

# 파이썬 프로그래밍 입문

## 자료형

최 윤 정

[cris.lecture@gmail.com](mailto:cris.lecture@gmail.com)

# 오늘은

## □ 자료의 종류에 대해 알아봅니다.

- 수 : 정수와 실수
- 문자열

## □ 문자열 연산자

## □ 그리고 출력 : +, \*, %

## □ 실습

- 거북이 : 대화식 처리
- 여러 형태의 자료형을 입력받아 저장하고 출력하기
- 리스트 미리보기 : 내 친구리스트 만들기



# 파이썬의 자료형

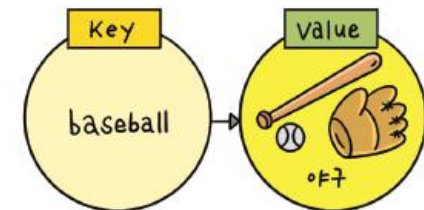
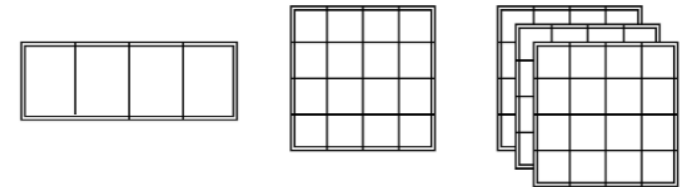
표 3-1 파이썬의 자료형

자료형	설명
bool	<ul style="list-style-type: none"> <li>• 설명: 이진 자료형으로, True 또는 False 값을 가진다.</li> <li>• 예시: flag=True, cond=False</li> </ul>
int, float, complex	<ul style="list-style-type: none"> <li>• 설명: 수치 자료형으로, 정수·실수·복소수의 값을 가진다.</li> <li>• 예시: a=123, b=3.14, c=3-2j</li> </ul>
str	<ul style="list-style-type: none"> <li>• 설명: 문자열 자료형으로, 다양한 문자열들을 가진다.</li> <li>• 예시: name='Kim', addr="Seoul", sex='''male''', room="33"</li> </ul>
list	<ul style="list-style-type: none"> <li>• 설명: 순서가 있는 자료형으로, 다양한 객체들을 멤버로 가질 수 있는 자료형이다. []를 사용한다.</li> <li>• 예시: member=["Seoul", 'Kim', 22.4, 30, True]</li> </ul>
tuple	<ul style="list-style-type: none"> <li>• 설명: list와 같으나 내용의 변경이 허용되지 않는 자료형이다. ()를 사용한다.</li> <li>• 예시: member=("Seoul", 'Kim', 22.4, 30, True)</li> </ul>
dict	<ul style="list-style-type: none"> <li>• 설명: 순서가 없는 자료형으로, 키와 값으로 이루어진 자료를 저장하는 자료형이다. 값은 중복될 수 있으나, 키는 중복될 수 없다. {}를 사용한다.</li> <li>• 예시: age={'Kim':22, 'Park':21, 'Lee':22, 'Jung':20}</li> </ul>
set	<ul style="list-style-type: none"> <li>• 설명: 순서가 없는 자료형으로, 값의 중복을 허용하지 않는다. {}를 사용한다.</li> <li>• 예시: grade={1,2,3,4}, item={'hand', 3, 3.14, "park", True}</li> </ul>

참(True), 거짓(False)을 저장한다

수치값을 저장한다.

문자열을 저장한다

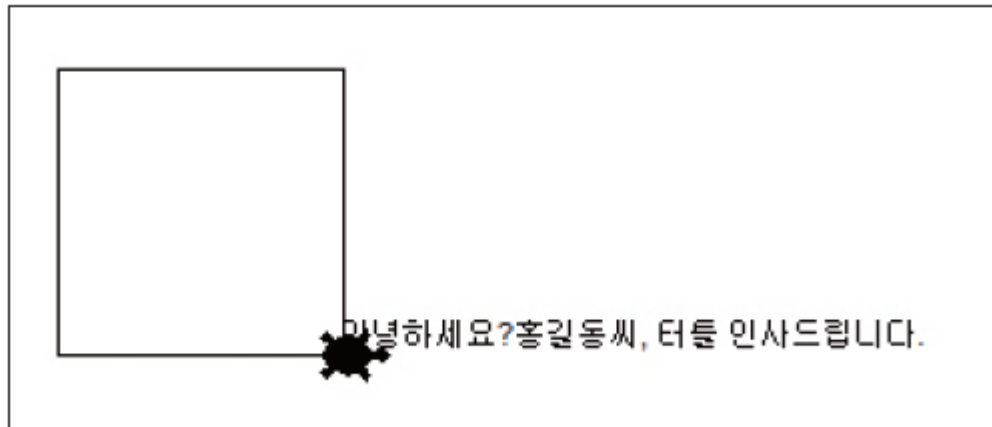


# 실습내용

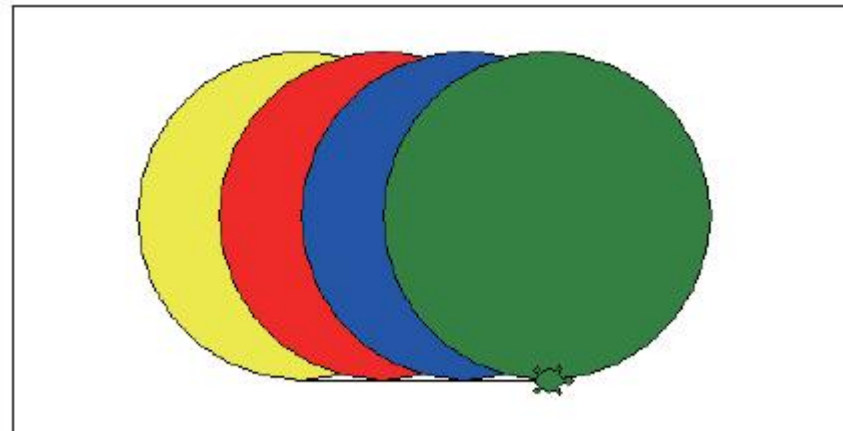
## □ 자료형 다루기

- 숫자 → 문자 : `p=str(123)`
- 문자 → 숫자 : `q=int(p)`

## □ 거북이



```
>>> n=123    # 정수 123이 저장된 n
>>> s = str(n) # n의 값을 문자열로 변환
>>> s         # 출력결과는 문자열
'123'
>>> p=int(s)  # s에 저장된 문자열을 수로 변환
>>> p         # 출력결과는 수
123
>>> int(s) + 100 # 산술계산으로 확인한다.
223
```



# 파이썬에서 사용할 수 있는 자료의 종류

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

자료형을 저장하는 변수의 형이 비교적 자유롭다.

(다른 언어들은

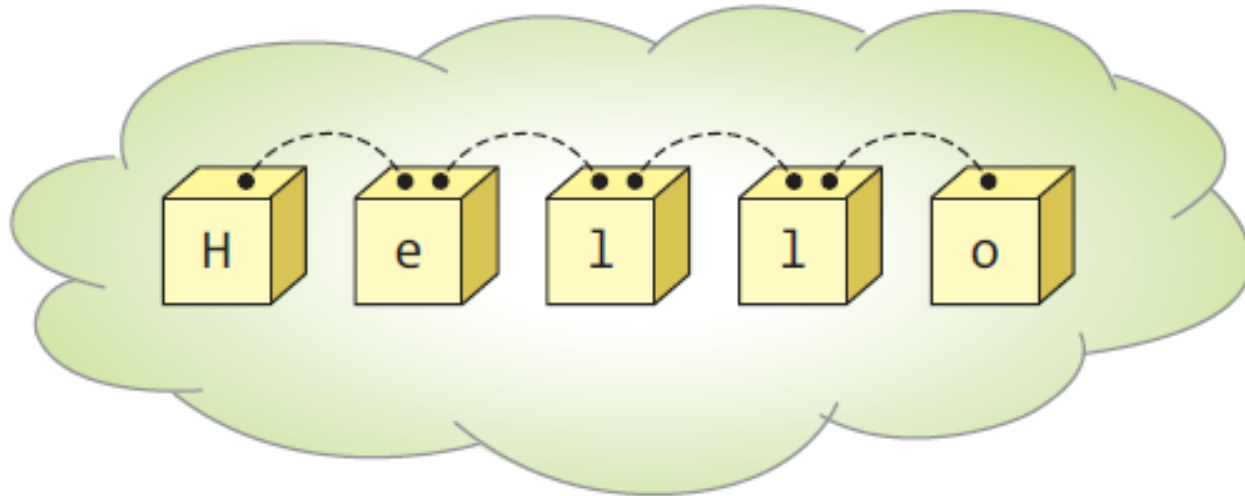
정수형 변수에 문자를 넣으면 에러!)

```
x = 10
print("x =", x)
x = 3.14
print("x =", x)
x = "Hello World!"
print("x =", x)
```

```
x = 10
x = 3.14
x = Hello World!
```

# 문자열

- 문자메시지, 이메일 등 인간에게 텍스트(text) 정보가 중요하다.
- 수 정보 와 함께 컴퓨터를 이용한 텍스트의 처리도 무척 중요하다
- 문자열(string)은 문자들의 나열(sequence of characters)이다.

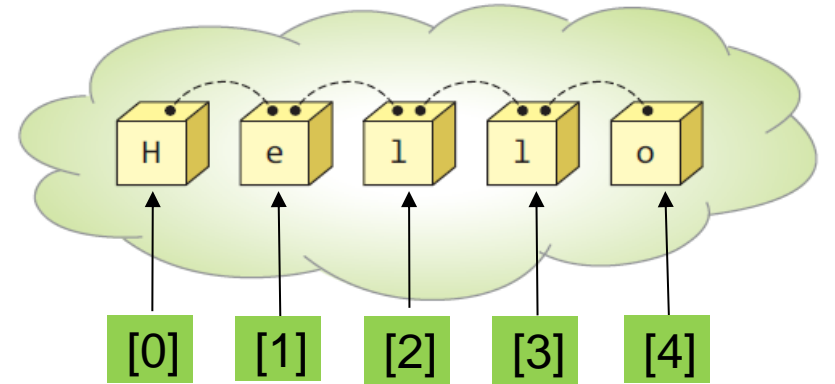


# 문자열을 만드는 방법

- 큰따옴표
- 작은 따옴표
- 짝을 맞추어 사용합니다.

- "abc"
- 'aaa'
- "aaa' #에러
- 'aaa" #에러

```
>>> 'Hello'
'Hello'
>>> msg = "Hello"
>>> msg
'Hello'
>>> print(msg)
Hello
>>> len(msg) # msg의 문자길이(length)
5
```



#문자가 여러개인 문자열

>>>msg[0] # msg문자열의 0번요소

'H'

>>>msg[1] #msg문자열의 1번요소

'e'

# 문법적인 오류

- 큰따옴표(“)로 시작했다가 작은따옴표(')로 끝내면 문법적인 오류이다.

```
>>> msg = "Hello'
```

```
SyntaxError: EOL while scanning string literal
```





# 100과 "100"의 차이

- 100 : 정수
- "100", '100' : 문자열
- 수의 계산은 산술연산
  - +, -, \*, /, //, %, \*\*
- 문자와 숫자의 연산은 연결하기
  - +, \*

```
>>> print(100+200)
```

```
300
```

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

```
100200
```

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

```
Traceback (most recent call last):
```

```
File "<pyshell#57>", line 1, in <module>
```

```
print("100" + 200)
```

```
TypeError: can only concatenate str (not "int") to str
```

+ : 피연산자가 숫자이면 '산술계산'으로,  
문자열이면 '연결하기'를 수행합니다.

# 문자열과 숫자간의 변환 : 문자열 → 숫자

- `int()`: 문자열을 정수로 변환
- `float()`: 문자열을 실수로 변환

```
t = input("정수를 입력하시오: ")
x = int(t)
t = input("정수를 입력하시오: ")
y = int(t)
print(x+y)
```

한 줄로 쓰는 것도 가능!

```
x = int(input("정수를 입력하시오: "))
```

```
정수를 입력하시오: 100
정수를 입력하시오: 200
300
```

숫자로 읽을 수 있는 경우만 가능합니다.  
"abcd123" 은 안되요!

# 문자열과 숫자간의 변환 : 숫자 → 문자열

- print 문 안에서 , 와 + 을 종종 사용합니다.
- 다음 코드에 오류가 발생하는 이유는 무엇일까요?



```
>>> print("나는 현재 " + 1 + "학년이다.")
```

Traceback (most recent call last):

File "<pyshell#1> ", line 1, in <module>

```
print("나는 현재 " + 1 + "학년이다.")
```

TypeError: Can't convert 'int' object to str implicitly

문자열과 숫자를 + 로 연결하여 발생한 문제로  
int형 변수를 str로 변환할 수 없다는 의미입니다.

# 문자열과 숫자의 변환 : 숫자 → 문자열

## □ str() 함수 사용

```
>>> print('나는 현재 ' + str(1) + ' 학년이다.')
```

나는 현재 1학년이다.

```
>>> print('원주율은 ' + str(3.14) + '입니다.')
```

원주율은 3.14입니다.

```
>>> 학년 = 1
```

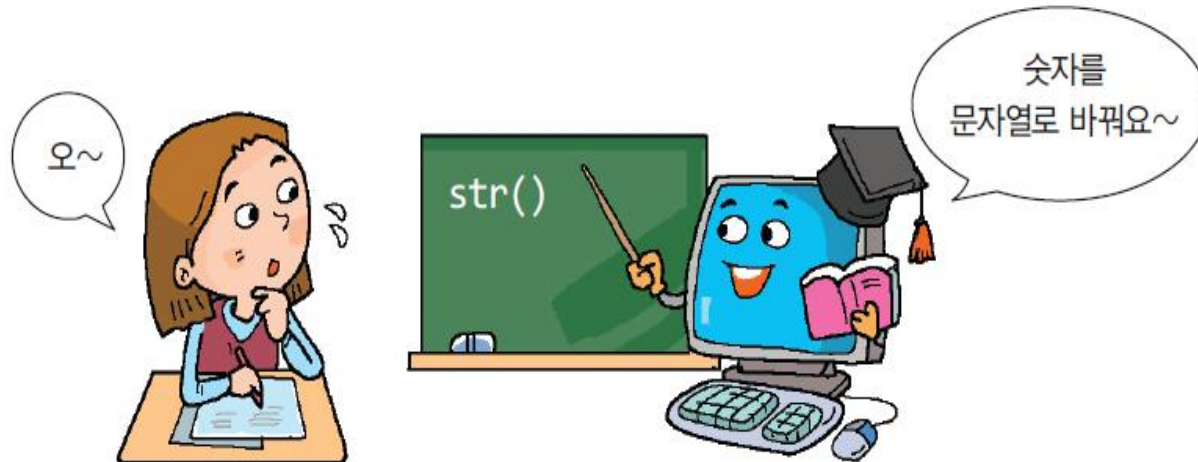
```
>>> print('나는 현재 ' + str(학년) + ' 학년이다.')
```

나는 현재 1학년이다.

```
>>> 원주율 = 3.14
```

```
>>> print(' 원주율은 ' + str(원주율) + ' 입니다.')
```

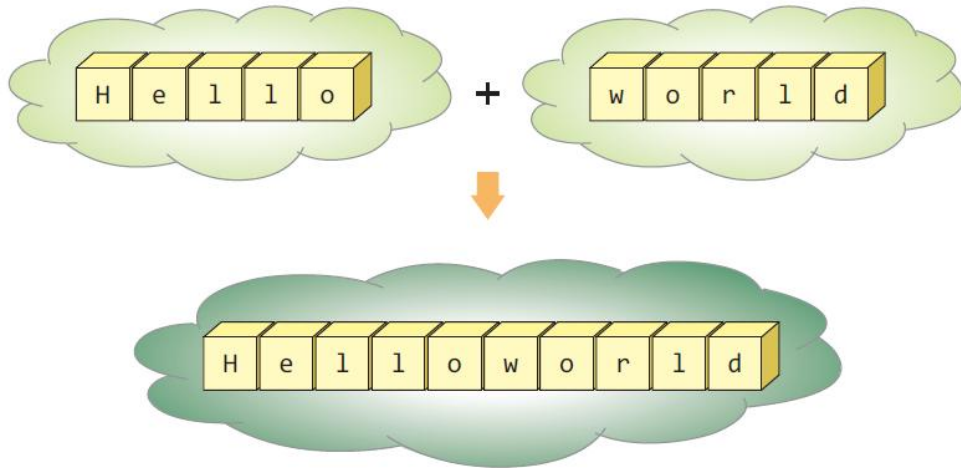
원주율은 3.14입니다.



# 문자열 연결 : + , \*

- 2개의 문자열을 합치려면 ➔ + 연산자

```
>>> 'Hello ' + 'World!'
'Hello World!'
```



- 문자열 연결을 반복하려면 ➔ \* 연산자

```
>>> message = " Congratulations!"
>>> message*3
Congratulations!Congratulations!Congratulations!
>>> m = message*3
>>> m
Congratulations!Congratulations!Congratulations!

>>> print("="*30)
=====
```

# 출력 문자열에 변수값 포함하기 : f-string { }

- 문자열에 변수의 값을 삽입하여 출력하고 싶으면 ➔ { }괄로 이용
- print 문 내의 " "에는 , 와 + 가 없다는 것에 유의합니다.

## □ 예)

- num 이라는 변수에 정수가 들어있을 때
- `print(f" 변수에 들어있는 값 :{num}")`
- num1은 정수, num2는 실수, num3은 문자열 일 때
- `print(f" num1: {num}, num2: {num2}, num3: {num3}")`

num=123 이 저장되었을 때  
변수에 들어있는 값 : 123 이 출력됨

num1=123  
num2=3.14  
num3="abcd" 이 들어있을 때  
  
num1 : 123, num2: 3.14, num3: abc 가 출력됨

# 출력 문자열에 변수값 포함하기 : %s %d %f

- 문자열에 변수의 값을 삽입하여 출력하고 싶으면 ➔ %기호 사용
- print 문 내의 " "에는 , 와 + 가 없다는 것에 유의합니다.
- 문자열은 %s, 정수는 %d, 실수는 %f를 씁니다.
- 예) print(" 문자열과 형식 " %(변수이름))

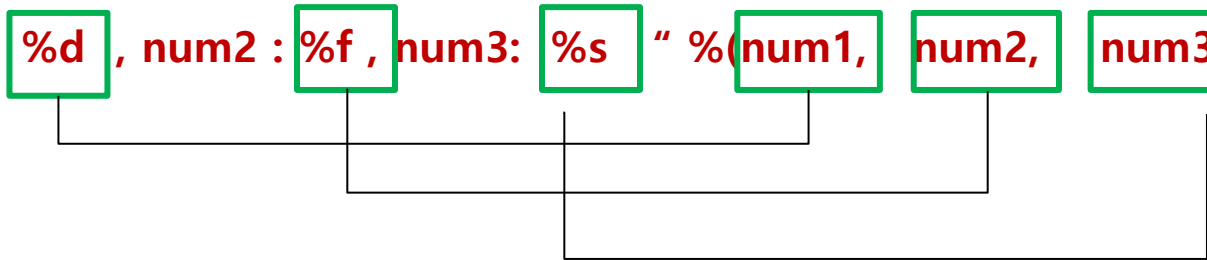
- num 이라는 변수에 정수가 들어있을 때

- **print(" 변수에 들어있는 값 : %d " % (num))**



- num1은 정수, num2는 실수, num3은 문자열 일 때

- **print(" num1: %d , num2 : %f , num3: %s " %(num1, num2, num3) )**



#하나 더! 소수의 자리수제어하기

#1.234567을 반올림하여 소수점이하 한자리로 출력

```
print(" %.1f" % 1.234567)
```

결과는? 1.2

# 출력 문자열에 변수값 포함하기 : %s %d %f

```
price = 10000  
print("상품의 가격은 %d원입니다." % price)
```

결과 : 상품의 가격은 10000원입니다.

```
name = "냉모밀"  
price = 500  
print("맛있는 %s의 가격은 %d원 입니다. " % (name, price))
```

```
반지름 = 6  
넓이 = 반지름 * 반지름 * 3.14  
print("반지름이 %d 인 원의 넓이: %f" % (반지름, 넓이))
```

```
number = 1.23456789  
print("반올림하여 소수 두자리까지 : %.2f " % number)
```

```
===== RESTART: C:\Users\User\AppData\Local\Programs\Python\Python38-32\python.exe  
맛있는 냉모밀의 가격은 500원 입니다.  
반지름이 6 인 원의 넓이: 113.040000  
반올림하여 소수 두자리까지 : 1.23  
^^^
```



# 개별 문자 추출 : 인덱스 사용

- 문자열에서 개별 문자들을 추출하려면 : 인덱스라는 번호를 사용한다.
- 인덱스 번호는 0부터 시작

- `s[0]` 은 'M'
- `s[11]` 은 'n'

0	1	2	3	4	5	[6:10]				10	11
M	o	n	t	y		P	y	t	h	o	n
-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
[-12:-7]											

`s = "Monty Python"` 일 때, 아래 코드의 결과는?

```
print(s[6:10])
```

```
ss = s[6:10]
```

```
print(ss+"!@#")
```

```
print(s[-12:-7])
```

"Pyth"

"Pyth!@#"

"Monty"

# 특수 문자열

표 3-7 파이썬의 주요 특수문자

문자	설명	문자	설명
\	다음 줄과 연속임을 나타냄	\"	“ 문자
\\	\\문자	\n	줄을 바꿈
\'	' 문자	\t	탭만큼 띄운다

```
>>> print('Hi \
... Python')           # \를 사용하여 두 개의 줄을 하나로 취급
Hi Python
>>> print('Hi \n Python') # 줄을 바꾸어 출력
Hi
Python
>>> print('Hi \t\t Python') # 탭만큼 띄우고 출력
Hi          Python
```

# 리스트 미리보기: 간단 버전

- 리스트(list): 여러 개의 자료들을 모아서 하나의 묶음으로 저장하는 것
- 자료형이 달라도 묶어 사용할 수 있지만, 목적에 따라 저장할 자료를 정리하여 사용하는 것이 좋습니다.

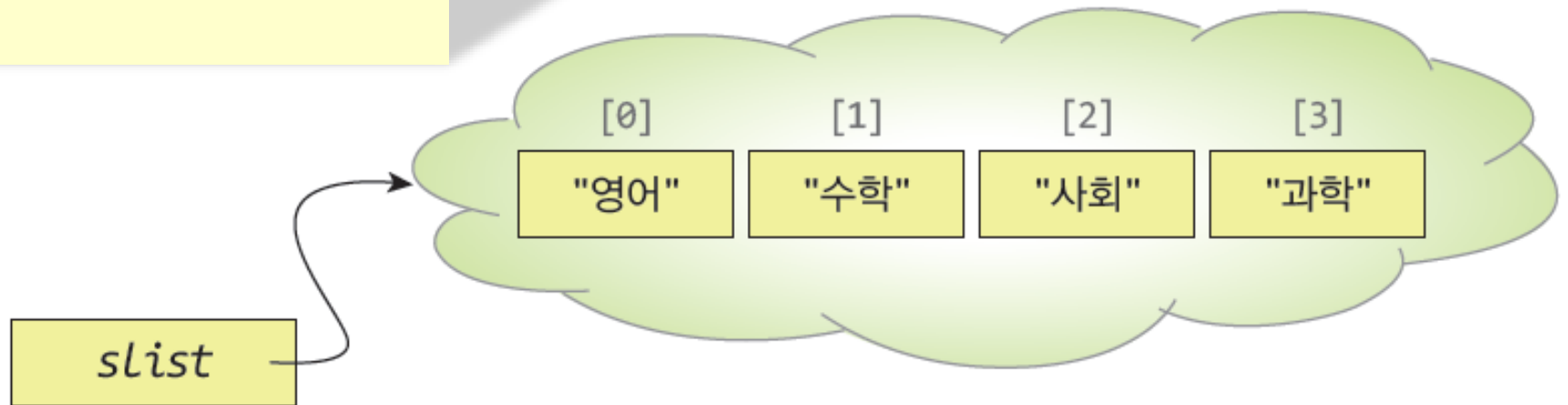
# 문자열과 정수, 실수를 한꺼번에 저장할 수는 있지만 구분하여 사용하는 것이 좋습니다!

```
>>> info=["울랄라" , 20, 3.14 ]
```

```
>>> info
```

```
['울랄라', 20, 3.14]
```

```
slist = [ '영어', '수학', '사회', '과학' ]
```



# 리스트 요소 접근하기

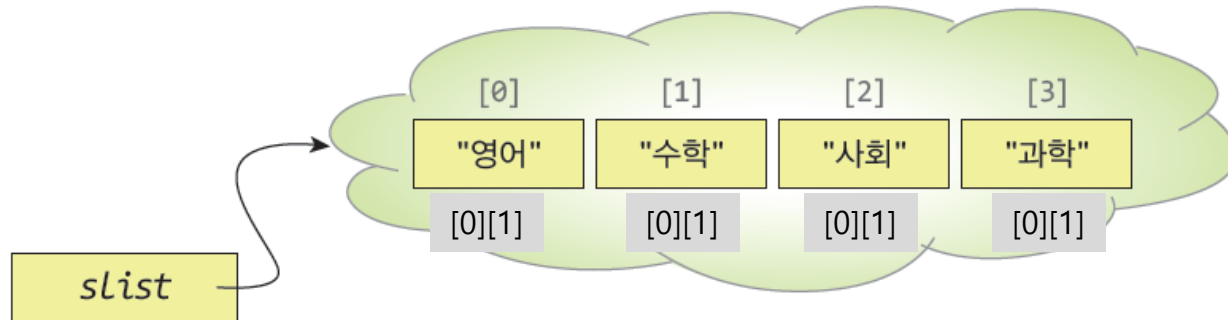
```
slist = [ '영어', '수학', '사회', '과학' ]
```

```
slist[0] # 리스트의 첫번째 요소
```

```
slist[0][0] #리스트의 첫번째요소의 첫번째 요소
```

```
slist[0][1]
```

영어  
영  
어



자세한 내용은 8장에서 튜플과 함께 다루겠습니다.

# 리스트에 항목을 추가하기 : append

- 공백 리스트를 생성한 후에 코드로 리스트에 값을 추가하는 것
- 추가 : `append(값)`, 삽입 : `insert(인덱스, 값)`
- 삭제 : `remove(값)` , 정렬 : `sort`

```
list = [ ]    # 리스트 생성
list.append(1) # 요소 추가
list.append(2)
list.append(6)
list.append(3) # list =[1,2,6,3] 과 같은 결과
```

[1, 2, 6, 3]

```
list.insert(2,3) #2번에 3 추가
list.insert(3,4) # 3번에 4 추가
print(list)
list.sort()
print(list)
```

[1, 2, 3, 4, 6, 3]

[1, 2, 3, 3, 4, 6]

# Lab : 가벼운 실습

---

Turtle : 대화식 프로그램과 리스트 색 적용하기

다양한 형태의 자료형 입력

친구리스트 만들기

# Lab: 거북이와 이야기하자

□ 터틀 그래픽에서 사용자의 이름을 받아 다음과 같이 출력해보자.

□ 입력과 출력

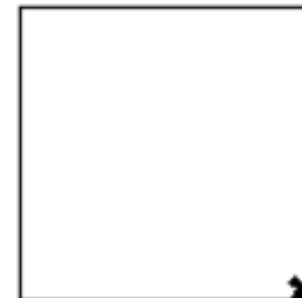
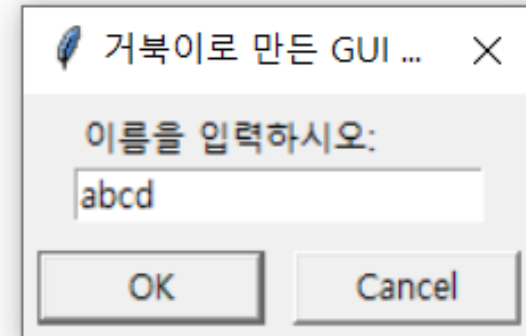
```
import turtle
t = turtle.Turtle()
s = turtle.textinput("거북이로 만든 GUI 입력창 ", "이름을 입력하시오: ")
```

#네모를 그린 후에

어떤 코드를 넣어야할까요?

```
t.write("안녕하세요?" + s + "씨, 터틀 인사드립니다.")
```

안녕하세요?  
○○○ 씨  
터틀 인사드립니다.



안녕하세요?abcd씨, 터틀 인사드립니다.

# Solution

```
t.left(90)
```

```
t.fd(100)
```

```
t.left(90)
```

```
t.fd(100)
```

```
t.left(90)
```

```
t.fd(100)
```

```
t.left(90)
```

```
t.fd(100)
```

도전 : 지난주에 배운 변수와 간단  
반복문을 적용해볼까요?

```
angle=90
```

```
dist=100
```

```
for i in range(4) :
```

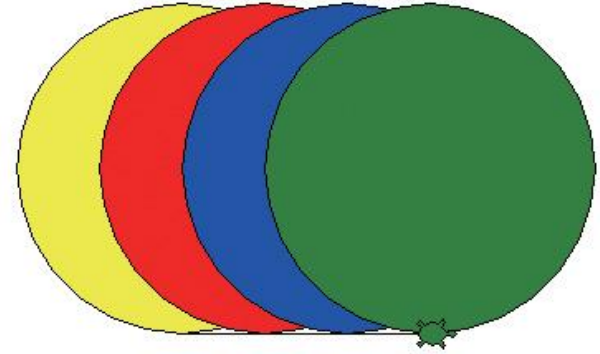
```
    t.left(angle)
```

```
    t.fd(dist)
```



# Lab: 리스트에 저장된 색상으로 원그리기

- 리스트에 색상을 문자열로 저장하였다가
- 하나씩 꺼내어 거북이의 채우기 색상으로 설정하고
- 원을 그려보자



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

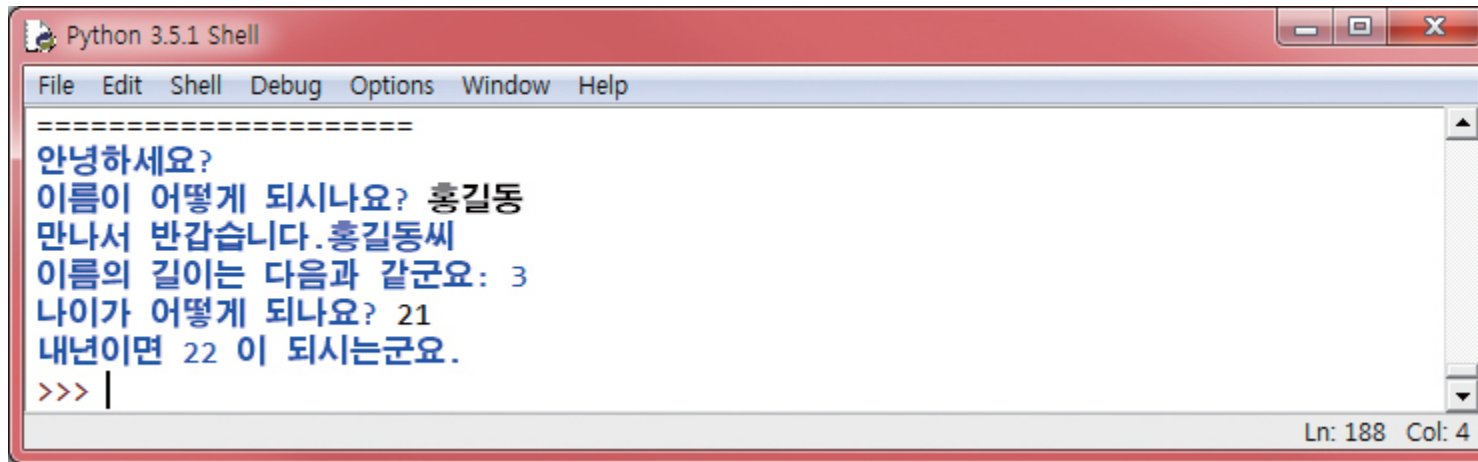
```
# 리스트를 사용하여 색상을 문자열로 저장한다.
color_list = [ "yellow", "red", "blue", "green" ]
```

```
t.fillcolor(color_list[0]) # 채우기 색상을 설정한다.
t.begin_fill() # 채우기를 시작한다.
t.circle(100) # 속이 채워진 원이 그려진다.
t.end_fill() # 채우기를 종료한다.
```

```
t.forward(50)
t.fillcolor(color_list[1]) # 채우기 색상을 설정한다.
t.begin_fill() # 채우기를 시작한다.
t.circle(100) # 속이 채워진 원이 그려진다.
t.end_fill() # 채우기를 종료한다.
```

# Lab: 친근하게 대화하는 프로그램

- 변수를 사용하여 사용자의 이름과 나이를 문자열 형태로 기억하고
- 출력할 때 사용하는 프로그램을 작성해 보자. 마치 이야기 하는 것 처럼~



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
=====
안녕하세요?
이름이 어떻게 되시나요? 홍길동
만나서 반갑습니다. 홍길동씨
이름의 길이는 다음과 같군요: 3
나이가 어떻게 되나요? 21
내년이면 22 이 되시는군요.
>>> |
```

Ln: 188 Col: 4

- 문자열의 길이를 계산할 때는 `len(s)` : 단, 공백도 문자
- 문자열을 다음 줄이 아니라 바로 연결하여 출력할 때는 `end`를 `""`로 지정한다.
  - `print( 'aa', end="" )`

# Solution

## □ 다양한 방법을 사용하여 출력할 수 있다

```
# +와 , 를 이용할 때
print('안녕하세요?')
name = input('이름이 어떻게 되시나요? ')

print('만나서 반갑습니다.' + name + "씨")
print('이름의 길이는 다음과 같군요:', end=' ')
print(len(name))

age = int(input("나이가 어떻게 되나요? "))
print("내년이면", str(age+1), "이 되시는군요.")
```

```
# f - string을 이용할 때
print('안녕하세요?')
name = input('이름이 어떻게 되시나요? ')

print(f'만나서 반갑습니다.{name}씨')
print(f'이름의 길이는 {len(name)}글자 군요')
age = int(input("나이가 어떻게 되나요? "))
Print(f " 내년이면 {age+1} 살이 되시는군요.")
```

# Lab: 친구들의 리스트 생성하기

- 제일 친한 친구 5명의 이름을 리스트에 저장하고
- 출력하는 프로그램을 작성하자.
- 5번의 입력문을 써주거나
- 5번을 반복하도록 만들거나 ^^

친구의 이름을 입력하세요 : 다니엘

친구의 이름을 입력하세요 : 공유

친구의 이름을 입력하세요 : 현빈

친구의 이름을 입력하세요 : 유빈

친구의 이름을 입력하세요 : 원빈

내 친구들 : ['다니엘','공유','현빈','유빈','원빈 ']

# Solution

```
friend_list = [ ]
```

```
friend = input("친구의 이름을 입력하세요: ")  
friend_list.append(friend)
```

```
friend = input("친구의 이름을 입력하세요: ")  
friend_list.append(friend)
```

```
friend = input("친구의 이름을 입력하세요: ")  
friend_list.append(friend)
```

```
friend = input("친구의 이름을 입력하세요: ")  
friend_list.append(friend)
```

```
friend = input("친구의 이름을 입력하세요: ")  
friend_list.append(friend)
```

```
print("내 친구들 :", friend_list)
```

```
friend_list = [ ]
```

```
n = 5
```

```
for i in range(n):  
    friend = input("친구의 이름을 입력하세요: ")  
    friend_list.append(friend)
```

```
print("내 친구들 :", friend_list)
```

# 이번 장 정리

---

- 파이썬에서 기본적인 자료형은 정수, 실수, 문자열이다.
- 문자열은 큰따옴표("...")나 작은 따옴표('...')를 사용할 수 있다.
- 문자열을 정수로 변경하려면 `int()`를 사용한다.
- 문자열을 실수로 변경하려면 `float()`를 사용한다.
- 정수나 실수를 문자열로 변경하려면 `str()`을 사용한다.
- 문자열과 문자열을 합치려면 `+` 연산자를 사용한다.
- 문자열을 반복하려면 `*` 연산자를 사용한다.
- `input()`은 사용자로부터 문자열을 받아서 반환한다.
- `\n`은 줄 바꿈을 나타내는 특수 문자열이다.
- `리스트`는 자료들을 모아서 저장할 수 있다.

수고하셨습니다.