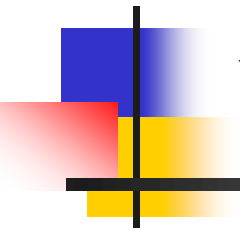
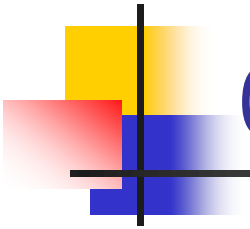


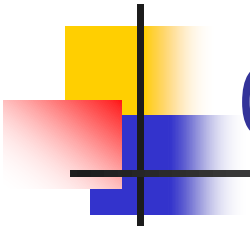
02장 파이썬 프로그래밍의 기초, 자료형

자료형을 알고 있다면 그 언어의 절반을 터득한 것



02-1 숫자형

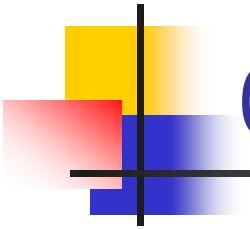
- 정수형 (1, 2, -2)
- 실수 (1.24, -34.56)
- 컴퓨터식 지수 표현 방식 (4.24e10, 4.24e-10)
- 복소수 (1+2j)
- 8진수 (0o37)
- 16진수 (0x7A)



02-1 숫자형

사칙연산

```
>>> a = 3
>>> b = 4
>>> a + b
7
>>> a * b
12
>>> a / b
0.75
```



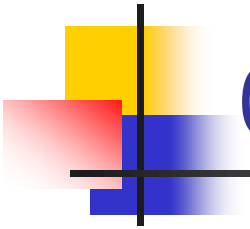
02-1 숫자형

제곱

```
>>> a = 3
>>> b = 4
>>> a ** b
81
```

% 연산자

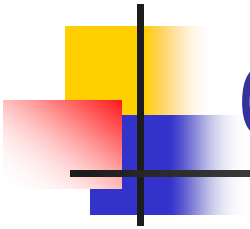
```
>>> 7 % 3
1
>>> 3 % 7
3
```



02-1 숫자형

// 연산자

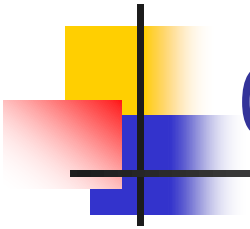
```
>>> 7 / 4
1.75
>>> 7 // 4
1
```



02-2 문자열 자료형

문자열 자료형 만드는 4가지 방법

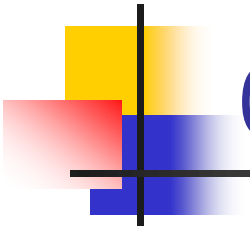
```
"Hello World"

'Python is fun'

"""Life is too short, You need python"""

'''Life is too short, You need python'''
```



02-2 문자열 자료형

문자열에 따옴표 포함시키기

```
>>> food = "Python's favorite food is perl"
>>> say = '"Python is very easy." he says.'
>>> food = 'Python\'s favorite food is perl'
>>> say = "\"Python is very easy.\" he says."
```

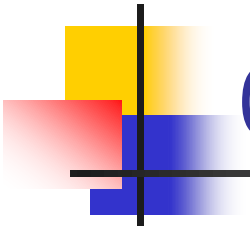


02-2 문자열 자료형

여러 줄로 이루어진 문자열

```
>>> multiline = "Life is too short\nYou need python"
```

```
>>> multiline='''
... Life is too short
... You need python
... '''
```



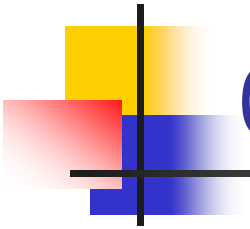
02-2 문자열 자료형

문자열 더해서 연결하기 (Concatenation)

```
>>> head = "Python"
>>> tail = " is fun!"
>>> head + tail
'Python is fun!'
```

문자열 곱하기

```
>>> a = "python"
>>> a * 2
'pythonpython'
```



02-2 문자열 자료형

인덱싱(Indexing)

```
>>> a = "Life is too short, You need Python"
>>> a[0]
'L'
>>> a[12]
's'
>>> a[-1]
'n'
```

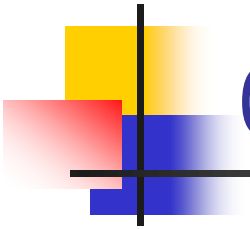
파이썬은 0부터 숫자를 센다

02-2 문자열 자료형

슬라이싱(Slicing)

```
>>> a = "Life is too short, You need Python"
>>> a[0:4]
'Life'
```

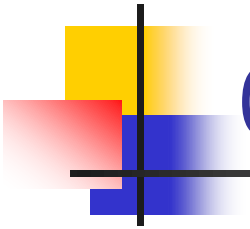
```
>>> a = "20010331Rainy"
>>> date = a[:8]
>>> weather = a[8:]
>>> date
'20010331'
>>> weather
'Rainy'
```



02-2 문자열 자료형

문자열 포매팅

```
>>> "I eat %d apples." % 3
'I eat 3 apples.'
>>> number = 10
>>> day = "three"
>>> "I ate %d apples. so I was sick for %s days." % (number,
day)
'I ate 10 apples. so I was sick for three days.'
```



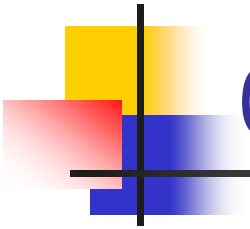
02-2 문자열 자료형

정렬과 공백

```
>>> "%10s" % "hi"
'        hi'
>>> "%-10sjane." % 'hi'
'hi        jane.'
```

소수점 표현

```
>>> "%0.4f" % 3.42134234
'3.4213'
>>> "%10.4f" % 3.42134234
'    3.4213'
```



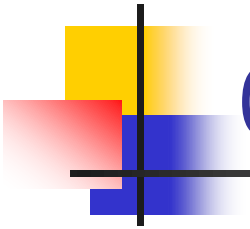
02-2 문자열 자료형

문자열 개수 세기(count)

```
>>> a = "hobby"
>>> a.count('b')
2
```

위치 알려주기1(find)

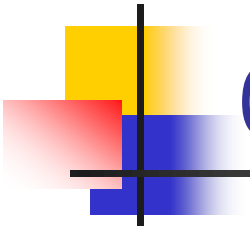
```
>>> a = "Python is best choice"
>>> a.find('b')
10
>>> a.find('k')
-1
```



02-2 문자열 자료형

위치 알려주기2(index)

```
>>> a = "Life is too short"
>>> a.index('t')
8
>>> a.index('k')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: substring not found
```



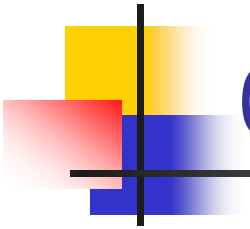
02-2 문자열 자료형

문자열 삽입(join)

```
>>> a= ","
>>> a.join('abcd')
'a,b,c,d'
```

소문자를 대문자로 바꾸기(upper)

```
>>> a = "hi"
>>> a.upper()
'HI'
```



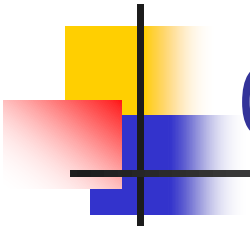
02-2 문자열 자료형

대문자를 소문자로 바꾸기(lower)

```
>>> a = "HI"
>>> a.lower()
'hi'
```

양쪽 공백 지우기(strip)

```
>>> a = " hi "
>>> a.strip()
'hi'
```



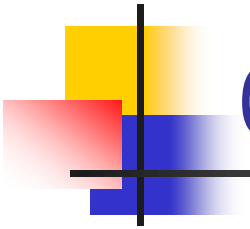
02-2 문자열 자료형

문자열 바꾸기(replace)

```
>>> a = "Life is too short"
>>> a.replace("Life", "Your leg")
'Your leg is too short'
```

문자열 나누기(split)

```
>>> a = "Life is too short"
>>> a.split()
['Life', 'is', 'too', 'short']
>>> a = "a:b:c:d"
>>> a.split(':')
['a', 'b', 'c', 'd']
```



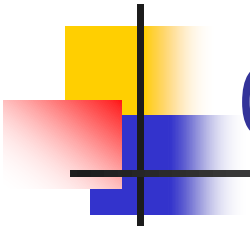
02-3 리스트 자료형

1, 3, 5, 7, 9라는 숫자 모음

```
>>> odd = [1, 3, 5, 7, 9]
```

리스트명 = [요소1, 요소2, 요소3, ...]

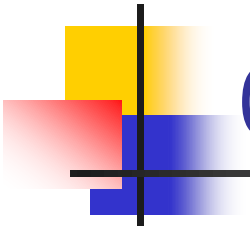
```
>>> a = [ ]
>>> b = [1, 2, 3]
>>> c = ['Life', 'is', 'too', 'short']
>>> d = [1, 2, 'Life', 'is']
>>> e = [1, 2, ['Life', 'is']]
```



02-3 리스트 자료형

리스트의 인덱싱

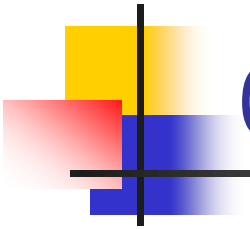
```
>>> a = [1, 2, 3]
>>> a[0]
1
>>> a[0] + a[2]
4
>>> a[-1]
3
```



02-3 리스트 자료형

리스트의 슬라이싱

```
>>> a = [1, 2, 3, 4, 5]
>>> a[0:2]
[1, 2]
>>> b = a[:2]
>>> c = a[2:]
>>> b
[1, 2]
>>> c
[3, 4, 5]
```



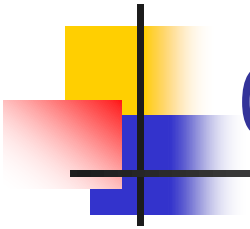
02-3 리스트 자료형

리스트 더하기

```
>>> a = [1, 2, 3]
>>> b = [4, 5, 6]
>>> a + b
[1, 2, 3, 4, 5, 6]
```

리스트 반복하기

```
>>> a = [1, 2, 3]
>>> a * 3
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```



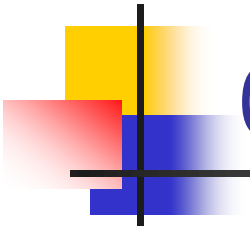
02-3 리스트 자료형

리스트에서 하나의 값 수정하기

```
>>> a = [1, 2, 3]
>>> a[2] = 4
>>> a
[1, 2, 4]
```

리스트에서 연속된 범위의 값 수정하기

```
>>> a[1:2]
[2]
>>> a[1:2] = ['a', 'b', 'c']
>>> a
[1, 'a', 'b', 'c', 4]
```



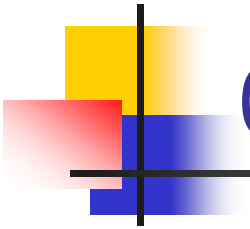
02-3 리스트 자료형

[ ] 사용해 리스트 요소 삭제하기

```
>>> a = [1, 'a', 'b', 'c', 4]
>>> a[1:3] = []
>>> a
[1, 'c', 4]
```

del 함수 사용해 리스트 요소 삭제하기

```
>>> a
[1, 'c', 4]
>>> del a[1]
>>> a
[1, 4]
```



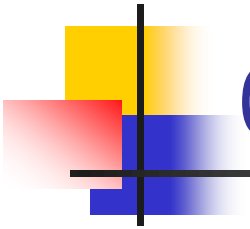
02-3 리스트 자료형

리스트에 요소 추가(append)

```
>>> a = [1, 2, 3]
>>> a.append(4)
>>> a
[1, 2, 3, 4]
```

리스트 정렬(sort)

```
>>> a = [1, 4, 3, 2]
>>> a.sort()
>>> a
[1, 2, 3, 4]
```



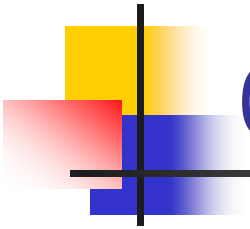
02-3 리스트 자료형

리스트 뒤집기(reverse)

```
>>> a = ['a', 'c', 'b']
>>> a.reverse()
>>> a
['b', 'c', 'a']
```

위치 반환(index)

```
>>> a = [1,2,3]
>>> a.index(3)
2
>>> a.index(1)
0
```



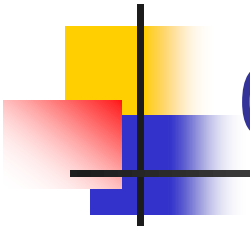
02-3 리스트 자료형

리스트에 요소 삽입(insert)

```
>>> a = [1, 2, 3]
>>> a.insert(0, 4)
[4, 1, 2, 3]
```

리스트 요소 제거(remove)

```
>>> a = [1, 2, 3, 1, 2, 3]
>>> a.remove(3)
[1, 2, 1, 2, 3]
```



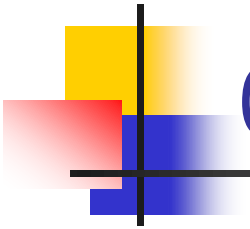
02-3 리스트 자료형

리스트 요소 끄집어내기(pop)

```
>>> a = [1,2,3]
>>> a.pop()
3
>>> a
[1, 2]
```

리스트에 포함된 요소 x의 개수 세기(count)

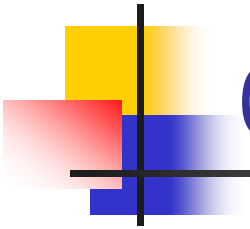
```
>>> a = [1,2,3,1]
>>> a.count(1)
2
```



02-3 리스트 자료형

리스트 확장(extend)

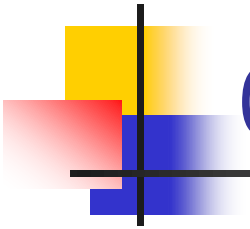
```
>>> a = [1,2,3]
>>> a.extend([4,5])
>>> a
[1, 2, 3, 4, 5]
>>> b = [6, 7]
>>> a.extend(b)
>>> a
[1, 2, 3, 4, 5, 6, 7]
```



02-4 튜플 자료형

튜플 요소값 삭제 시 오류

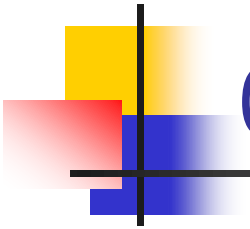
```
>>> t1 = (1, 2, 'a', 'b')
>>> del t1[0]
Traceback (innermost last):
File "", line 1, in ?del t1[0]
TypeError: object doesn't support item deletion
```



02-4 튜플 자료형

튜플 요소값 변경 시 오류

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[0] = 'c'
Traceback (innermost last):
File "", line 1, in ?t1[0] = 'c'
TypeError: object doesn't support item assignment
```



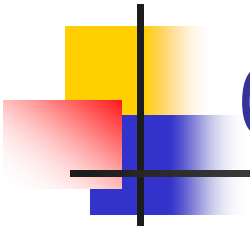
02-4 튜플 자료형

인덱싱

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[0]
1
>>> t1[3]
'b'
```

슬라이싱

```
>>> t1 = (1, 2, 'a', 'b')
>>> t1[1:]
(2, 'a', 'b')
```



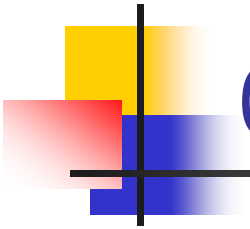
02-4 튜플 자료형

더하기

```
>>> t2 = (3, 4)
>>> t1 + t2
(1, 2, 'a', 'b', 3, 4)
```

곱하기

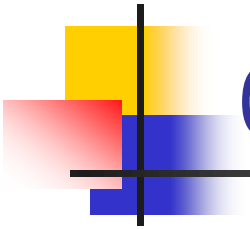
```
>>> t2 * 3
(3, 4, 3, 4, 3, 4)
```



02-5 딕셔너리 자료형

- 연관 배열(Associative array) 또는 해시(Hash)
- 단어 그대로 해석하면 사전이라는 뜻
- Key를 통해 Value를 얻는다

```
>>> dic = {'name':'pey', 'phone':'0119993323', 'birth': '1118'}
```



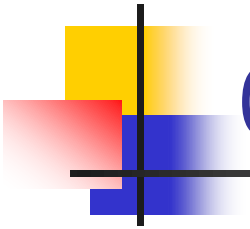
02-5 딕셔너리 자료형

딕셔너리 쌍 추가하기

```
>>> a = {1: 'a'}
>>> a[2] = 'b'
>>> a
{2: 'b', 1: 'a'}
```

딕셔너리 요소 삭제하기

```
>>> del a[1]
>>> a
{'name': 'pey', 3: [1, 2, 3], 2: 'b'}
```



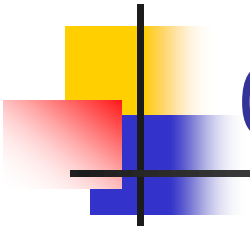
02-5 딕셔너리 자료형

딕셔너리에서 Key 사용해 Value 얻기

```
>>> grade = {'pey': 10, 'julliet': 99}
>>> grade['pey']
10
>>> grade['julliet']
99
```

딕셔너리 만들 때 주의할 사항

```
>>> a = {1:'a', 1:'b'}
>>> a
{1: 'b'}
```



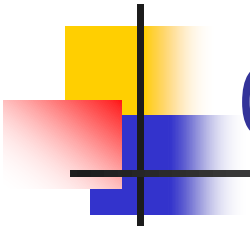
02-5 딕셔너리 자료형

Key 리스트 만들기(keys)

```
>>> a = {'name': 'pey', 'phone': '0119993323', 'birth':
'1118'}
>>> a.keys()
dict_keys(['name', 'phone', 'birth'])
```

Value 리스트 만들기(values)

```
>>> a.values()
dict_values(['pey', '0119993323', '1118'])
```



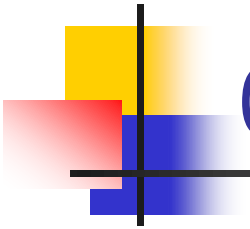
02-5 딕셔너리 자료형

Key, Value 쌍 얻기(items)

```
>>> a = {'name': 'pey', 'phone': '0119993323', 'birth':
'1118'}
>>> a.items()
dict_items([('name', 'pey'), ('phone', '0119993323'),
('birth', '1118')])
```

Key: Value 쌍 모두 지우기(clear)

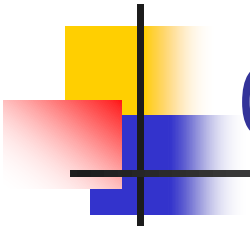
```
>>> a.clear()
>>> a
{}
```



02-5 딕셔너리 자료형

Key로 Value얻기(get)

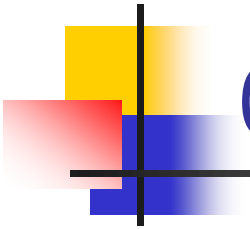
```
>>> a = {'name':'pey', 'phone':'0119993323', 'birth': '1118'}
>>> a.get('name')
'pey'
>>> a.get('phone')
'0119993323'
```



02-5 딕셔너리 자료형

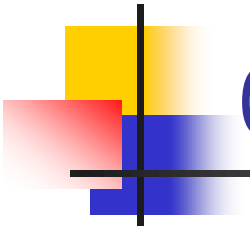
해당 Key가 딕셔너리 안에 있는지 조사하기(in)

```
>>> a = {'name':'pey', 'phone':'0119993323', 'birth': '1118'}
>>> 'name' in a
True
>>> 'email' in a
False
```



02-6 집합 자료형

- 집합에 관련된 것들을 쉽게 처리하기 위해 만들어진 자료형
- 중복을 허용하지 않는다.
- 순서가 없다(Unordered).



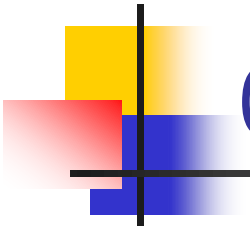
02-6 집합 자료형

집합 자료형

```
>>> s1 = set([1,2,3])
>>> s1
{1, 2, 3}
```

순서가 없고 중복이 허용되지 않는다

```
>>> s2 = set("Hello")
>>> s2
{'e', 'l', 'o', 'H'}
```



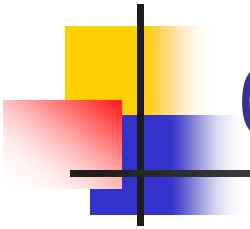
02-6 집합 자료형

교집합 1

```
>>> s1 = set([1, 2, 3, 4, 5, 6])
>>> s2 = set([4, 5, 6, 7, 8, 9])
>>> s1 & s2
{4, 5, 6}
```

교집합 2

```
>>> s1.intersection(s2)
{4, 5, 6}
```



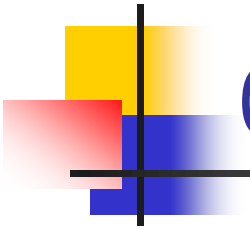
02-6 집합 자료형

합집합 1

```
>>> s1 = set([1, 2, 3, 4, 5, 6])
>>> s2 = set([4, 5, 6, 7, 8, 9])
>>> s1 | s2
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```

합집합 2

```
>>> s1.union(s2)
{1, 2, 3, 4, 5, 6, 7, 8, 9}
```



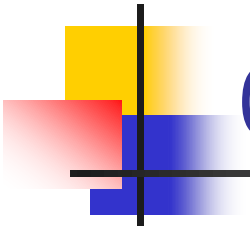
02-6 집합 자료형

차집합 1

```
>>> s1 = set([1, 2, 3, 4, 5, 6])
>>> s2 = set([4, 5, 6, 7, 8, 9])
>>> s1 - s2
{1, 2, 3}
>>> s2 - s1
{8, 9, 7}
```

차집합 2

```
>>> s1.difference(s2)
{1, 2, 3}
>>> s2.difference(s1)
{8, 9, 7}
```



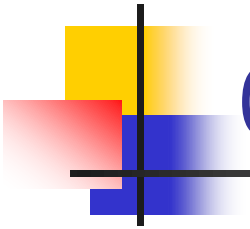
02-6 집합 자료형

값 1개 추가하기(add)

```
>>> s1 = set([1, 2, 3])
>>> s1.add(4)
>>> s1
{1, 2, 3, 4}
```

값 여러 개 추가하기(update)

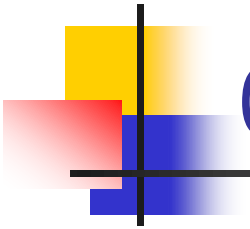
```
>>> s1 = set([1, 2, 3])
>>> s1.update([4, 5, 6])
>>> s1
{1, 2, 3, 4, 5, 6}
```



02-6 집합 자료형

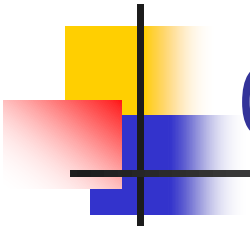
특정 값 제거하기(remove)

```
>>> s1 = set([1, 2, 3])
>>> s1.remove(2)
>>> s1
{1, 3}
```



02-7 자료형의 참과 거짓

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



02-7 자료형의 참과 거짓

자료형의 참과 거짓은 어떻게 사용되나?

```
>>> a = [1, 2, 3, 4]
>>> while a:
...     a.pop()
...
4
3
2
1
```

02-8 자료형의 값을 저장하는 공간, 변수

다음 예와 같은 a, b, c를 변수라고 한다.

```
>>> a = 1
>>> b = "python"
>>> c = [1,2,3]
```

변수를 만들 때는 =(assignment) 기호를 사용한다.

02-8 자료형의 값을 저장하는 공간, 변수

파이썬에서 사용하는 변수는 객체를 가리키는 것

```
>>> a = 3
```

- 3이라는 값을 가지는 정수 자료형(객체)이 자동으로 메모리에 생성
- 변수 a는 객체가 저장된 메모리의 위치를 가리키는 레퍼런스(Reference)
- a라는 변수는 3이라는 정수형 객체를 가리키고 있다

02-8 자료형의 값을 저장하는 공간, 변수

리스트 변수 주의사항

```
>>> a = [1,2,3]
>>> b = a
>>> a[1] = 4
>>> a
[1, 4, 3]
>>> b
[1, 4, 3]
```