

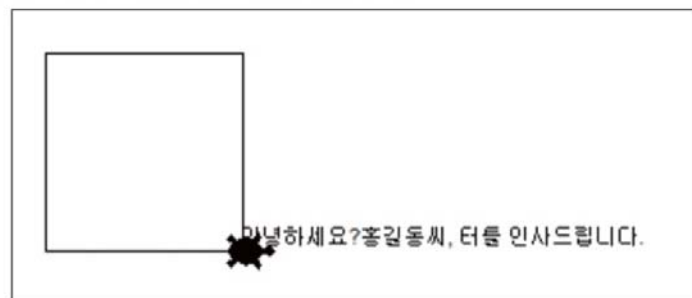
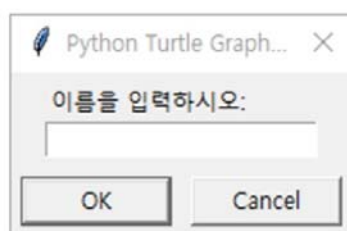
## 4. 자료의 종류

### 학습 내용

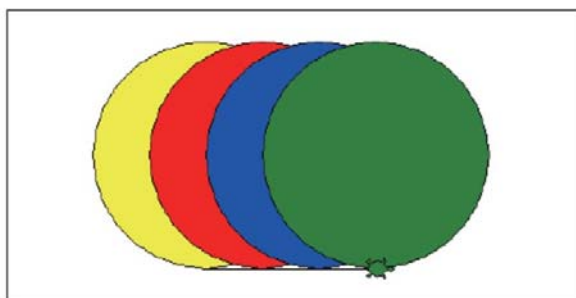
- 파이썬에서 사용 가능한 자료형을 살펴봅니다.
- 정수, 실수, 문자열을 구별할 수 있습니다.
- 정수를 문자열로, 문자열을 정수로 변환할 수 있습니다.
- 문자열에 관련된 연산을 살펴봅니다.
- 특수 문자열에 대하여 살펴봅니다.
- 리스트에 대하여 간단하게 살펴봅니다.

### 이번 장에서 만들 프로그램

- 터틀 그래픽의 거북이와 인사하는 프로그램을 작성해 보자.



- 여러 개의 색상을 리스트에 저장하였다가 하나씩 꺼내서 원들을 그려 보자



## 이번 장에서 만들 프로그램

- 변수를 사용하여 사용자의 이름과 나이를 문자열 형태로 기억했다가 친근하게 대화하는 프로그램을 작성해보자.

```
Python 3.7.2 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.
>>>
```

## 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"     |



## 파이썬과 자료형

- 파이썬에서는 변수에 어떤 종류의 자료도 저장할 수 있다

```
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!
```

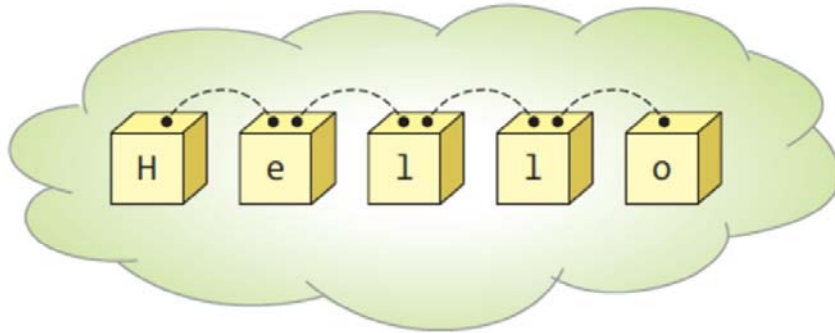
## 문자열

- 컴퓨터에게는 숫자가 중요하지만 인간에게는 텍스트(text)가 중요하다.
  - (예) 문자 메시지, 인터넷 도메인 이름
- 컴퓨터를 이용한 텍스트의 처리도 무척 중요하다.



## 문자열

- 문자열(string)은 문자들의 나열(sequence of characters)이다.



## 문자열을 만드는 방법

- 큰따옴표
- 작은 따옴표

```
>>> "Hello"
'Hello'

>>> msg = "Hello"
>>> msg
'Hello'
>>> print(msg)
Hello
```

## 문법적인 오류

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

```
>>> msg = "Hello'  
SyntaxError: EOL while scanning string literal
```



## 왜 큰따옴표와 작은따옴표를 동시에 사용할까?

```
>>> message="철수가 "안녕"이라고 말했습니다."  
SyntaxError: invalid syntax
```

"...안에 ..."가 있어서 파이썬 인터프리터가 문자열의 시작과 끝을 구분할 수 없다는 의미이다.

"철수가 "안녕"이라고 말했습니다."

문자열      문자열

```
>>> message = "철수가 '안녕'이라고 말했습니다."  
>>> print(message)  
철수가 '안녕'이라고 말했습니다.  
>>> message = '철수가 "안녕"이라고 말했습니다.'  
>>> print(message)  
철수가 "안녕"이라고 말했습니다.
```

## 100과 "100"의 차이

- 100 -> 정수
- "100", '100' -> 문자열

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

- 100+200을 하면 (정수+정수) 형태가 되어서 덧셈이 가능하다.
- "100"+"200"은 텍스트와 텍스트끼리 합하는 것이기 때문에 2개의 텍스트가 붙어 버린다.

## 문자열을 숫자로 변환하기

- int( 문자열 )
  - 문자열을 정수로 변환
- float( 문자열 )
  - 문자열을 실수로 변환

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

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



## 문자열을 숫자로 변환하기

### ●type() 함수

#### ●변수의 자료형을 알아보는 함수

```
>>> type("Hello World!")
<class 'str'>
>>> type(3.2)
<class 'float'>
>>> type(17)
<class 'int'>
>>> x = 3.2
>>> type(x)
<class 'float'>
```

## 문자열 -> 숫자 (오류가 발생할 수 있다.)

### ●주의

```
>>> t = "1.0"
>>> x = int(t)
Traceback (most recent call last):
  File "<pyshell#9>", line 1, in <module>
    x = int(t)
ValueError: invalid literal for int() with base 10: '1.0'
>>> x = float(t)
>>> x
1.0
>>> t = "1"
>>> x = float(t)
>>> x
1.0
```

## 숫자->문자열

### ●다음 코드에 오류가 발생하는 이유는?

```
>>> print('나는 현재 ' + 21 + '살이다.')
```

Traceback (most recent call last):

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

print('나는 현재 ' + 21 + '살이다.')

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

문자열과 숫자를 합칠 수 없는 의미입니다.



## 숫자->문자열

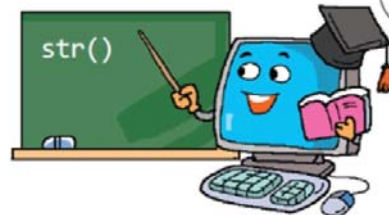
### ●str() 함수 사용

```
>>> print('나는 현재 ' + str(21) + '살이다.')
```

나는 현재 21살이다.

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

원주율은 3.14입니다.



숫자를  
문자열로 바꿔요~

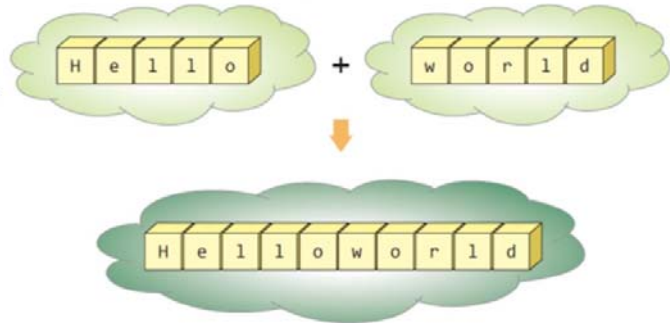


## 문자열 연결

- 2개의 문자열을 연결하려면 ➔ + 연산자

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

```
>>> first_name="길동"
>>> last_name="홍"
>>> name = last_name+first_name
>>> print(name)
홍길동
```



## 문자열 반복

- 문자열을 반복하려면 ➔ \* 연산자

```
>>> message = " Congratulations!"
>>> print(message*3)
Congratulations!Congratulations!Congratulations!
>>> msg = " Congratulations!" * 3
>>> print(msg)
Congratulations!Congratulations!Congratulations!
```

```
>>> print("="*50)
=====
```

## 문자열에 변수값 포함

- 문자열에 변수의 값을 삽입하여 출력하고 싶으면 → %기호 사용

```
>>> price = 10000
>>> print("상품의 가격은 %s원입니다." % price)
상품의 가격은 10000원입니다.
```

- 문자열 포맷 코드

| 코드 | 설명                    |
|----|-----------------------|
| %s | 문자열 (String)          |
| %c | 문자 1개(character)      |
| %d | 정수 (Integer)          |
| %f | 부동소수 (floating-point) |
| %o | 8진수                   |
| %x | 16진수                  |
| %% | Literal % (문자 % 자체)   |

## 문자열에 변수값 포함

- 변수 ch 에 65를 저장하고, 다양한 포맷으로 프린트하여 보자

```
>>> ch = 65
>>> print("문자출력 %c" % (ch))
문자출력 A
>>> print("문자출력 %d" % (ch))
문자출력 65
>>> print("문자출력 %s" % (ch))
문자출력 65
>>> print("문자출력 %f" % (ch))
문자출력 65.000000
>>> print("문자출력 %o" % (ch))
문자출력 101
>>> print("문자출력 %x" % (ch))
문자출력 41
```

## 문자열에 변수값 포함

- `ord()` : 문자를 ASCII 코드로 변환
- `chr()` : ASCII 코드를 문자로 변환

```
>>> ord("가")
44032
>>> hex(ord("가"))
'0xac00'
>>> chr(0xac00)
'가'
>>> chr(44032)
'가'
>>> ord("가나다")
Traceback (most recent call last):
  File "<pyshell#44>", line 1, in <module>
    ord("가나다")
TypeError: ord() expected a character, but string of length 3 found
>>> chr(0xac00, 0xac00)
Traceback (most recent call last):
  File "<pyshell#45>", line 1, in <module>
    chr(0xac00, 0xac00)
TypeError: chr() takes exactly one argument (2 given)
```

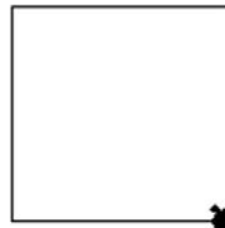
## 문자열에 변수값 포함

### ● ASCII 코드

| 숫자     | 알파벳    |        |         |         | 기능 키                   | 방향 키   |
|--------|--------|--------|---------|---------|------------------------|--------|
| 0 = 48 | A = 65 | N = 78 | a = 97  | n = 110 | Backspace = 8          | ← = 37 |
| 1 = 49 | B = 66 | O = 79 | b = 98  | o = 111 | Tab = 9                | ↑ = 38 |
| 2 = 50 | C = 67 | P = 80 | c = 99  | p = 112 | Enter = [CR=13, LF=10] | → = 39 |
| 3 = 51 | D = 68 | Q = 81 | d = 100 | q = 113 | Shift = 16             | ↓ = 40 |
| 4 = 52 | E = 69 | R = 82 | e = 101 | r = 114 | Ctrl = 17              |        |
| 5 = 53 | F = 70 | S = 83 | f = 102 | s = 115 | Alt = 18               |        |
| 6 = 54 | G = 71 | T = 84 | g = 103 | t = 116 | ESC = 27               |        |
| 7 = 55 | H = 72 | U = 85 | h = 104 | u = 117 | Space = 32             |        |
| 8 = 56 | I = 73 | V = 86 | i = 105 | v = 118 | PAGEUP = 33            |        |
| 9 = 57 | J = 74 | W = 87 | j = 106 | w = 119 | PAGEDN = 34            |        |
|        | K = 75 | X = 88 | k = 107 | x = 120 |                        |        |
|        | L = 76 | Y = 89 | l = 108 | y = 121 |                        |        |
|        | M = 77 | Z = 90 | m = 109 | z = 122 |                        |        |

## 실습 - P0401 : 거북이와 인사해보자.

- 터틀 그래픽에서 사용자의 이름을 받아서 다음과 같이 출력해보자.

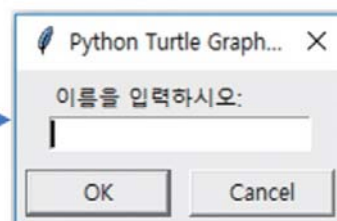


안녕하세요?김창성씨, 터틀 인사드립니다.

## 실습 - P0401 : 거북이와 인사해보자.

- 터틀 그래픽에서 문자열을 입력받는 방법

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

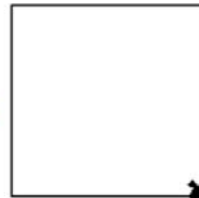


- textinput( 타이틀, 표시문자열 )

## 실습 - P0401 : 거북이와 인사해보자.

### ●터틀 그래픽에서 문자열을 출력하는 방법

```
t.write("안녕하세요? 터틀 인사드립니다.")
```



안녕하세요?김창성씨, 터틀 인사드립니다.

## 실습 - P0401 : 거북이와 인사해보자.

```
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)
```

- 이름을 출력할 때 % 기호를 사용하여 출력하는 프로그램을 작성하여 제출하세요.

안녕하세요?  
○○○ 씨  
터틀 인사드립니다.

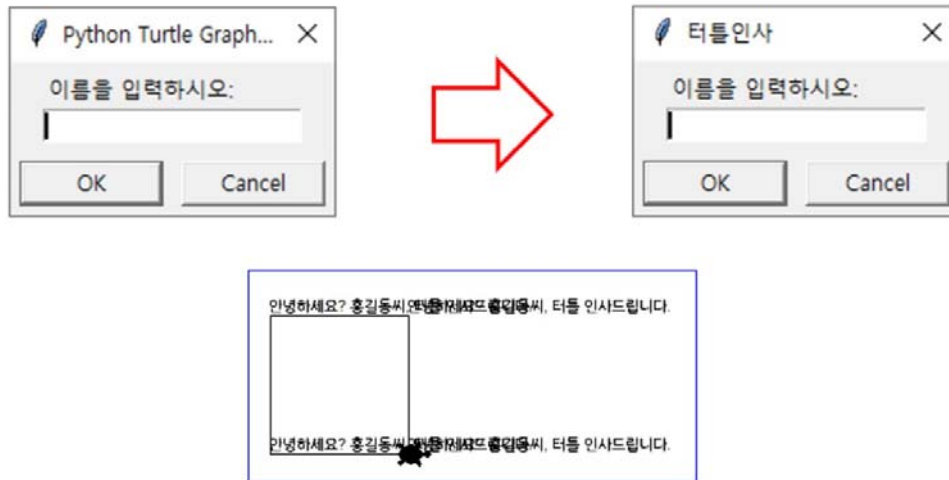




## 실습 - P0402 : 거북이와 인사해보자.

### ●도전 문제

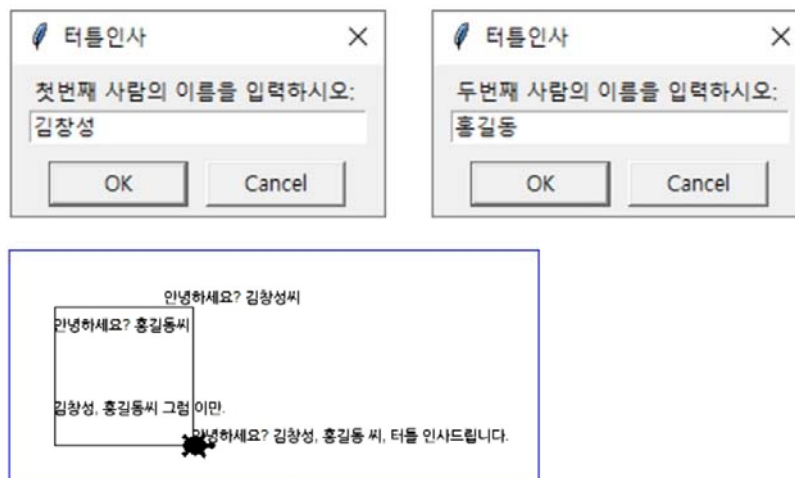
- 사각형의 각 꼭지점에 “안녕하세요? 홍길동씨, 터틀 인사드립니다.” 를 출력해보자
- 이름 입력 윈도우를 다음과 같이 변경하여 보자



## 실습 - P0403 : 거북이와 인사해보자.

### ●두사람이 거북이와 인사하려고 합니다.

- 실습 - P0401 를 수정하여 2명의 이름을 입력하고 다음과 같이 출력하세요.





## 개별 문자 추출

- 문자열에서 개별 문자들을 추출하려면 → 인덱스라는 번호를 사용한다.

|          |     |     |    |    |    |        |    |    |    |    |    |
|----------|-----|-----|----|----|----|--------|----|----|----|----|----|
|          |     |     |    |    |    | [6:10] |    |    |    |    |    |
| 0        | 1   | 2   | 3  | 4  | 5  | 6      | 7  | 8  | 9  | 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"  
print(s[6:10])
```

Pyth

## 실습 - P0404

- 변수 str1 에 'Hello Python' 이 저장되어 있다.
  - 양수 인덱스를 사용하여 'lo Py' 를 추출하여 출력하시오.
  - 음수 인덱스를 사용하여 'lo Py' 를 추출하여 출력하시오.
  - 양수 인덱스를 사용하여 'Python'을 추출하여 출력하시오.
- 변수 str2에 "안녕하세요. 파이썬입니다." 가 저장되어 있다.
  - '파이썬'을 추출하여 출력하시오.

## 특수 문자열

| 특수 문자열 | 의미       |
|--------|----------|
| \n     | 줄 바꿈 문자  |
| \t     | 탭 문자     |
| \\     | 역슬래시 자체  |
| \"     | 큰따옴표 자체  |
| \'     | 작은따옴표 자체 |

```
>>> print("말 한마디로\n천 냥 빚을 갚는다")
말 한마디로
천 냥 빚을 갚는다
```

## 특수 문자열

```
>>> message= 'doesn\'t'
>>> print(message)
doesn't
```

```
>>> message= 'doesn't'
SyntaxError: invalid syntax
```

## 실습 - P0405 : 친근하게 대화하는 프로그램

- 변수를 사용하여 사용자의 이름과 나이를 문자열 형태로 기억했다가 출력할 때 사용하는 프로그램을 작성해 보자.



```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
안녕하세요?
이름이 어떻게 되시나요? 홍길동
만나서 반갑습니다.홍길동씨
이름의 길이 : 3
나이가 어떻게 되나요? 21
내년이면 22 이 되시는군요.
>>>
```

- 문자열의 길이를 계산할 때는 `len(s)`를 사용한다.

## 실습 - P0405 : 친근하게 대화하는 프로그램

```
print('안녕하세요?')
name = input('이름이 어떻게 되시나요? ')
print('만나서 반갑습니다.' + name + "씨")

print('이름의 길이 :', end=' ')
print(len(name))

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

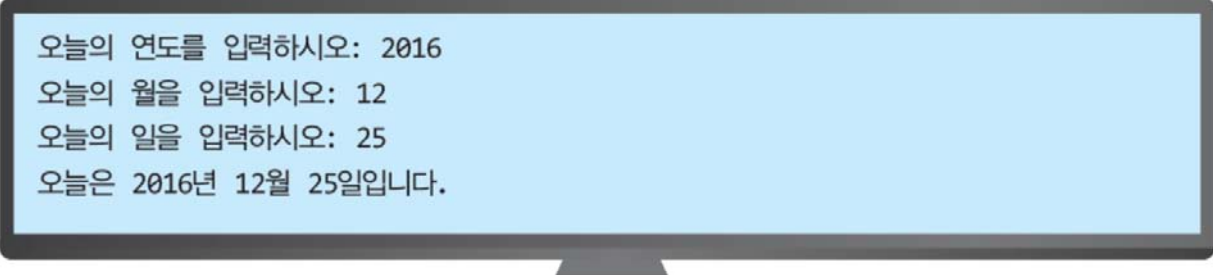
← % 를 사용하여 출력

← % 를 사용하여 출력

- % 를 사용하여 출력하는 프로그램을 작성하세요.

## 실습 - P0406 : 연, 월, 일을 합하여 출력하기

- 문자열을 저장하는 변수를 사용하여 사용자가 입력하는 오늘의 연도, 월, 일을 모두 합하여 화면에 출력하는 프로그램을 작성해 보자.

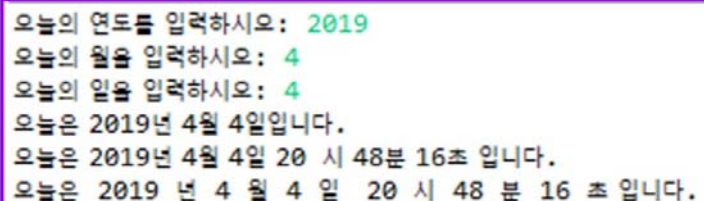


```
오늘의 연도를 입력하시오: 2016
오늘의 월을 입력하시오: 12
오늘의 일을 입력하시오: 25
오늘은 2016년 12월 25일입니다.
```

## 실습 - P0406 : 연, 월, 일을 합하여 출력하기

```
year = input("오늘의 연도를 입력하시오: ")
month = input("오늘의 월을 입력하시오: ")
date = input("오늘의 일을 입력하시오: ")
print("오늘은", year+"년", month+"월", date+"일입니다.")

from datetime import datetime
d = datetime.now()
print("오늘은 " + str(d.year) + "년 " + str(d.month) + "월 " + str(d.day) + "일 "
      + str(d.hour) + "시 " + str(d.minute) + "분 " + str(d.second) + "초 입니다.")
print("오늘은 ", d.year, "년 ", d.month, "월 ", d.day, "일 ", d.hour,
      "시 ", d.minute, "분 ", d.second, "초 입니다.")
```



```
오늘의 연도를 입력하시오: 2019
오늘의 월을 입력하시오: 4
오늘의 일을 입력하시오: 4
오늘은 2019년 4월 4일입니다.
오늘은 2019년 4월 4일 20 시 48분 16초 입니다.
오늘은 2019 년 4 월 4 일 20 시 48 분 16 초 입니다.
```

## 실습 - P0407 : 연, 월, 일을 합하여 출력하기

- 실습 - 05를 %s 를 사용하여 출력하도록 수정하시오

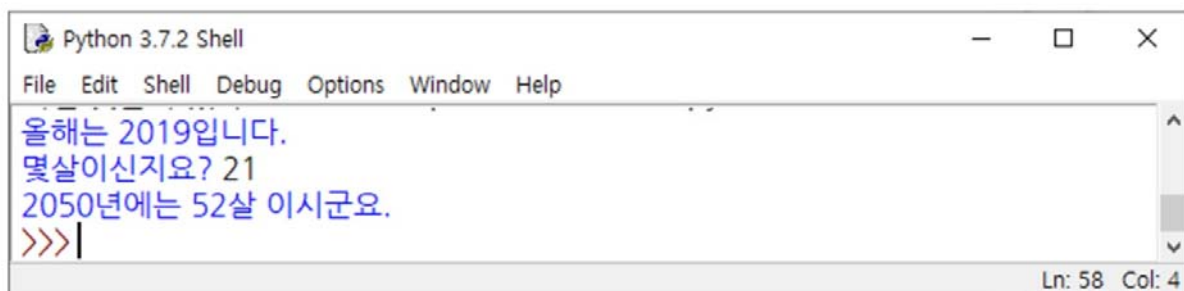
```
year = input("오늘의 연도를 입력하시오: ")
month = input("오늘의 월을 입력하시오: ")
date = input("오늘의 일을 입력하시오: ")
print("_____ " % (_____))
```

```
오늘의 연도를 입력하시오: 2019
오늘의 월을 입력하시오: 4
오늘의 일을 입력하시오: 2
오늘은 2019년 4월 2일입니다.
```

- % 기호를 사용할 때 여러개의 변수 값을 출력하기 위해서는 % ( A, B, C, ... ) 와 같이 변수를 지정해 주어야 한다.

## 실습 - P0408 : 2050년에는 몇 살이 될까?

- 자신이 2050년에 몇 살이 될 것인지를 계산하는 프로그램을 작성해 보자.



```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
올해는 2019입니다.
몇살이신지요? 21
2050년에는 52살 이시군요.
>>>|
```

Ln: 58 Col: 4

```
import time
now = time.time()
thisYear = int(1970 + now//(365*24*3600))
print("올해는 " + str(thisYear)+"입니다.")
```

- time() 함수 : 1970년 1월 1일부터 지금까지 경과된 초가 반환된다.



## 실습 - P0408 : 2050년에는 몇 살이 될까?

```
import time

now = time.time()
thisYear = int(1970 + now//(365*24*3600))
print("올해는 " + str(thisYear)+"입니다.")

age = int(input("몇 살이신지요? "))
print("2050년에는 "+str(age + 2050-thisYear)+"살 이시군요.")
```



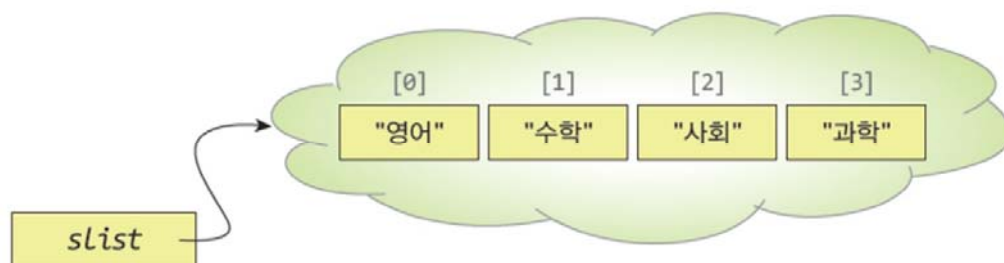
### 도전문제

str()을 사용하지 않고 print("올해는 ", thisYear, "입니다.")와 같이 쉼표를 사용하여 변수와 문자열을  
동시 출력할 수 있는가? 위의 프로그램을 이런 식으로 변경해보자. 어떤 방법이 편리한가?

## 리스트

- 리스트(list): 여러 개의 자료들을 모아서 하나의 묶음으로 저장하는 것
- [ ] 기호를 사용한다.

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





## 리스트에 항목을 동적으로 추가하기

- 공백 리스트를 생성한 후에 코드로 리스트에 값을 추가하는 것

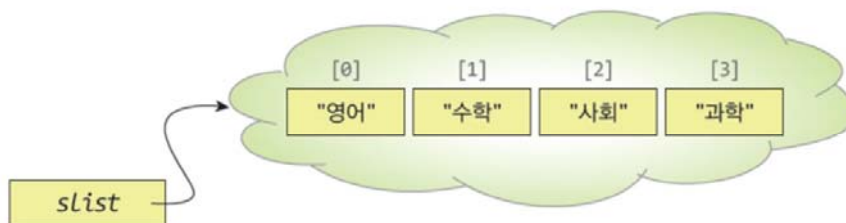
```
list = []  
list.append(1)  
list.append(2)  
list.append(6)  
list.append(3)  
  
print(list)
```

[1, 2, 6, 3]

## 리스트 요소 접근하기

```
slist = ['영어', '수학', '사회', '과학']  
print(slist[0])
```

영어



## 실습 - P0409 : 친구들의 리스트 생성하기

- 제일 친한 친구 5명의 이름을 리스트에 저장했다가 출력하는 프로그램을 작성하자.



```
Python 3.6.5 Shell
File Edit Shell Debug Options Window Help
친구의 이름을 입력하시오: 홍길동
친구의 이름을 입력하시오: 강감찬
친구의 이름을 입력하시오: 이순신
친구의 이름을 입력하시오: 권율
친구의 이름을 입력하시오: 정약용
['홍길동', '강감찬', '이순신', '권율', '정약용']
>>>
```

## 실습 - P0409 : 친구들의 리스트 생성하기

```
friend_list = []

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

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

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

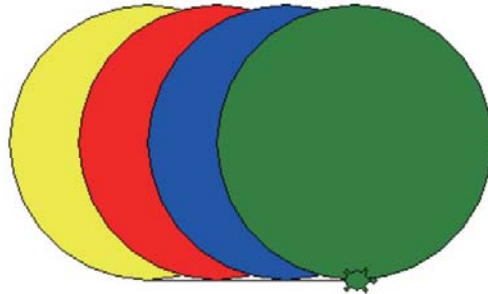
friend = input("친구의 이름을 입력하시오: ")
friend_list.append(friend)

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

print(friend_list)
```

## 실습 - P0410 : 리스트에 저장된 색상으로 원그리기

- 리스트에 색상을 문자열로 저장하였다가 하나씩 꺼내서 거북이의 채우기 색상으로 설정하고 원을 그려 보자.



## 실습 - P0410 : 리스트에 저장된 색상으로 원그리기

```
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.
```

## 실습 - P0410 : 리스트에 저장된 색상으로 원그리기

```
t.forward(50)
t.fillcolor(color_list[2])      # 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[3])      # 채우기 색상을 설정한다.
t.begin_fill()                 # 채우기를 시작한다.
t.circle(100)                  # 속이 채워진 원이 그려진다.
t.end_fill()                   # 채우기를 종료한다.
```

## 이번 장에서 배운 것

- 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 우리에게 반환한다.
- \n은 줄 바꿈을 나타내는 특수 문자열이다.
- 리스트는 자료들을 모아서 저장할 수 있다.

## 보충 자료

### 추가 자료형

- 튜플
- 딕셔너리
- 집합
- 불(bool)

## bool 형

### ●참(True)과 거짓(False)을 나타내는 자료형

- 2가지 값 이외의 값은 가지지 않는다.

```
>>> a = True
>>> b = False
>>> 1 == 1
True
>>> 2 > 1
True
>>> 2 < 1
False
>>> bool( [ ] )
False
>>> bool( [1,2,3] )
True
>>> bool("")
False
```

## bool 형

### ●자료형의 참과 거짓 구분

| 값        | 참 or 거짓 |
|----------|---------|
| "python" | 참       |
| ""       | 거짓      |
| [1,2,3]  | 참       |
| []       | 거짓      |
| ()       | 거짓      |
| {}       | 거짓      |
| 1        | 참       |
| 0        | 거짓      |
| None     | 거짓      |