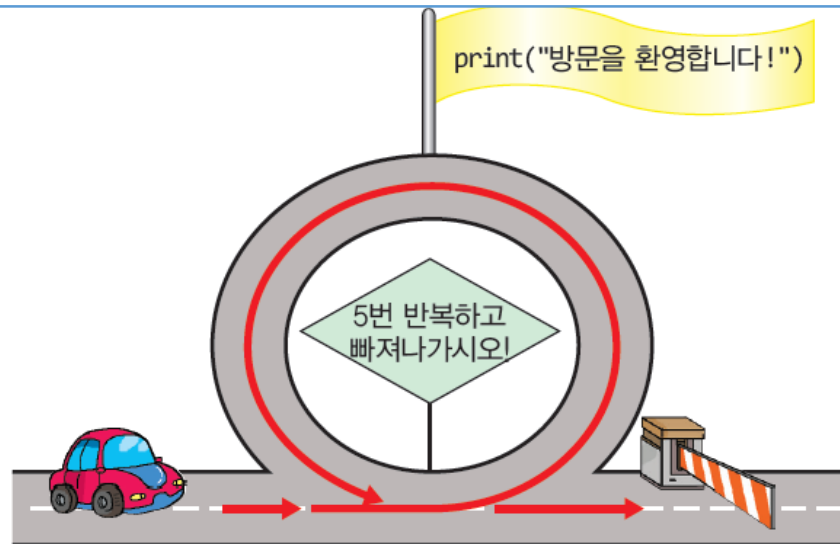


- 파이썬에서 횟수 제어 반복은 for 루프라고도 한다.

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


# 끝에 :이 있어야 함

# 들여쓰기 하여야 함



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

# 끝에 :이 있어야 함  
# 들여쓰기 하여야 함



```
방문을 환영합니다!  
방문을 환영합니다!  
방문을 환영합니다!  
방문을 환영합니다!  
방문을 환영합니다!
```

*i가 1부터 5까지 변경되면서 반복된다.*

## i의 값을 출력해보자.



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

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

## 구구단을 출력해보자.



```
for i in [1, 2, 3, 4, 5]:  
    print("9*", i, "=", 9*i)
```

```
9* 1 = 9  
9* 2 = 18  
9* 3 = 27  
9* 4 = 36  
9* 5 = 45
```

# range() 함수

```
for 문
```

```
for 변수 in range( 종료 값 ) :
```

```
    문장
```

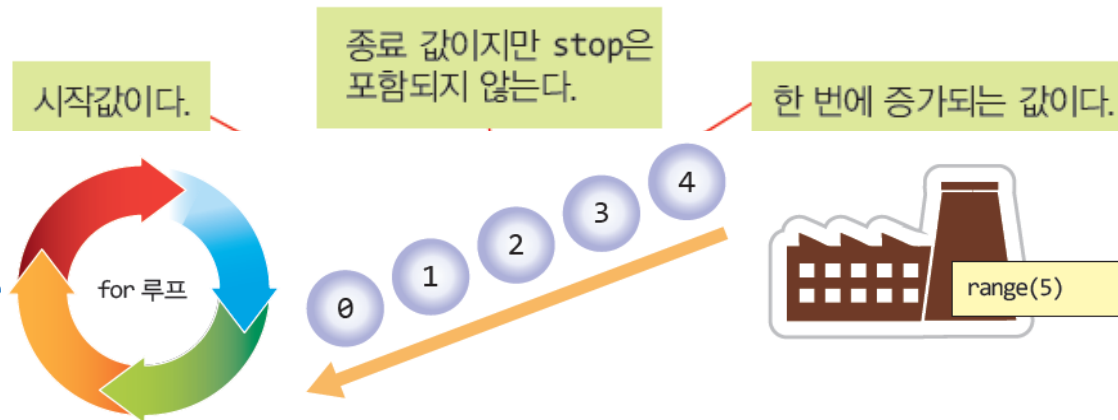
0에서 (종료 값-1)까지의 숫자를 반환한다.

반복되는 문장으로  
들여쓰기 하여야 한다.

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

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

## range() 함수



- 만약 1부터 시작하여서 5까지 반복하고 싶다면 어떻게 하면 될까?

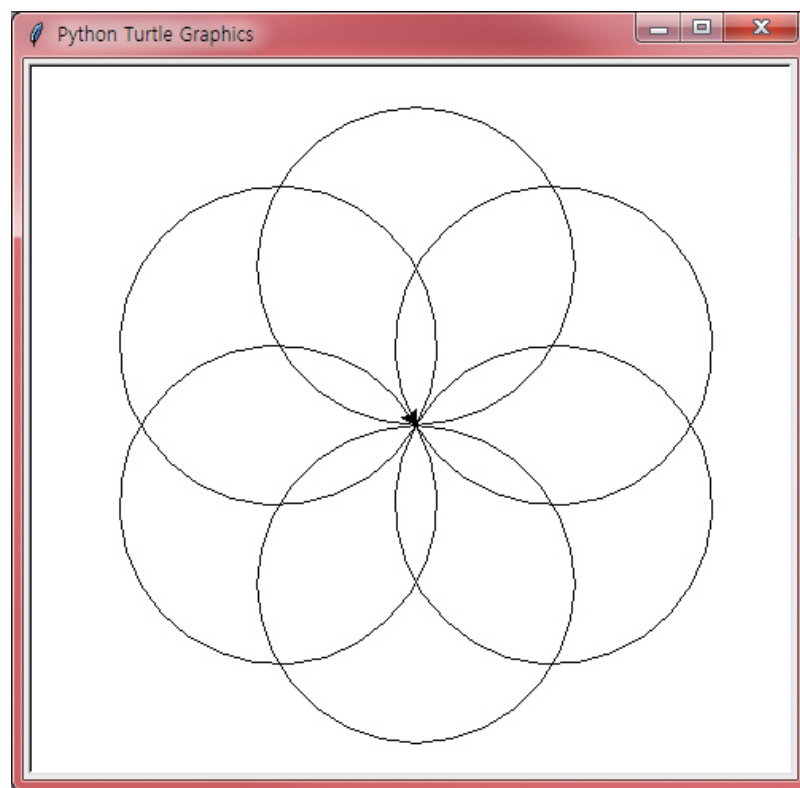
1 2 3 4 5



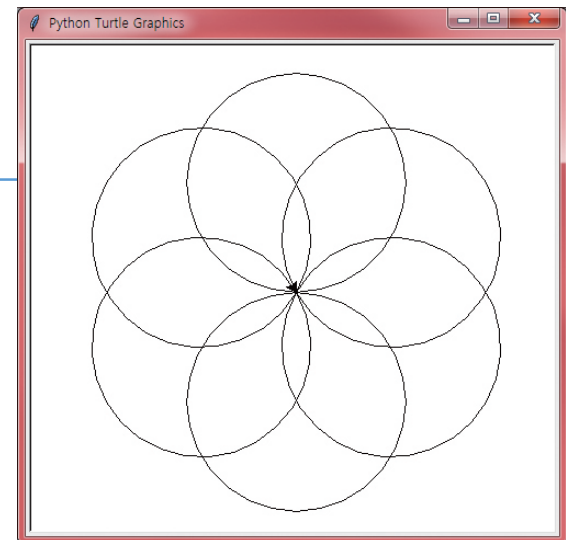
```
for i in range(1, 6, 1):  
    print(i, end=" ")
```

1 2 3 4 5

## ▪ 6개의 원 그리기



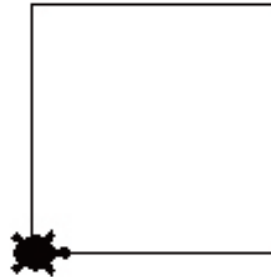
```
import turtle  
t = turtle.Turtle()  
  
for count in range(6):  
    t.circle(100)  
    t.left(360/6)
```



## Lab1 : 반복을 사용하여 도형 그리기



- 정삼각형과 정사각형을 반복을 이용하여 화면에 그려 보자.



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

# 정삼각형 그리기
for i in range(3):
    t.forward(100)
    t.left(360/3)

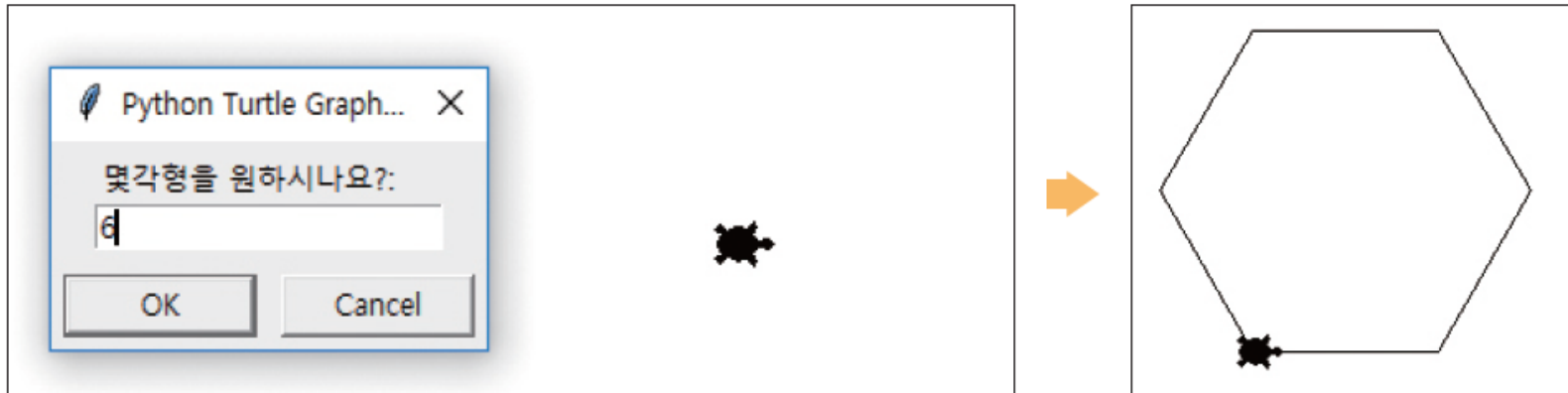
# 이동하기
t.penup()
t.goto(200, 0)
t.pendown()

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

## Lab2: n-각형 그리기



- 사용자로부터 정수  $n$ 을받아서  $n$ -각형을 그리는 프로그램을 작성할 수 있는가?



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

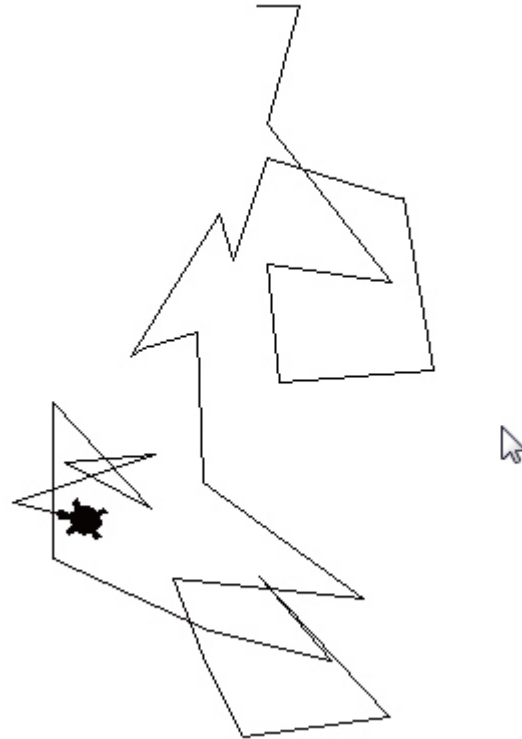
s = turtle.textinput("", "몇각형을 원하시나요?:")
n=int(s)

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

## Lab3: 거북이를 랜덤하게 움직이게 하자



- 터틀 그래픽에서 거북이가 술에 취한 것처럼 랜덤하게 움직이게 해보자.





## 30번 반복

- \*  $[1, 100]$  사이의 난수를 발생하여 변수 `length`에 저장한다.
- \* 거북이를 `length`만큼 움직인다.
- \*  $[-180, 180]$  사이의 난수를 발생하여 변수 `angle`에 저장한다.
- \* 거북이를 `angle`만큼 회전시킨다.

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

for i in range(30):
    randrange(1, 100)
    length = random.randint(1, 100)
    t.forward(length)
    angle = random.randint(-180, 180)
    t.right(angle)
```

## Lab4 : 팩토리얼 계산하기



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

정수를 입력하시오: 10  
 $10!$ 은 3628800이다.

```
n = int(input("정수를 입력하시오: "))
```

```
fact = 1
```

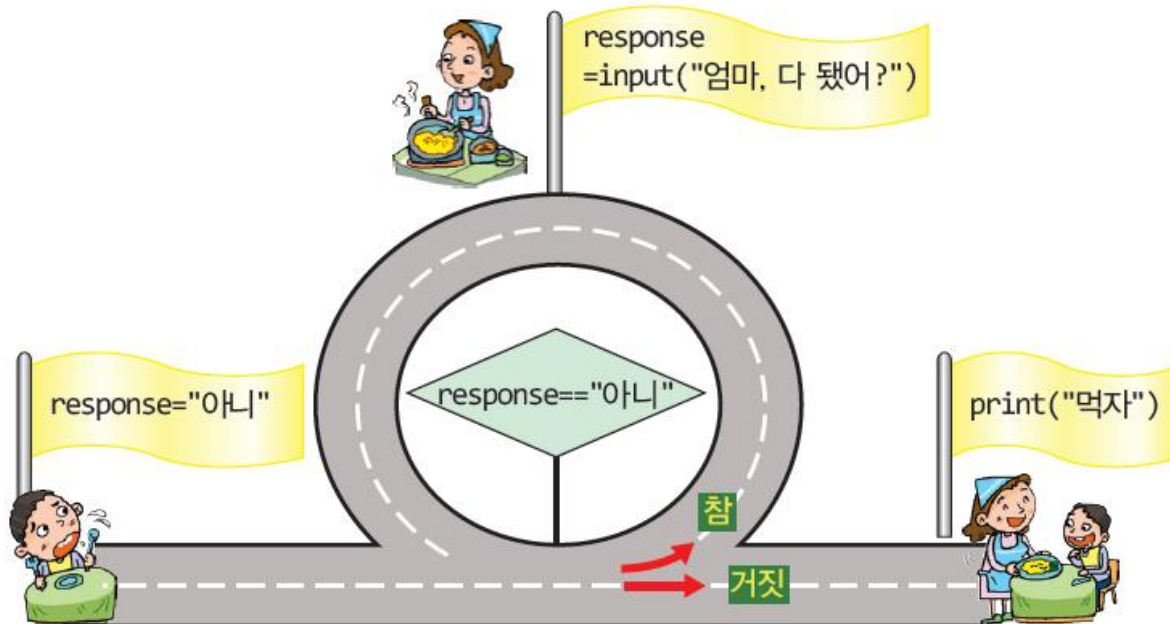
```
for i in range(1, n+1):
```

```
    fact = fact * i
```

```
print(str(n) + "!은", fact, "이다.")
```

- 6.2 조건 제어 반복

- 조건 제어 반복은 어떤 조건이 만족되는 동안 반복하는 구조



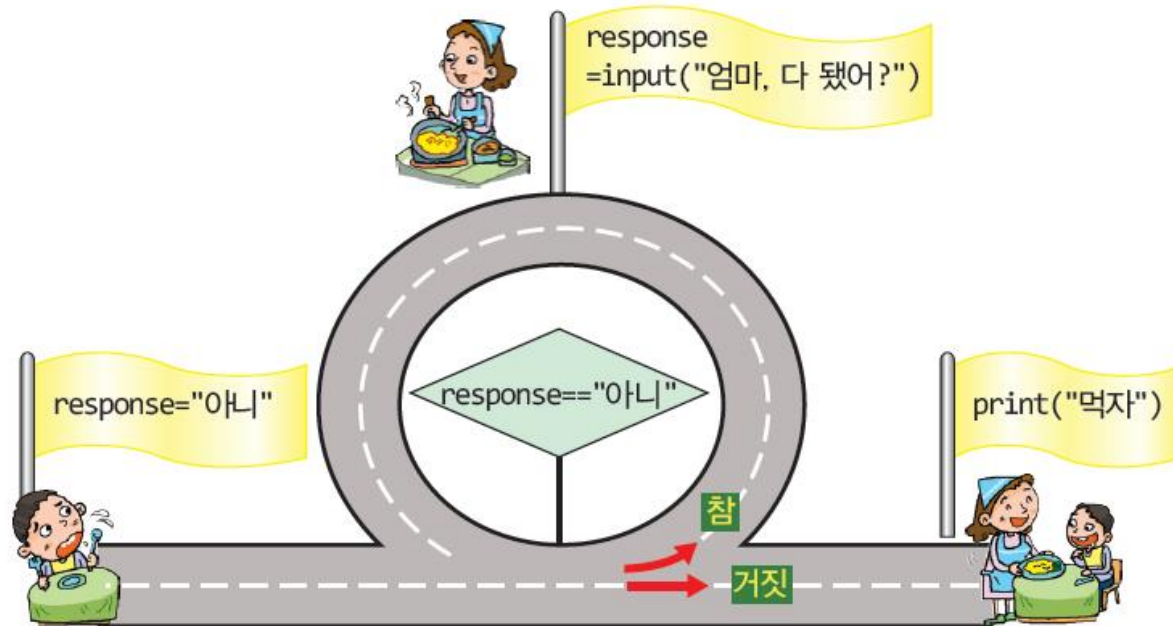
while 루프

while 조건 :

반복 문장

반복을 하는 조건이다. 조건이 참이면 반복을 계속한다.

반복되는 문장이다.



```
response = "아니"
```

```
while response == "아니":
```

```
    response = input("엄마, 다 됐어?");
```

```
print("먹자")
```



- 사용자가 암호를 입력하고 프로그램에서 암호가 맞는지를 체크한다고 하자.

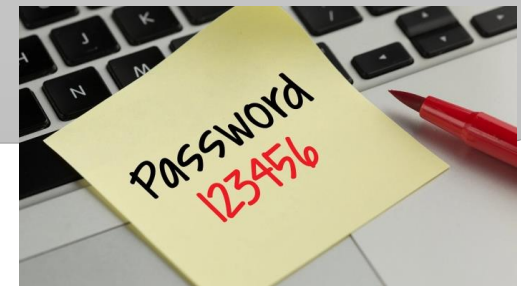
암호를 입력하시오: idontknow

암호를 입력하시오: 12345678

암호를 입력하시오: password

암호를 입력하시오: pythonisfun

로그인 성공



```
password = ""  
while password != "pythonisfun":  
    password = input("암호를 입력하시오: ")  
print("로그인 성공")
```

- 예를 들어서 1부터 10까지의 합을 계산하는 예제를 while 루프로 작성해 보자.

합계는 55



1

1



1+2

= 3



1+2+3

= 6

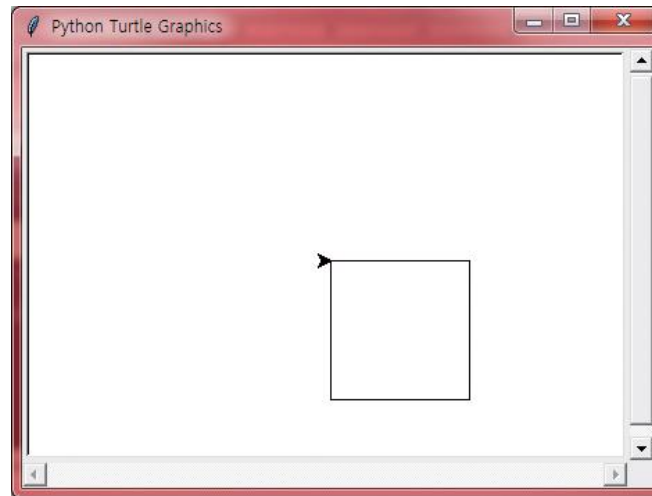


1+2+3+4

= 10

```
count = 1
sum = 0
while count <= 10 :
    sum = sum + count
    count = count + 1
print("합계는", sum)
```

- `while` 루프를 이용하여서 화면에 사각형을 그리는 코드를 작성해보자.



```
import turtle
t = turtle.Turtle()
i = 0
while i < 4:
    t.forward(100)
    t.right(90)
    i = i + 1
```



- 구구단 중에서 9단을 반복문을 이용하여 출력해보자.  $9*1$ ,  $9*2$ ,  $9*3$ , ...,  $9*9$ 까지 9번 반복시키면 출력하면 될 것이다.

원하는 단은: 9

$$9*1=9$$

$$9*2=18$$

$$9*3=27$$

$$9*4=36$$

$$9*5=45$$

$$9*6=54$$

$$9*7=63$$

$$9*8=72$$

$$9*9=81$$

```
dan = int(input("원 하는 단은: "))  
i = 1  
  
while i <= 9:  
    print("%s*%s=%s" % (dan, i, dan*i))  
    i = i + 1
```



## 도전문제

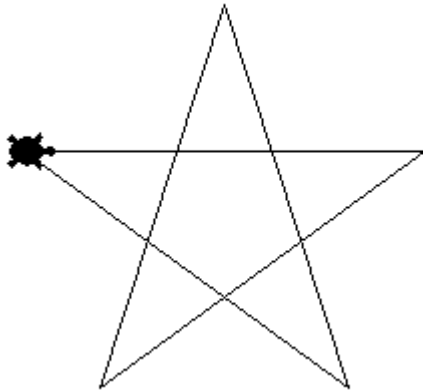
구구단의 1단부터 9단까지를 모두 출력하도록 위의 프로그램을 수정해보자.



## Lab6: 별 그리기



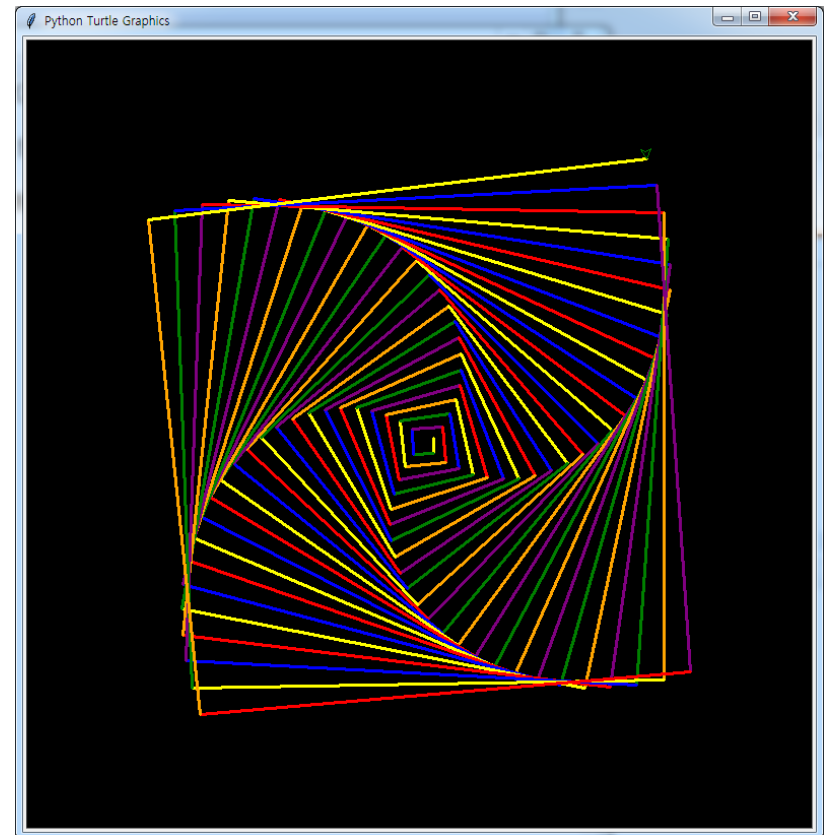
- 반복문을 사용하여 별을 그려보자.
- 5번 반복하고 반복할 마다 거북이를 50픽셀만큼 전진시키고 오른쪽으로 144도 회전하면 별이 그려진다.



```
import turtle
t = turtle.Turtle()
t.shape("turtle")
i = 0
while i < 5:
    t.forward(200)
    t.right(144)
    i = i + 1
```

### ■ 사각형을 그리는 것이지만 한 번 반복할 때마다 각도가 90도가 아니라 89도로 회전

- 색상은 리스트에 저장했다가 하나씩 꺼내 변경
- 배경색 변경
- 거북이의 속도 설정 (0이 최대 속도)
- 거북이가 그리는 선의 두께



```
import turtle
```

```
# 색상은 리스트에 저장했다가 하나씩 꺼내서 변경하도록 하자.
colors = ["red", "purple", "blue", "green", "yellow", "orange" ]
t = turtle.Turtle()
```

```
# 배경색은 다음과 같은 문장으로 변경이 가능하다.
```

```
turtle.bgcolor("black")
```

```
t.bgcolor()
turtle!
```

```
# 거북이의 속도는 0으로 설정하면 최대가 된다.
```

```
t.speed(0)
```

```
# 거북이가 그리는 선의 두께는 width()를 호출하면 된다.
```

```
t.width(3)
```

```
length = 10      # 초기 선의 길이는 10으로 한다.
```

```
# while 반복문이다. 선의 길이가 500보다 작으면 반복한다.
```

```
while length < 500:
```

```
    t.forward(length)
```

```
    t.pencolor(colors[length%6])
```

```
    t.right (89)
```

```
    length += 5
```

```
# length만큼 전진한다.
```

```
# 선의 색상을 변경한다.
```

```
# 89도 오른쪽으로 회전한다.
```

```
# 선의 길이를 5만큼 증가한다.
```

- 사용자가 입력하는 숫자를 더하는 프로그램
- 사용자가 yes라고 답한 동안에만 숫자를 입력 받음

```
숫자를 입력하시오: 10  
계속?(yes/no): yes  
숫자를 입력하시오: 20  
계속?(yes/no): yes  
숫자를 입력하시오: 5  
계속?(yes/no): no  
합계는 : 35
```

- total을 0으로 설정
- answer를 'yes'로 설정
- answer가 'yes'인 동안에 다음을 반복
  - ① 숫자를 입력 받는다
  - ② 숫자를 total에 더한다
  - ③ '계속? Yes/no'를 묻는다
- total의 값을 출력한다

```
total = 0
answer = "yes"
while answer == "yes":
    number = int(input("숫자를 입력하시오: "))
    total = total + number
    answer = input("계속?(yes/no): ")
print("합계는 : ", total)
```

- 프로그램이 선택한 정수를 사용자가 맞히는 게임
- 사용자가 답을 제시하면 프로그램의 정수와 비교하여 높은지 낮은지 만을 알려 줌

- 게임이 끝나며 몇 번 만에 맞추어느지도 출력  
1부터 100 사이의 숫자를 맞추시오  
숫자를 입력하시오: 50  
낮음!  
숫자를 입력하시오: 85  
낮음!  
숫자를 입력하시오: 93  
낮음!  
숫자를 입력하시오: 97  
높음!  
숫자를 입력하시오: 96  
축하합니다. 시도횟수= 5

```
import random
```

```
tries = 0
```

```
guess = 0;
```

```
answer = random.randint(1, 100)
```

```
print("1부터 100 사이의 숫자를 맞추시오")
```

```
while guess != answer:
```

```
    guess = int(input("숫자를 입력하시오: "))
```

```
    tries = tries + 1
```

```
    if guess < answer:
```

```
        print("낮음!")
```

```
    elif guess > answer:
```

```
        print("높음!")
```

```
if guess == answer:
```

```
    print("축하합니다. 시도횟수=", tries)
```

```
else:
```

```
    print("정답은 ", number)
```

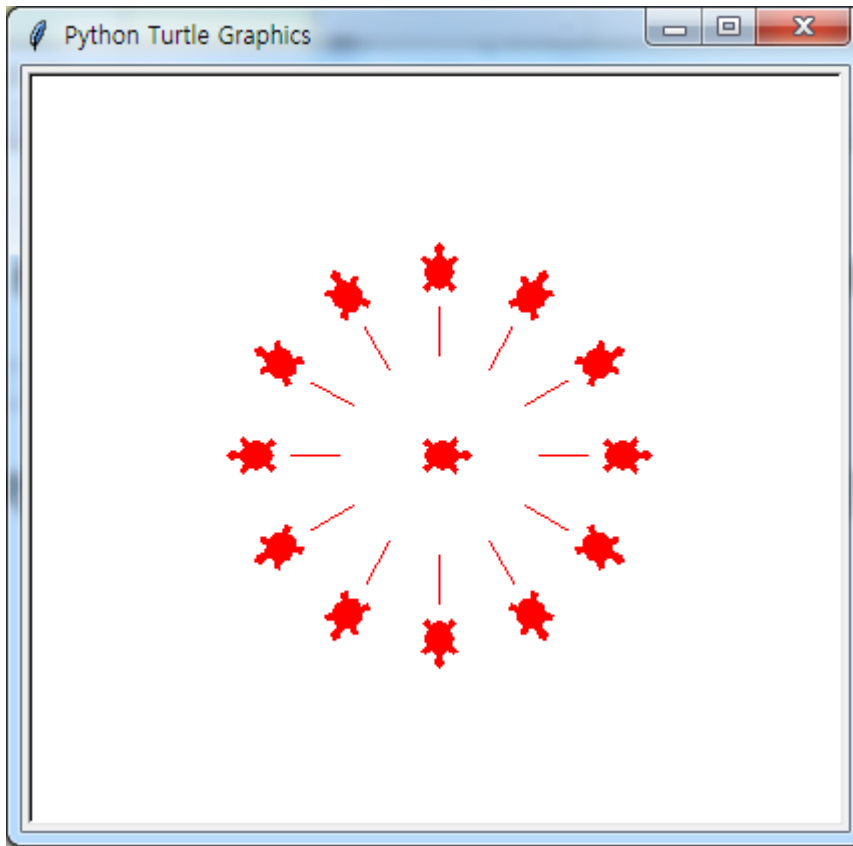
0  
!!!!  
NOT "1"



- 무한 루프 사용시 제어를 위해 break문을 사용

```
while True :  
    반복 문장  
    반복 문장  
    if 조건 :  
        break;
```

- `stamp()` 함수 이용



```
import turtle
t = turtle.Turtle()
t.shape("turtle")
t.color("red")
t.stamp()
move = 30
for i in range(12):
    t.penup()
    t.forward(50)
    t.pendown()
    t.forward(25)
    t.penup()
    t.forward(15)
    t.stamp()
    t.home()
    t.right(move)
    move = move+30
```

```
# 12번 반복
# 펜을 올린다.
# 50만큼 전진
# 펜을 내린다.
```

- 관계 연산자를 학습하였다.
- 논리 연산자 and나 or 를 사용하면 조건들을 묶을 수 있다.
- 블록은 조건이 맞았을 때 묶어서 실행되는 코드로 파이썬에서 들여쓰기로 블록을 만든다.
- if-else 문 안에 다른 if-else 문이 포함될 수 있다.



# Q & A

