

Day 3 : 변수와 자료형 & 간단한 제어

(Lab : 무작정 따라해보세요~)

변수를 이용한 비교 & 반복

그리고 거북이(Turtle)^^

Yun-Jeong Choi(cris2@ewha.ac.kr)

Dept. of Computer Science & Engineering

Ewha Womans Univ.

□ 기본이론

- 변수와 자료형
- 변수에 입력받기 : input
- 비교와 반복문

오늘은 마지막 시간으로

‘변수와 자료형’ & ‘비교문과 반복문’에 대해

두루두루 가볍게 알아봅니다

□ Lab

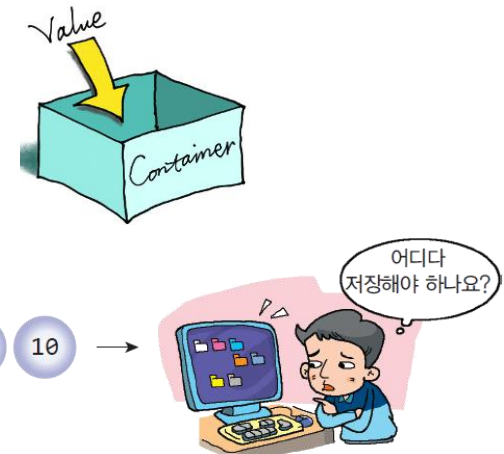
- 계산기 : 두 수를 입력받아 $+$, $-$, $*$, $/$
- 거북이로 그리기 : 크기를 입력받아 집 그리기

□ 학습활동

□ Supplement

□ 변수(variable) : 변수(variable)는 값을 저장하는 상자!

- 저장하는 능력이 있어요
- 우리말로 '그릇'이라고 생각하면 편합니다.
- 입력받은 데이터나, 보관하고 싶은 데이터를 저장할 때 사용되지요?



□ 자료형

- 수와 문자
- 소수점이 없으면 정수, 있으면 실수!
- "나" '를 이용하면 문자열

자료형	예
정수	..., -2, -1, 0, 1, 2, ...
실수	3.2, 3.14, 0.12
문자열	'Hello World!', "123"

100 : 정수

"100", '100' : 문자열

```
>>> print(100+200)
300
>>> print("100"+"200")
100200
```

□ 파이썬에서 변수를 생성하려면

- 그냥 씁니다.

- x : 아무것도 들어있지 않는 상자, 이름은 x

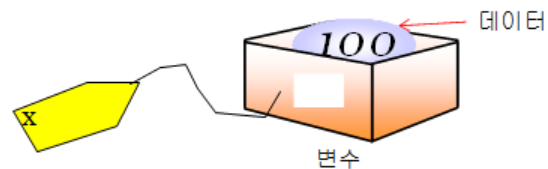
- $x=100$ 의 뜻은 **x 라는 상자에 숫자 100을 넣는다!**

□ x 와 100은 같다!가 아니라 100을 x 에 저장한다는 의미입니다.

프로그래밍 언어에서 “=” 는 ‘같다’의 등호의 의미가 아니라 대입기호!

프로그래밍 언어에서 등호는 “==” 을 씁니다.

```
>>> x
>>> x = 100
>>>
```



- 생성된 변수에는 얼마든지 다른 값을 저장할 수 있고, 여러 개의 변수도 만들 수 있어요

```
>>> x = 100
```

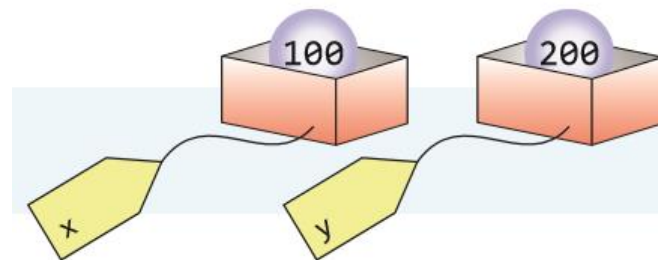
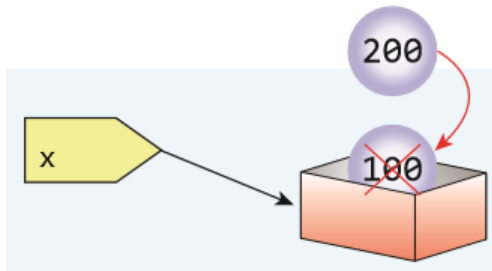
```
>>> x = 200
```

```
>>> print(x)
```

```
200
```

```
>>> x = 100
```

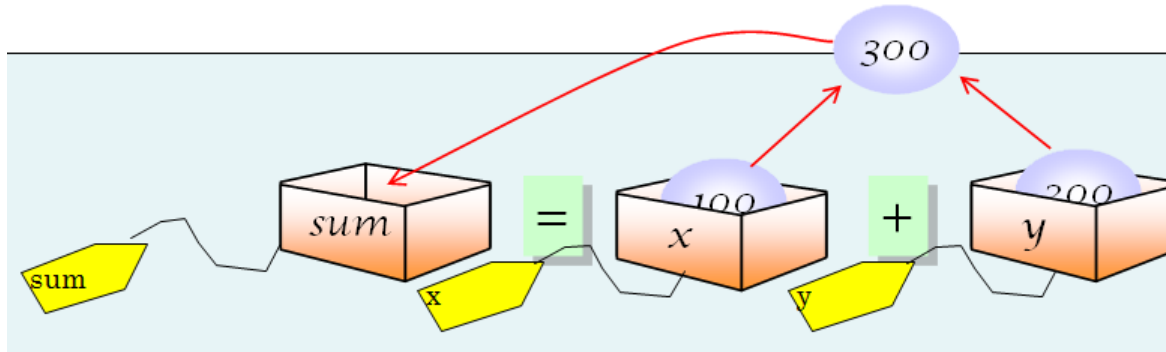
```
>>> y = 200
```



변수를 이용한 계산

```
>>> x = 100
>>> y = 200
>>> sum = x + y
>>> print(sum)
300
```

연산자	기호	사용례	결괏값
덧셈	+	$7 + 4$	11
뺄셈	-	$7 - 4$	3
곱셈	*	$7 * 4$	28
나눗셈	//	$7 // 4$	1
나눗셈	/	$7 / 4$	1.75
나머지	%	$7 \% 4$	3



문자열도 변수에 저장할 수 있다!

- 파이썬의 변수에는 정수뿐만 아니라 문자열도 저장할 수 있다.

```
>>> name = "이은지"
```

```
>>> address = "이화여자대학교"
```



```
>>> print(name)
```

```
이은지
```

```
>>> print(address)
```

```
이화여자대학교
```

잠깐 Quiz: 출력되는 결과는?



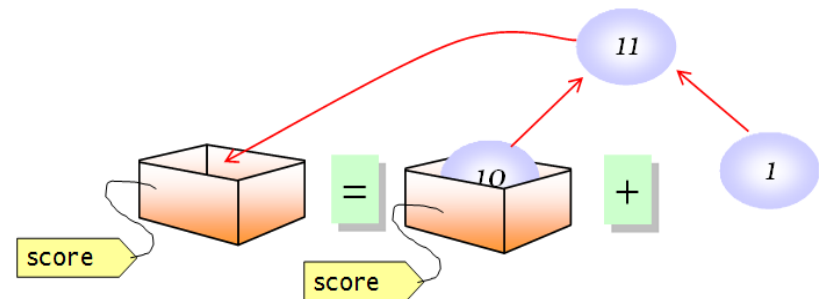
도전문제

무엇이 출력될까?

```
>>> x = 7  
>>> y = 6  
>>> print(x + y)
```

```
>>> x = '7'  
>>> y = '6'  
>>> print(x + y)
```

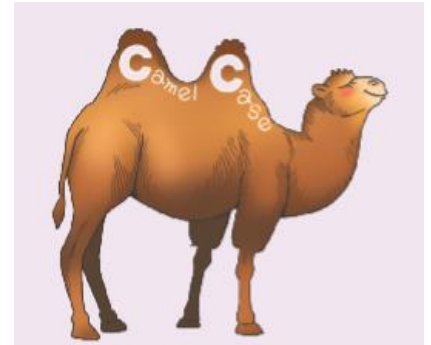
```
>>score = 10  
>>score = score + 1  
>>print(score)
```



식별자와 낙타체

sum	# 영문 알파벳 문자로 시작
_count	# 밑줄 문자로 시작할 수 있다.
number_of_pictures	# 중간에 밑줄 문자를 넣을 수 있다.
King3	# 맨 처음이 아니라면 숫자도 넣을 수 있다.

2nd_base (X)	# 숫자로 시작할 수 없다.
money# (X)	# #과 같은 기호는 사용할 수 없다.



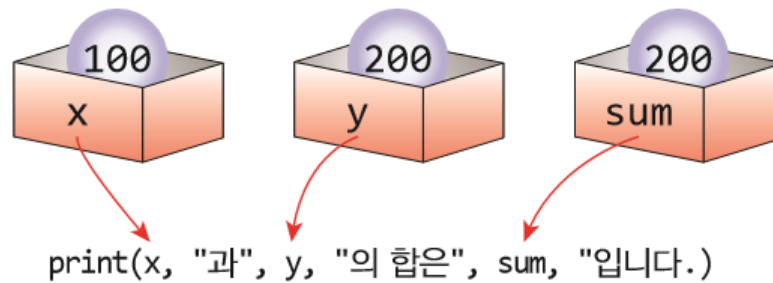
□ 낙타체란

- 변수의 첫 글자는 소문자로, 나머지 단어의 첫 글자는 대문자로 적는 방법!
- myNewCar처럼 첫 'm'은 소문자로, 나머지 단어들의 첫 글자는 대문자로 표기.
- 보통 변수와 함수는 소문자로 시작하고 클래스 이름은 대문자로 시작합니다.

여러 값을 함께 출력하려면

```
x = 100  
y = 200  
sum = x + y  
print(x, "과", y, "의 합은", sum, "입니다.")
```

100 과 200 의 합은 300 입니다.



입력받기 : 정수 입력받기

- `input ("메시지");`
- 모든 입력은 문자열로 처리하므로 키보드로 입력된 "1234"는 문자열 "1234"
- 숫자로 저장하려면 형변환 함수 `int` 와 함께 써 줄 수 있습니다.

input() 사용법

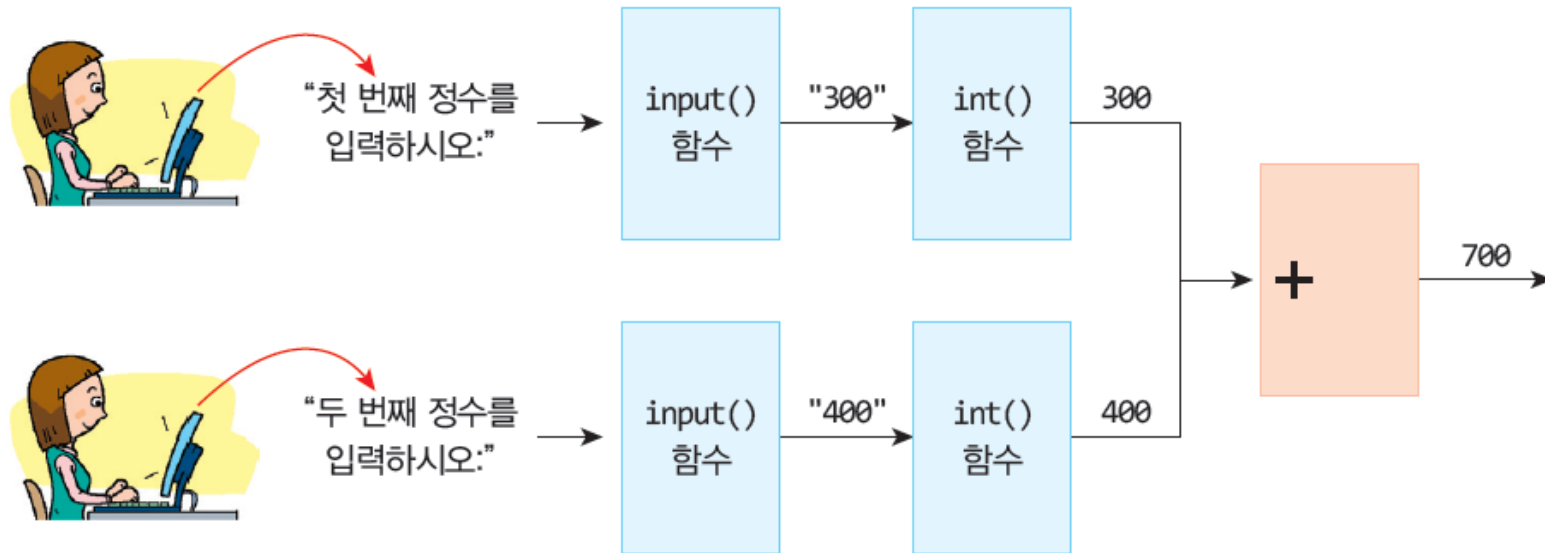
변수

사용자가 입력한 문자열을 숫자로 변환한다.

```
x = int(input("첫 번째 정수를 입력하시오: "))
```

안내 메시지를 출력하고 사용자가 입력한 값을 문자열 형태로 받는다.

두 수를 입력받아 저장하려면

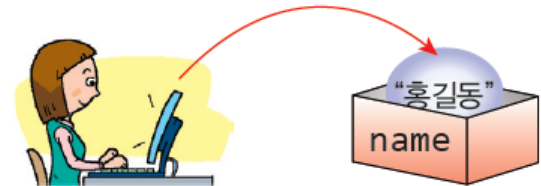


```
x = int(input("첫 번째 정수를 입력하시오: "))  
y = int(input("두 번째 정수를 입력하시오: "))  
sum = x + y  
print(x, "과", y, "의 합은", sum, "입니다.")
```

첫 번째 정수를 입력하시오: 300
두 번째 정수를 입력하시오: 400
100 과 200 의 합은 300 입니다.

문자열 입력받기

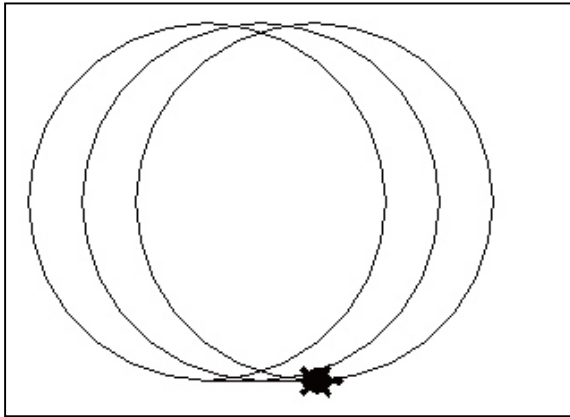
```
name = input("이름을 입력하시오: ")  
print(name, "씨, 안녕하세요?")  
print("파이썬에 오신 것을 환영합니다.")
```



이름을 입력하시오: 홍길동
홍길동 씨, 안녕하세요?
파이썬에 오신 것을 환영합니다.

변수는 어디에 유용할까?

- 터틀 그래픽을 사용하여 반지름이 100픽셀인 3개의 원을 그려봅시다.



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

radius = 100

t.circle(radius) # 반지름이 100인 원이 그려 진다.
t.fd(30) # fd(forward)
t.circle(radius) # 반지름이 100인 원이 그려 진다.
t.fd(30)
t.circle(radius) # 반지름이 100인 원이 그려 진다.
```

3개라서 다행!!

만약에 원 300개를 그리라고 하면 변수보다도 반복처리가 가능한 명령어가 필요하겠죠?^^

변수는 어디에 유용할까?

- 원의 반지름을 50으로 변경하여서 다시 그려야 한다!!
- Code B처럼 원의 반지름이 변수로 표현되었기 때문에 변수만 변경하면 된다.
- 변수를 사용하지 않았다면, Code A처럼 사용한 곳마다 찾아서 수정해야 합니다.

Code A

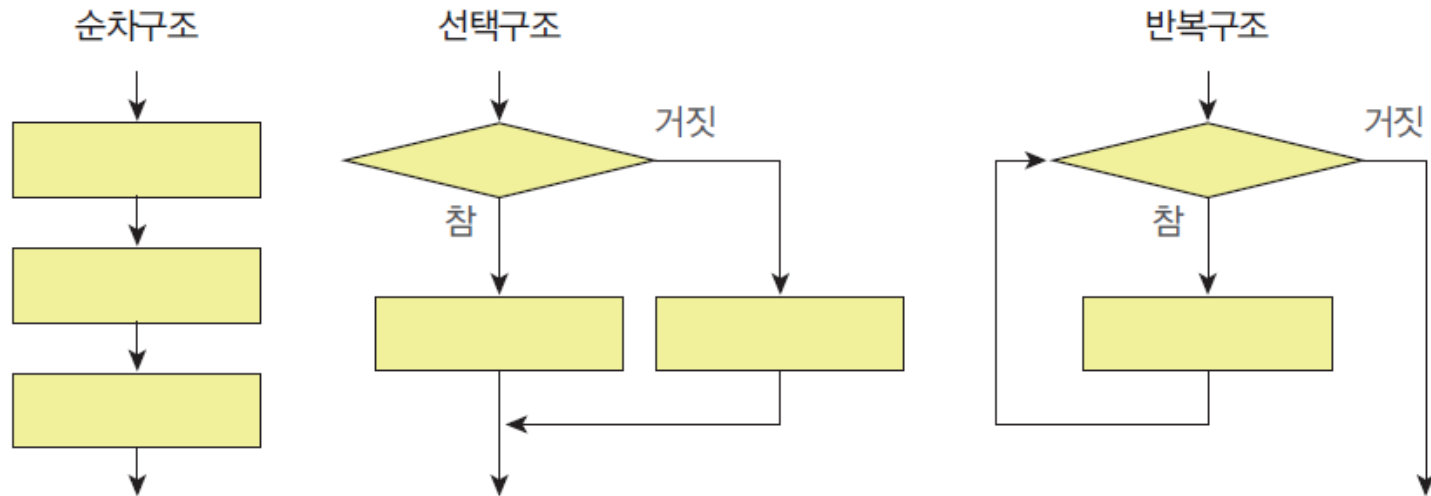
```
t.circle(100)
t.fd(30)
t.circle(100)
t.fd(30)
t.circle(100)
t.fd(30)
t.circle(100)
t.fd(30)
t.circle(100)
```

Code B

```
radius = 100
t.circle(radius)
t.fd(30)
t.circle(radius)
t.fd(30)
t.circle(radius)
t.fd(30)
t.circle(radius)
t.fd(30)
t.circle(radius)
```

제어문 : 비교와 반복문

- 순차적으로 실행하는 명령문
- 조건에 따라 비교하는 비교문
- 그리고 반복문

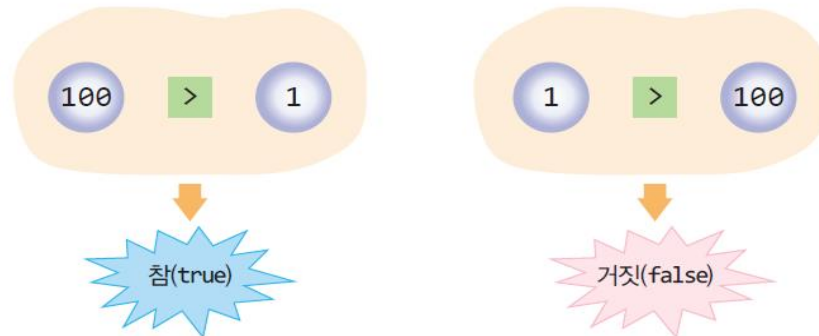


비교 : 관계 연산자와 논리연산자

□ 관계 연산자(relational operator)

- 비교에 사용하는 연산자
- 두 개의 피연산자를 비교하는 연산자
- 결과값은 참 또는 거짓!

연산	의미
$x == y$	x와 y가 같은가?
$x != y$	x와 y가 다른가?
$x > y$	x가 y보다 큰가?
$x < y$	x가 y보다 작은가?
$x \geq y$	x가 y보다 크거나 같은가?
$x \leq y$	x가 y보다 작거나 같은가?



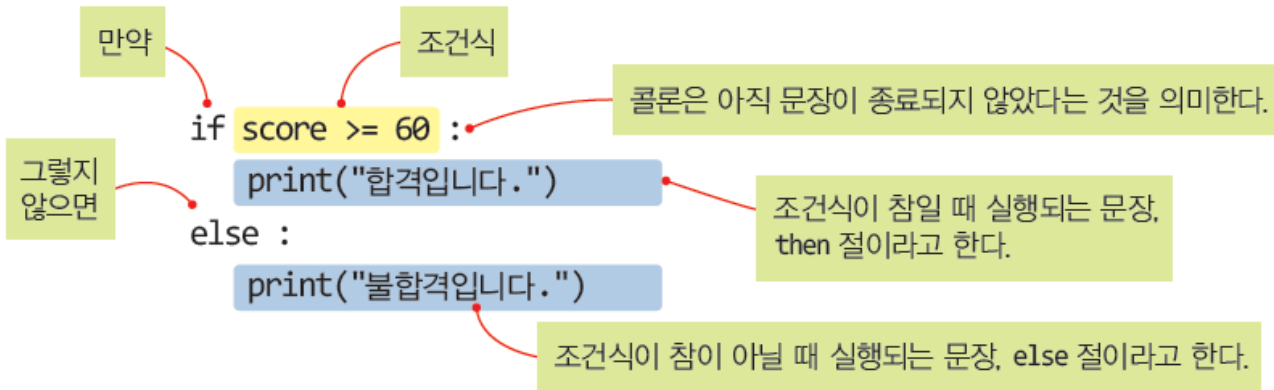
□ 논리 연산자

- and , or , not

연산	의미
$x \text{ and } y$	AND 연산, x와 y가 모두 참이면 참, 그렇지 않으면 거짓
$x \text{ or } y$	OR 연산, x나 y중에서 하나만 참이면 참, 모두 거짓이면 거짓
$\text{not } x$	NOT 연산, x가 참이면 거짓, x가 거짓이면 참

비교할 때 사용하는 문장 : if-else 문

if-else 문



```
num = int(input("정수를 입력하시오: "))  
if num % 2 == 0 :  
    print("짝수입니다.")  
else:  
    print("홀수입니다.")
```

정수를 입력하시오: 10
짝수입니다.

비교 : 관계 연산자와 논리연산자 사용하기

조건 1

조건 2

나이가 10살 이상이고, 그리고 키가 140 cm 이상이면
→ 놀이기구를 탈 수 있다.

(나이가 10살 이상이다) and (키가 140 cm 이상이다)
→ 놀이기구를 탈 수 있다.

age >= 10 and height >=140



만약 조건이 참인 경우에 여러 개의 문장이 실행되어야 한다면?

□ 블록문을 이용합니다

블록문

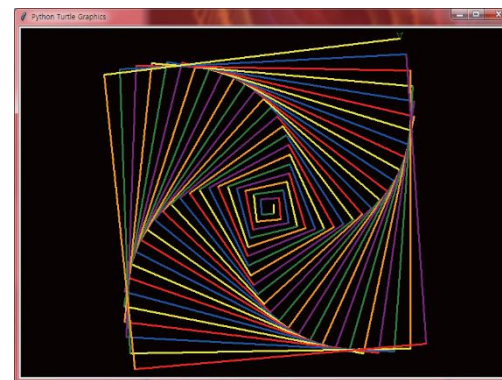
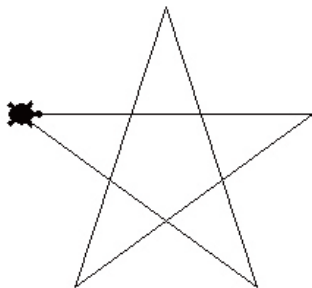
```
if score > 90 :
```

```
    print("합격입니다.")
```

```
    print("장학금도 받을 수 있습니다.")
```

: 다음라인부터
들여쓰기를 넣어 블록을 설정합
니다.

- 동일한 문장을 여러 번 반복시키는 구조
- 같은 라인이나 블록을 여러 번 반복할 때 유용합니다.
- 이미 사용해봤던 for 와 while 을 사용하여
 - 지정한 횟수만큼
 - 조건만큼
 - 무한루프



왜 반복이 중요한가? 5번 반복하기

- 전광판에 '방문을 환영합니다!'를 5번 출력한다고 하자.

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



5번 째이야!!!!

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

복사해서 붙여넣기

만약 1000번 반복해야 한다면?

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 ?
이건 좀...



□ 반복 구조를 사용한다!

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

0부터 999까지
1000번 반복시키는 구조

횟수를 제어하는 반복문 for : range() 함수를 사용한다

for 문

```
for 변수 in range(종료 값) :
```

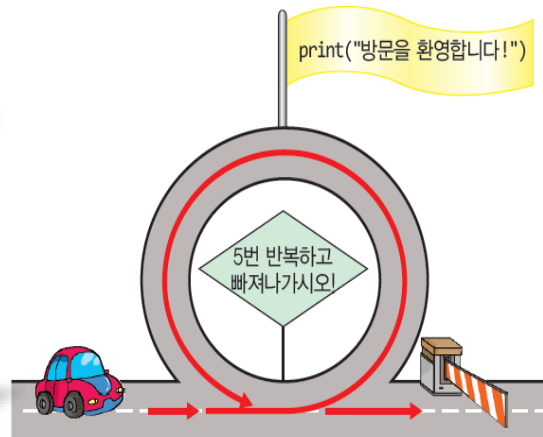
문장

0에서 (종료 값-1)까지의 숫자를 반환한다.

반복되는 문장으로
들어쓰기 하여야 한다.

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

```
#range 대신 [ ] 에 범위를 줄 수도 있어요  
for i in [1, 2, 3, 4, 5] :  
    print("방문을 환영합니다.")
```



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

range() 함수

range() 함수

시작값이다.

종료 값이지만 stop은
포함되지 않는다.

한 번에 증가되는 값이다.

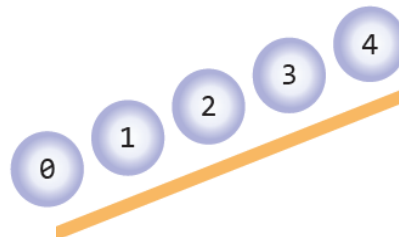
`range(start=0, stop, step=1)`

□ 만약 1부터 시작하여서 5까지 반복하고 싶다면?

1 2 3 4 5

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

`print(i)` 한 줄에 하나씩 출력합니다.
`end=""`를 넣어 줄바꾸기를 하지 않도록 합니다.



range() 함수

□ 그럼 10 부터 1까지 출력하는 방법도 있을까요?

10 9 8 7 6 5 4 3 2 1

```
for i in range(10, 0, -1):  
    print(i, end=" ")
```

10부터 시작하여 0이 되기 전까지,
1을 감소하면서 출력합니다.

반복문을 이용하여 구구단을 출력해보자.

□ 우선 9단만!!

```
for i in [1, 2, 3, 4, 5, 6, 7, 8, 9]:  
    print("9 *", i, "=", 9*i)
```

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

2단부터 9단까지 모두 출력하고 싶다면?

>>> 반복할 블록을 반복하기!

```
for dan in range(10) :
```

```
    for i in [1, 2, 3, 4, 5, 6, 7, 8, 9] :  
        print( dan, "*", i, "=", dan*i)
```

그런데, range(10)은 0부터 9까지..
구구단에 0단은 없어도 되는데..??

range() 함수로 제어한다.

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

1단부터 9단까지 구구단

반복문안에 반복문

```
for dan in range(10) :
```

```
    for i in [1, 2, 3, 4, 5,6,7,8, 9] :
```

```
        print( dan, "*", i, "=", dan*i)
```

```
for dan in range(1,10,1) :
```

```
    for i in [1, 2, 3, 4, 5,6,7,8,9]:
```

```
        print( dan, "*", i, "=", dan*i)
```

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



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

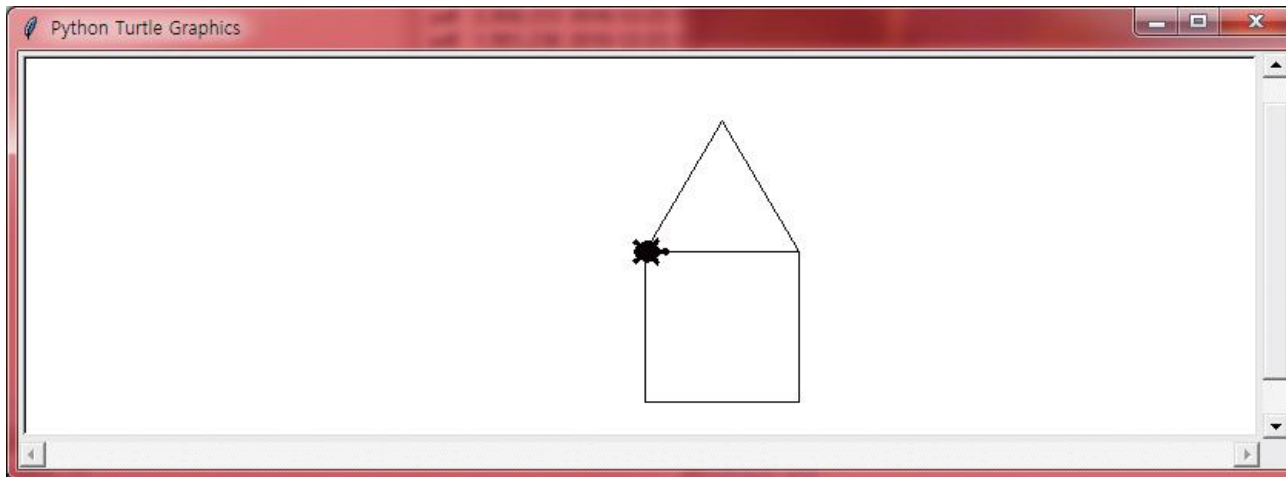
Lab : 가벼운 실습

- 크기를 입력받아 집그리기
- 거북이에게 입력받아보기
- 오른쪽 왼쪽 거북이제어하기
- 1부터 10까지 누적합 구해보기
 - 별 그리기

Lab#1: 집그리기

- 사용자로부터 집의 크기를 입력 받아서 크기에 맞는 집을 그려보자.

집의 크기는 얼마로 할까요? 100

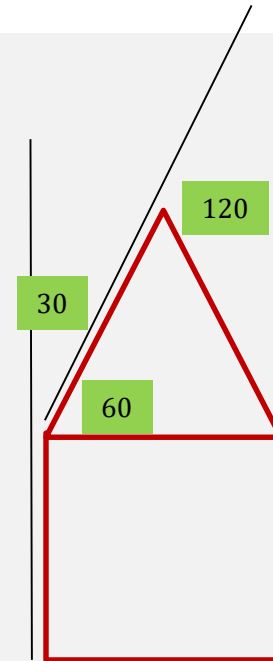


```
import turtle
t = turtle.Turtle()
t.shape("turtle")

size = int(input("집의 크기는 얼마로 할까요? "))

t.forward(size) # size 만큼 거북이를 전진시킨다.
t.right(90) # 거북이를 오른쪽으로 90도 회전시킨다.
t.forward(size)
t.right(90)
t.forward(size)

t.right(30)
t.forward(size)
t.right(120)
t.forward(size)
```

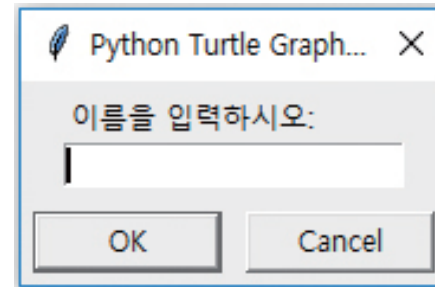


Lab#2: 거북이에게 입력받고 출력하기

□ 터틀 그래픽에서 문자열을 입력받는 방법

```
s = turtle.textinput("", "이름을 입력하시오: ")
```

```
t.write("안녕하세요? 터틀 인사드립니다.")
```



Solution

```
import turtle
t = turtle.Turtle()
t.shape("turtle")
s = turtle.textinput("", "이름을 입력하시오: ")
t.write("안녕하세요?" + s + "씨, 터틀 인사드립니다.")

t.left(90)
t.forward(100)
t.left(90)
t.forward(100)
t.left(90)
t.forward(100)
t.left(90)
t.forward(100)
```



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

변수와 가벼운
반복문을 적용해봅니다.

angle=90
dist=100

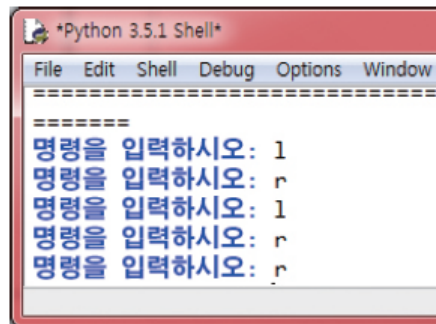
```
s = turtle.textinput("", "이름을 입력하시오: ")
t.write("안녕하세요?" + s + "씨, 터틀 인사드립니다.")
```

for i in range(4) :
 t.left(angle)
 t.forward(dist)

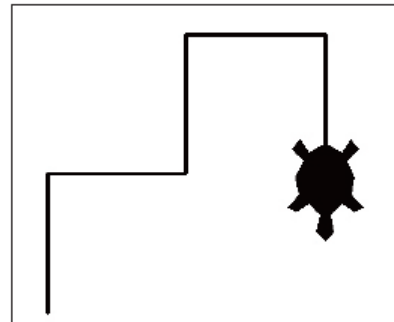
Lab#3: 오른쪽 왼쪽 거북이 제어하기

□ 파이썬 셸에서

- left의 "l"을 입력하면 거북이가 왼쪽으로 100픽셀
- right의 "r"을 입력하면 거북이가 오른쪽으로 100픽셀 이동하는 프로그램



```
*Python 3.5.1 Shell*
File Edit Shell Debug Options Window
=====
=====
명령을 입력하시오: l
명령을 입력하시오: r
명령을 입력하시오: l
명령을 입력하시오: r
명령을 입력하시오: r
```



- 아직 학습하지 않았지만 다음과 같은 코드를 사용하면 무한 반복!
- 들여쓰기로 무한 반복할 블록을 지정합니다.

```
while True:
```

```
    ...
```

```
    ...
```

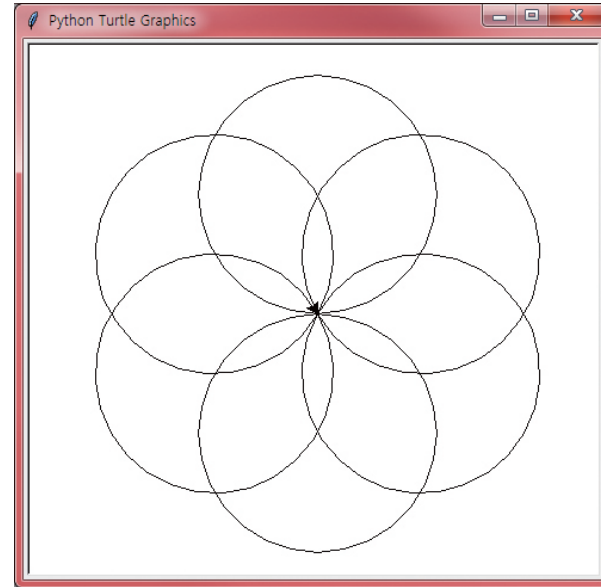
```
    ...
```

```
import turtle
t = turtle.Turtle()      # 거북이를 만든다.
t.width(3)               # 거북이가 그리는 선의 두께를 3으로 한다.
t.shape("turtle")        # 커서의 모양을 거북이로 한다.
t.shapesize(3, 3)        # 거북이를 3배 확대한다.

# 무한 루프
while True:
    command = input("명령을 입력하시오: ")
    if command == "l": # 사용자가 "l"을 입력하였으면
        t.left(90)
        t.forward(100)
    if command == "r": # 사용자가 "r"을 입력하였으면
        t.right(90)
        t.forward(100)
```

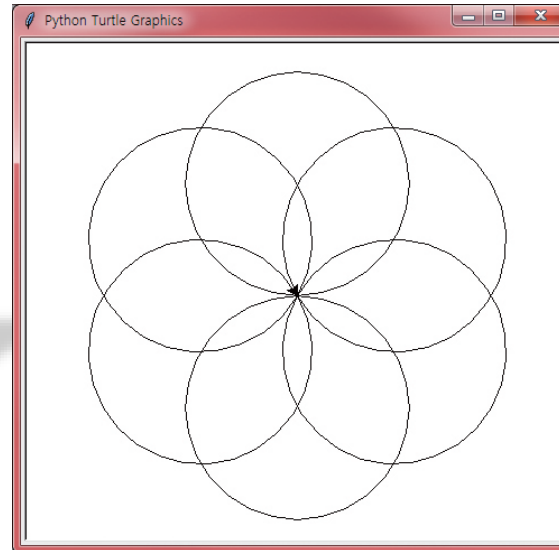
Lab #4: 거북이로 원그리기

- 6개의 원 그리기
- 60도씩 방향을 회전하면서
- 원을 그린다.
- 변수와 반복문을 활용해보자~!



Solution

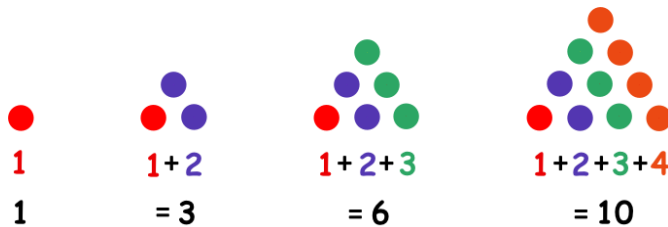
```
import turtle
t = turtle.Turtle()
n = 6
for count in range(n):
    t.circle(100)
    t.left(360/n)
```



Lab #5 : 1부터 10까지 합을 계산해보자

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

합계는 55



Solution : for 와 while 문으로

```
count = 1
sum = 0
while count <= 10 :
    sum = sum + count
    count = count + 1

print("합계는", sum)
```

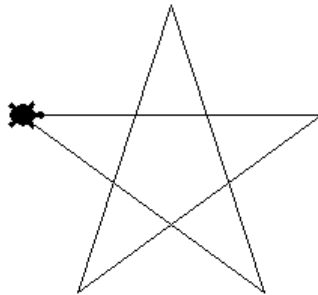
```
count = 1
sum = 0
for count in range (1,11,1) :
    sum = sum + count

print("합계는", sum)
```

Lab #6: 별 그리기

□ 반복문을 사용하여 별을 그려보자.

■ 거북이를 50픽셀만큼 전진시키고 오른쪽으로 144도 회전하기를 5번반복!



Solution

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

Day 3 학습활동 : 제어를 이용하는 프로그램 아무거나 ^^

비교문과 반복문을 적용하여 다양한 프로그램을 만들어보세요~

수업만족도 조사를 위한 설문참여하기

Supplement

Tic-tac-tok 게임 만들기

GUI 를 이용한 계산기 프로그램

**사이버캠퍼스 게시판을 참고하세요

수고하셨습니다!!