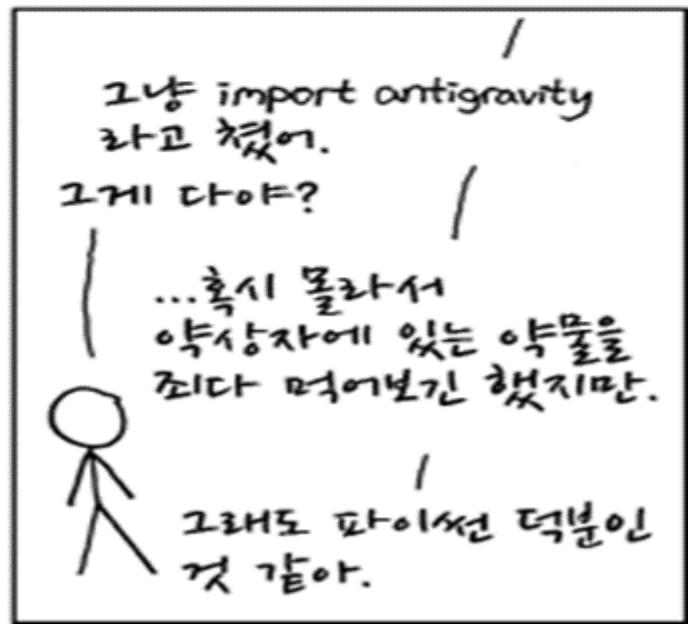
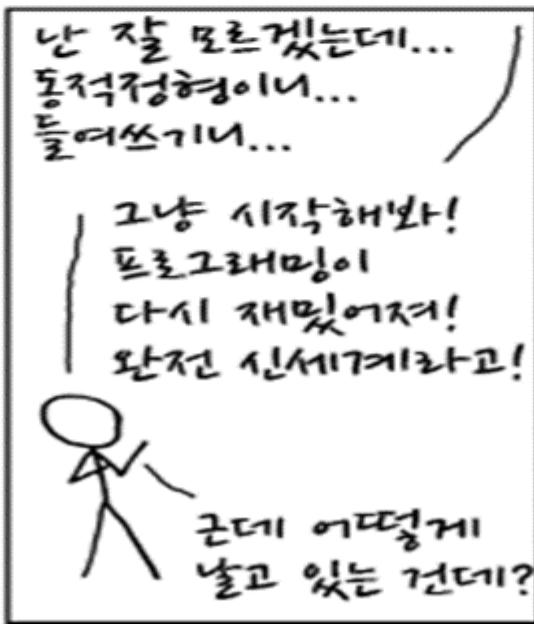


정보과학

I. 프로그래밍

1. 프로그래밍의 기초
2. 선택 실행 구조
3. 반복실행구조
4. 데이터와 작업의 모듈화



출력 함수 print()

값을 모니터에 출력하는 함수

print() 함수의 사용 예시

```
print("Hello World")
```

```
print(30)
```

```
print(3.14)
```

```
a = "apple"
```

```
b = "banana"
```

```
c = "kiwi"
```

```
print(a, b, c)
```

```
print("a", "b", "c")
```

```
print(a, b, c, sep='')
```

```
print(a, b, c, sep='하고 ')
```

```
print("%.2f"%3.141592)
```



변수와 (리터럴)상수

변수 : 기억장소(실행 중 값을 변경 가능)
(리터럴)상수 : 고유한 값(실행 중 변경 불가)

```
name = "홍길동"  
age = 17  
kor = 80  
eng = 100  
avg = (kor + eng) / 2  
print(name, "평균은", avg, "입니다.")
```



변수(기억장소)

정수(리터럴)상수 : 17, 80, 100, 2

문자(리터럴)상수 : "홍길동", "평균은", "입니다."

(리터럴)상수 – 고유한 값으로 실행 중 변하지 않음

▶ 숫자형 (리터럴)상수 – 수학적 연산이 가능

▶ 정수형 상수(<int>) – 숫자로 구성되고, 소수점이 없음

▶ 0, 1, -1, 23472, -402083464, ...

▶ 실수형 (리터럴)상수(<float>) – 숫자로 구성되고, 소수점이 있음

▶ 0.0, 1.0, -1.5, 3.14, -5.2, 542342.0, $1e+3 (=1000.0)$...

▶ 복소수형 (리터럴)상수(<complex>) – 복소수

▶ $3 + 4j$, $3 - 4j$, ...

▶ 문자형 (리터럴)상수(<str>) – 수학적 연산이 불가능

▶ 작은 or 큰 따옴표('??', "??")로 싸여진 상수(따옴표 안의 내용 무관)

▶ 'abc', "Hello", "안녕", "1", "\n", "3.14", "2022-03-08", ...

변수(variable) – 실행 중 값이 변경될 수 있음.

▶ 변수 : 값을 1개 기억하는 기억장소 혹은 기억장소 이름

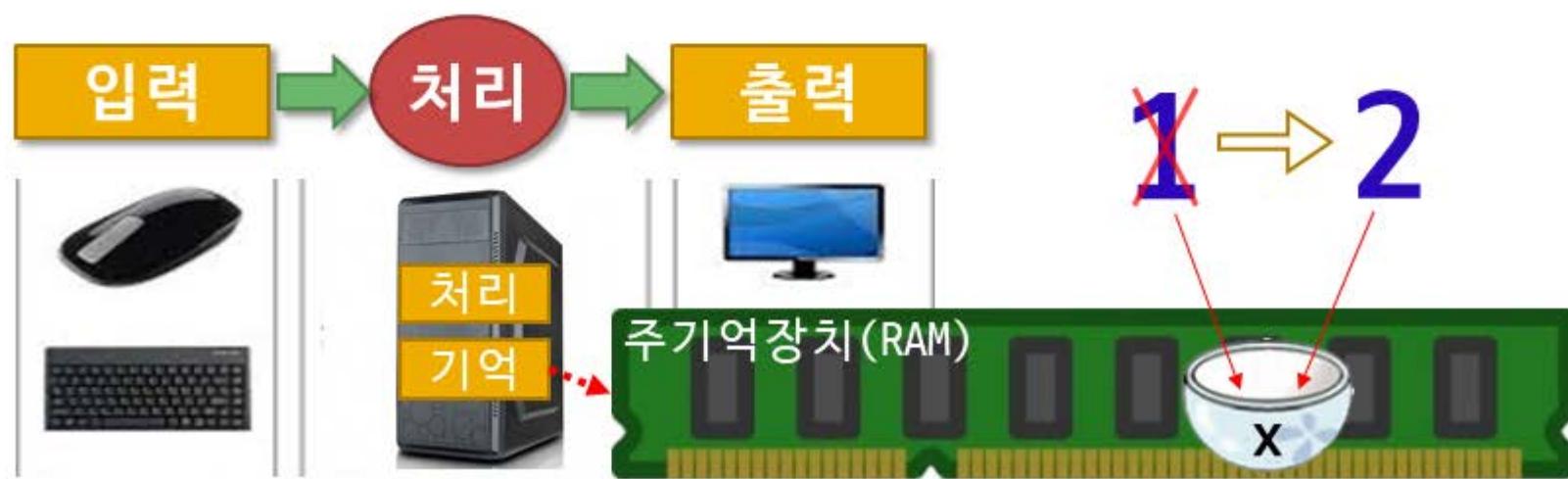
변수에 값을 저장 시킬 때는 =

형식) **변수 = 값**

`x = 1` # 변수 x에 1을 저장

변수 x에 저장된 값이
1에서 2로 바뀜.

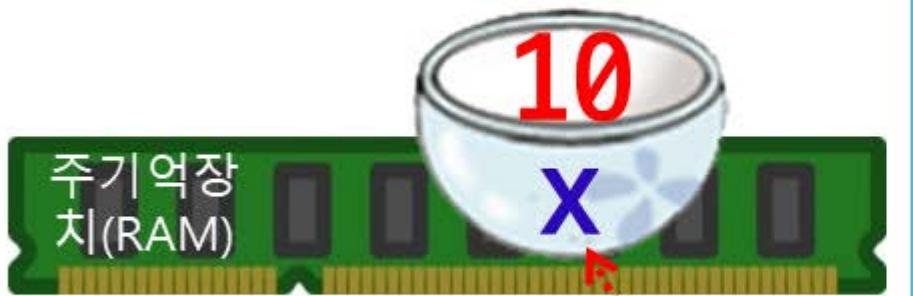
`x = 2` # 변수 x에 2을 저장



변수(variable)의 속성

형식) **변수 = 값**

예) **X = 10**



* 변수의 속성

가. 기억장소의 이름 : **X**
(변수명)

나. (기억장소에 저장된) 값 : **10**

다. 저장된 값의 자료형 :
(정수 or 실수 or 문자) **정수**

변수와 (리터럴)상수의 사용 예

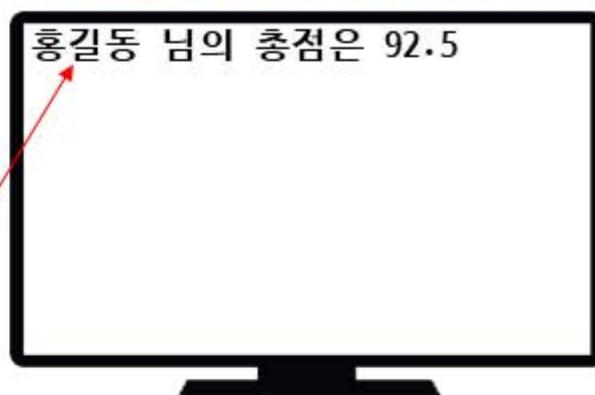
```
name = "홍길동"
```

```
kor = 90
```

```
eng = 95
```

```
avg = (kor + eng) / 2
```

```
print(name, "의 평균은", avg)
```



입력

처리

출력



변수의 이름의 명명규칙

변수의 명명규칙

1. 알파벳, 숫자, _ 의 조합으로만
2. 숫자로 시작하면 안됨
3. 예약어(**if, else, for, break** 등)를 변수로 사용하면 안됨.

변수명으로 사용 가능한 것?

- | | | | | | | | |
|--------------------|------------|------------|------------------|----------|----------|----------|-------------|
| 1
apink | 2
bts | 3
BTS | 4
B.T.S | 5
IU | 6
I_U | 7
I-U | 8
IZ*ONE |
| 9
<u>"cnsh"</u> | 10
cnsH | 11
10cm | 12
black pink | 13
if | | | |

다음 실수는 변수 or 상수(정수 or 실수 or 문자) 인가?

	(리터럴)상수			변수명	쓸모없음
	정수	실수	문자(열)		
0	0				
-3.141592		0			
"CNSH"				0	
"29th"				0	
29th					X
sh29				0	
name				0	
if					X(예약어)
"if"				0	
CNSH				0	

대입연산자 : = 변수에 값을 대입(할당)

L_value = R_value

기억장소

변수



값

상수, 변수, 연산식

```
name = "홍길동"
```

```
kor = 90
```

```
eng = 95
```

```
avg = (kor + eng) / 2
```

```
print(name, "의 평균은", avg, "입니다.")
```



홍길동 의 총점은 92.5 입니다.

변수의 값 할당 방식

`school = "CNSH"`



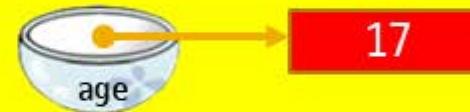
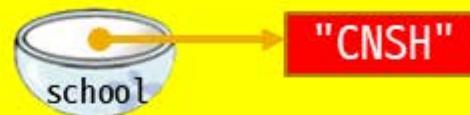
`age = 17`



`height = 160.2`

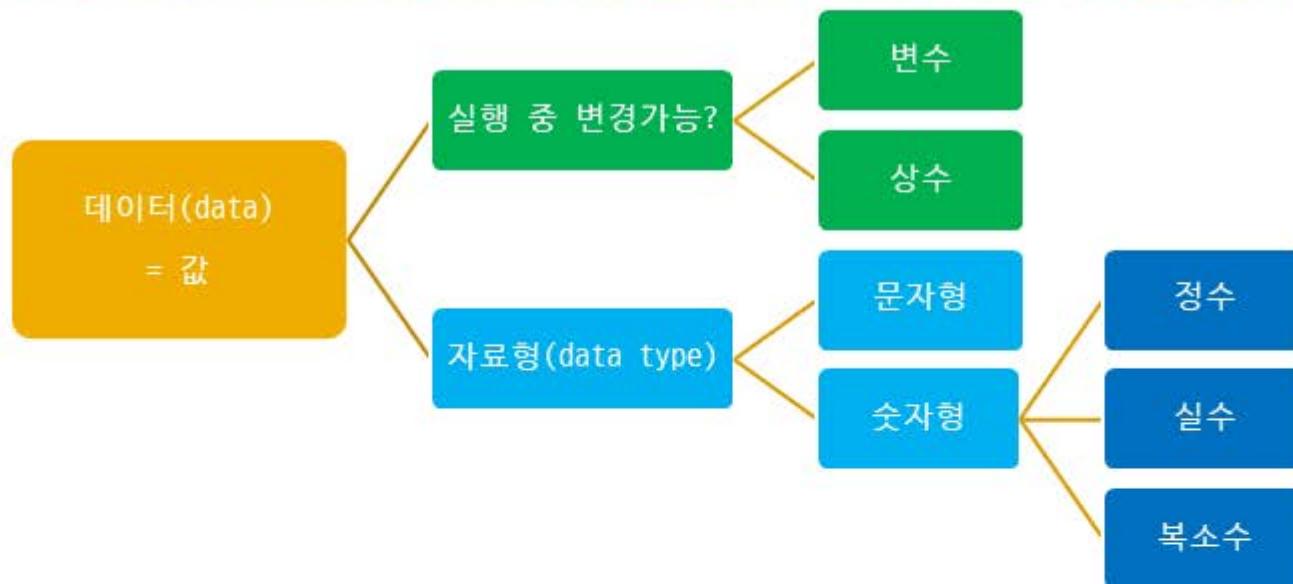


파이썬 언어에서
실제 변수와 데이터(값)의 저장 방식



파이썬에서 모든 데이터는 객체(Object)
실제 저장되는 방식
변수에는 데이터 객체의 주소(위치)를 기억

데이터(값)의 분류



상수를 분류해 보시오.

1. 문자형?
2. 정수형?
3. 실수형?

변수를 분류해 보시오.

1. 문자형 변수는?
2. 정수형 변수는?
3. 실수형 변수는?

```
name = "홍길동"  
age = 17  
height = 175.8  
print("나는", name, "이다.")
```



나는 홍길동 이다.

type() : 데이터(변수, 상수)의 자료형을 알고 싶다면

```
name = "홍길동"
```

```
age = 17
```

```
pi = 3.14
```

```
print("cnsh", type("cnsh"))
```

```
print(name, type(name))
```

연산자(operator)

수학시간에 사용하는 연산기호를 프로그래밍에서는 연산자라 함.

- 대입연산자 : =
- 산술연산자 : +, -, *, /, //, %, **
- 관계(비교)연산자 : >, <, >=, <=, ==, !=
- 논리연산자 : and, or, not

(실습자료) 숫자의 연산

```
a = 10
```

```
b = 2
```

```
print(a + b) # 12
```

```
print(a - b) # 8
```

```
print(a * b) # 20
```

```
print(a ** b) # 100
```

```
print(a / b) # 5.0
```

```
a = 13
```

```
b = 5
```

```
print(a / b) # 2.6
```

```
print(a // b) # 2
```

```
print(a % b) # 3
```

변수와 (리터럴)상수의 사용 예

```
name = "홍길동"
```

```
kor = 90
```

```
eng = 95
```

```
avg = (kor + eng) / 2
```

```
print(name, "의 평균은", avg)
```

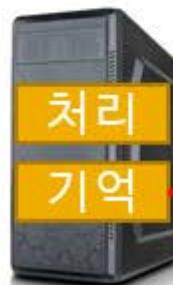
홍길동 님의 총점은 92.5

avg는 어떤 자료형(data type)?

입력

처리

출력



연산자 - 연산기호

▶ 산술연산자

▶ **+** : (더하기)

▶ 예시 : a = 10 + 5

10과 5를 더해서(결과 15) 변수 a에 저장(대입)

▶ **-** : (빼기)

▶ 예시 : b = 10 - 5

10에서 5를 빼서(결과 5) 변수 b에 저장(대입)

▶ ***** : (곱하기)

▶ 예시 : c = 10 * 5

10과 5를 곱해서(결과 50) 변수 c에 저장(대입)

▶ **/** : (나누기) ← 특이사항 : 결과는 실수가 됨.

▶ 예시 : d = 12 / 5

12를 5로 나눠서(결과 2.4) 변수 d에 저장(대입)

▶ **//** : (몫)

▶ 예시 : e = 13 // 5

13을 5로 나눈 몫(2)을 변수 e에 저장(대입)

▶ **%** : (나머지)

▶ 예시 : f = 13 % 5

13을 5로 나눈 나머지(3)을 변수 f에 저장(대입)

▶ 예시 : f = 10 % 5

13을 5로 나눈 나머지(0)을 변수 f에 저장(대입)

▶ ****** : (제곱)

▶ 예시 : g = 2 ** 3

2의 3제곱(2^3 즉, 8)을 변수 g에 저장(대입)

문자의 연산

- ▶ 문자는 연산할 수 없다.
- ▶ 단, 아래 2가지는 가능
 - ▶ 문자 + 문자 → 문자를 연결
 - ▶ $a = "충남" + "과학"$ # $a \leftarrow "충남과학"$
 - ▶ $b = a + "고등학교"$ # $b \leftarrow "충남과학고등학교"$
 - ▶ 문자 * 자연수 → 문자를 자연수번 만큼 연결
 - ▶ $s = "Hi" * 5$ # $c \leftarrow "HiHiHi"$

(실습자료) 문자의 연산

```
a = "충곽"
```

```
b = "30"
```

```
c = "2022"
```

```
print(a, b, c)
```

```
print(a + b)
```

```
print(b + c)
```

```
print(a * 3)
```

충곽 30 2022

충곽30

302023

충곽충곽충곽

문자에 대해 허용하는 연산

가. 문자 + 문자 : 두 문자를 연결

나. 문자 * 정수 : 문자를 정수번 연결

입력함수 input()과 형변환(casting)함수(int(), float(), str())

입력함수 input()

- 실행 중 키보드로 값을 입력 받을 수 있음
- 특징
 - 줄 단위로 입력
 - 무조건 문자로 입력

실습) 아래의 코드(들)을 비교하시오.

```
name = "홍길동"
```

홍길동 총점은 185

```
kor = 90
```

```
eng = 95
```

```
sum = kor + eng
```

```
print(name, "총점은", sum)
```

```
name = input()
```

```
kor = input()
```

```
eng = input()
```

```
sum = kor + eng
```

```
print(name, "총점은", sum)
```

홍길동 Enter

90 Enter

95 Enter

홍길동 총점은 9095

```
name = input()
```

홍길동 Enter

90 Enter

95 Enter

홍길동 총점은 185

```
kor = int(input())  
eng = int(input())
```

```
sum = kor + eng
```

```
print(name, "의 총점은", sum)
```

```
name = input("이름?")
```

```
kor = int(input("국어?"))  
eng = int(input("영어?"))
```

```
sum = kor + eng
```

```
print(name, "의 총점은", sum)
```

이름? 홍길동 Enter

국어? 90 Enter

영어? 95 Enter

홍길동 총점은 9095

```
name = input()
```

```
kor = input()
```

```
eng = input()
```

```
sum = kor + eng
```

```
print(name, "총점은", sum)
```



input() : 실행 중 값을 입력

특징 : 1. **줄 단위**로 입력됨.

2. 입력값에 관계없이 **문자형**으로 입력됨.

```
name = input()
```

```
kor = float(input())
```

```
eng = float(input())
```

```
sum = kor + eng
```

```
print(name, "총점은", sum)
```



float(값) : 괄호 안에 값을 **실수형**으로 변환

예) int("3.14")→3.14, int(2)→2.0

```
name = input()
```

```
kor = int(input())
```

```
eng = int(input())
```

```
sum = kor + eng
```

```
print(name, "총점은", sum)
```



int(값) : 괄호 안에 값을 **정수형**으로 변환

예) int("15")→15, int(12.5)→12

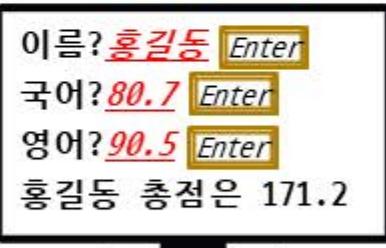
```
name = input("이름?")
```

```
kor = float(input("국어?"))
```

```
eng = float(input("영어?"))
```

```
sum = kor + eng
```

```
print(name, "총점은", sum)
```



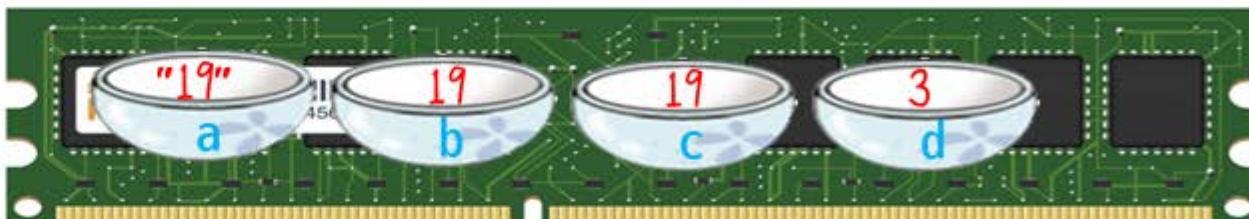
input("이름을 입력하시오")

→ 무엇을 입력해야 하는지를 안내 할 수 있음.

형(type) 변환 함수 – casting

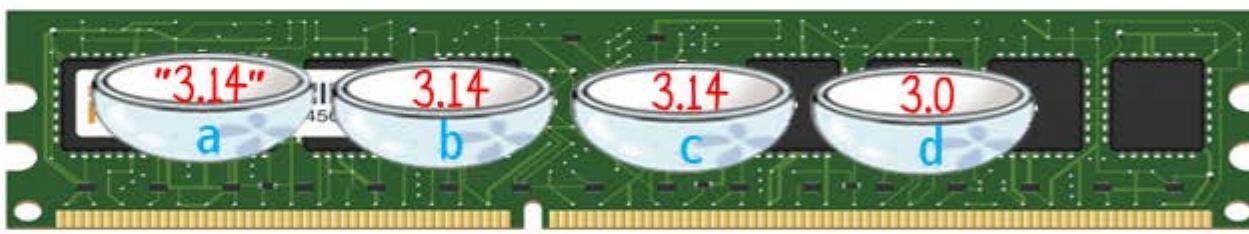
- ▶ `int()` : 정수형(integer)으로 형변환(<int>)하는 함수

```
a = "19"  
b = int("19")  
c = int(a)  
d = int(3.14)
```



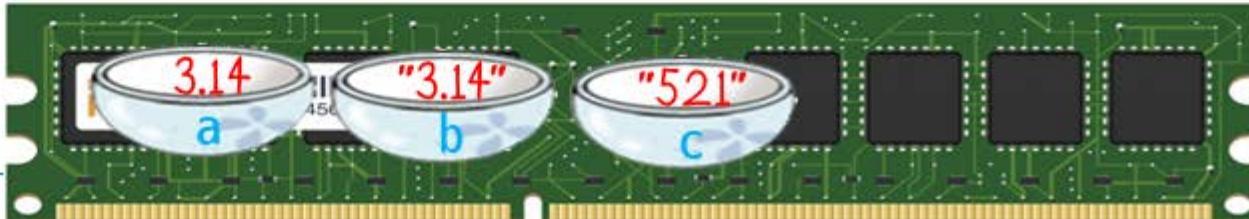
- ▶ `float()` : 실수형(float)으로 형변환(<float>)하는 함수

```
a = "3.14"  
b = float("3.14")  
c = float(a)  
d = float(3)
```

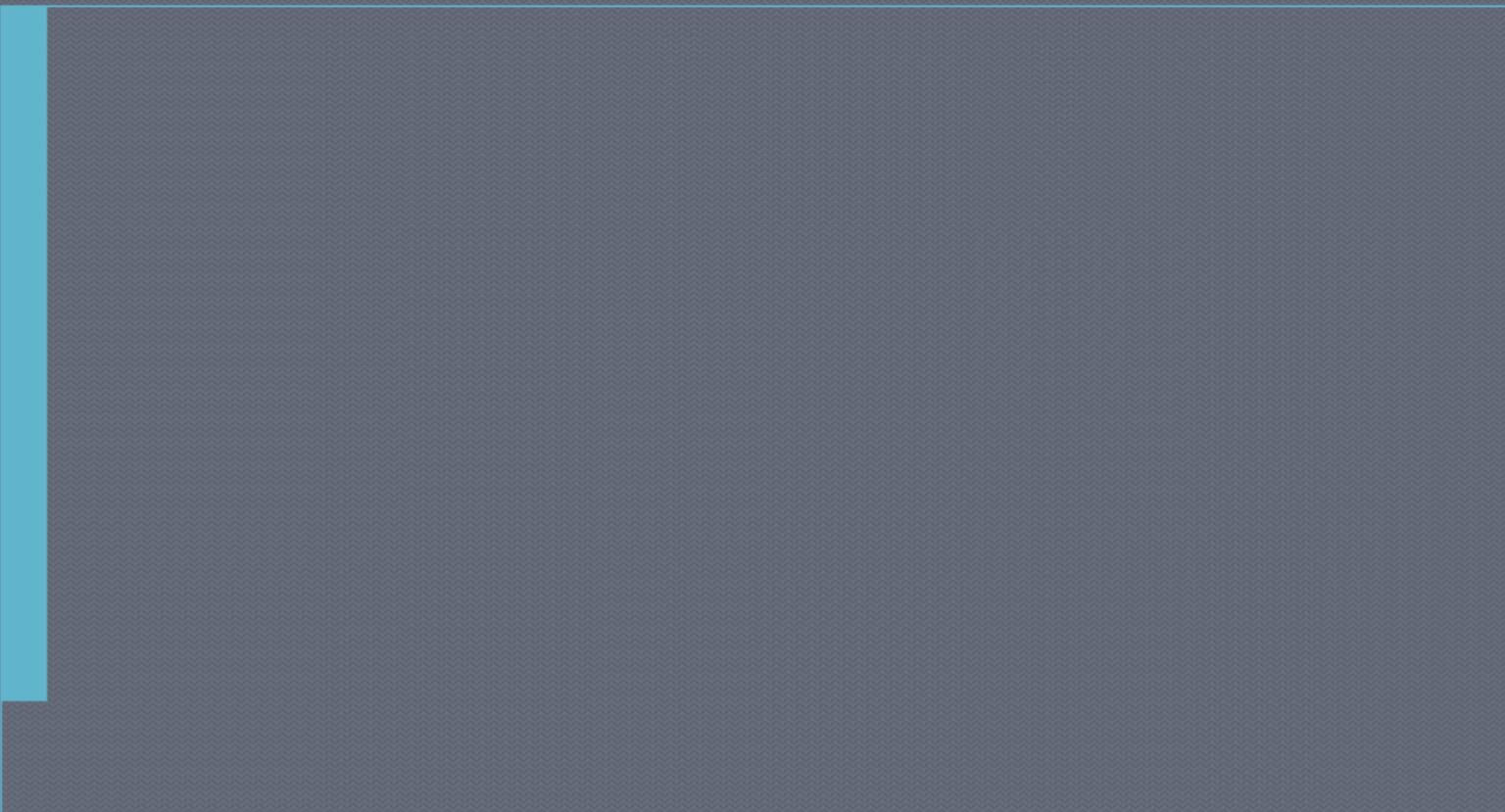


- ▶ `str()` : 문자형(str)으로 형변환(<str>)하는 함수

```
a = 3.14  
b = str(a)  
c = str(521)
```



input(), split(), map() 정리



입력함수 `input()` 정리

`input()`함수의 특성

- 무조건 문자로 입력됨.
- 줄단위로 입력 됨.

입력데이터

apple
lemon

입력문

"apple"
a = input()
b = input()
"lemon"

123
-35

"123"
a = input()
b = input()
"-35"

123
-35

"123"
a = int(input())
b = int(input())
"-35"

입력결과

a "apple"

a 'a' 'p' 'p' 'l' 'e'
[0] [1] [2] [3] [4]

b "lemon"

b 'l' 'e' 'm' 'o' 'n'
[0] [1] [2] [3] [4]

a "123"

a '1' '2' '3'
[0] [1] [2]

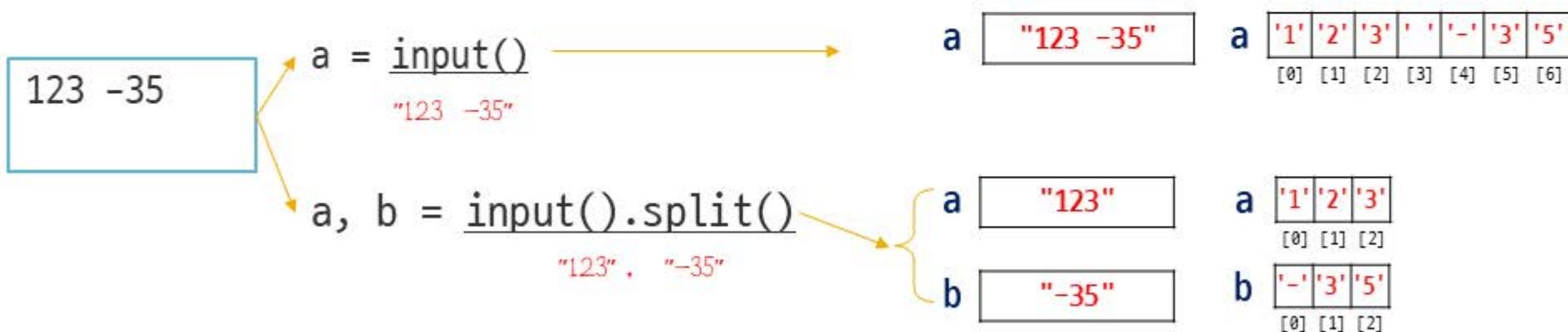
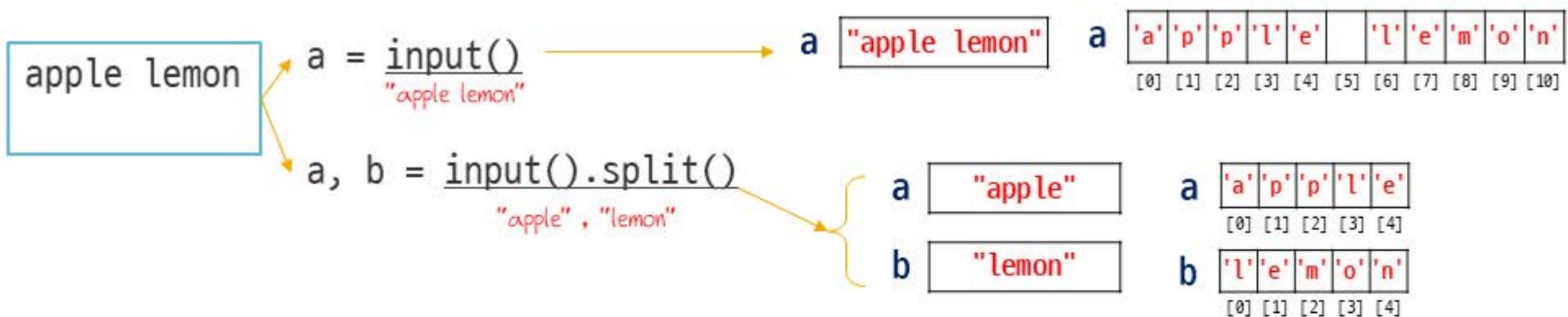
b "-35"

b '-' '3' '5'
[0] [1] [2]

a 123

b -35

입력함수 `input()`, 문자열 분리 메소드함수 `split()`



여러개 값을 1개씩 함수에 적용해주는 **map()** 함수

123 -35

a, b = input().split()

"123", "-35"

a "123"

b "-35"

a ['1' '2' '3']
[0] [1] [2]

b ['-3' '5']
[0] [1] [2]

a, b = int(input().split()) (오류)

"123", "-35"

a = int("123")

a, b = int("123", "-35") (X)

a, b = map(int, input().split())

"123", "-35"

a 123

b -35

123

int("123")

"123"

-35

int("-35")

"-35"

입력데이터

apple
lemon

입력문

a = input()
b = input()
"lemon"

123
-35

a = input()
b = input()
"-35"

123
-35

a = int(input())
b = int(input())
"-35"

lime kiwi

a = input()
"lime kiwi"

a, b = input().split()
"lime kiwi".split()
"lime" 과 "kiwi"

입력결과

a [] → 'a' 'p' 'p' 'l' 'e'
[0] [1] [2] [3] [4]

b [] → 'l' 'e' 'm' 'o' 'n'
[0] [1] [2] [3] [4]

a [] → '1' '2' '3'
[0] [1] [2]

b [] → '-' '3' '5'
[0] [1] [2]

a [] → 123

b [] → -35

a [] → 'l' 'i' 'm' 'e' 'k' 'i' 'w' 'i'
[0] [1] [2] [3] [4] [5] [6] [7] [8]

{ a [] → 'l' 'i' 'm' 'e'
[0] [1] [2] [3]

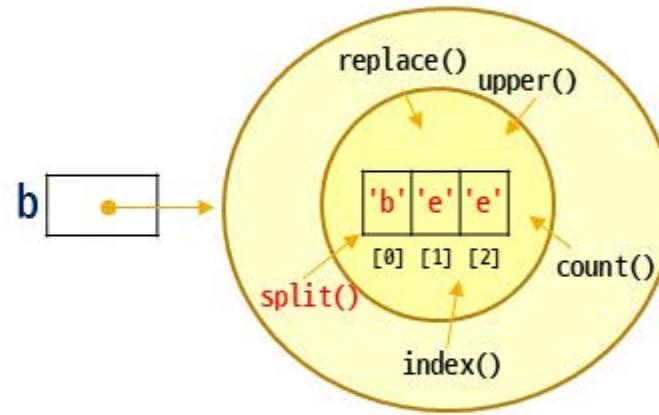
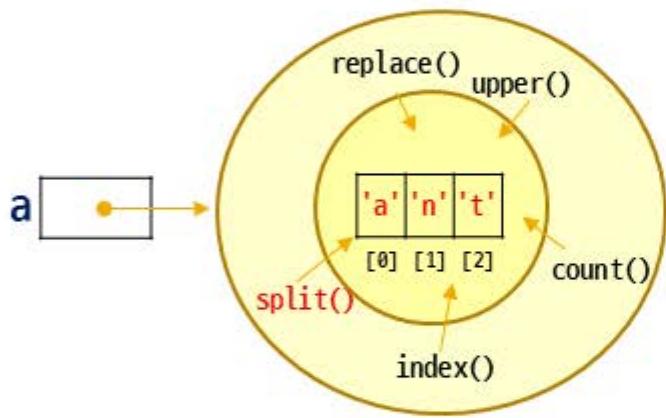
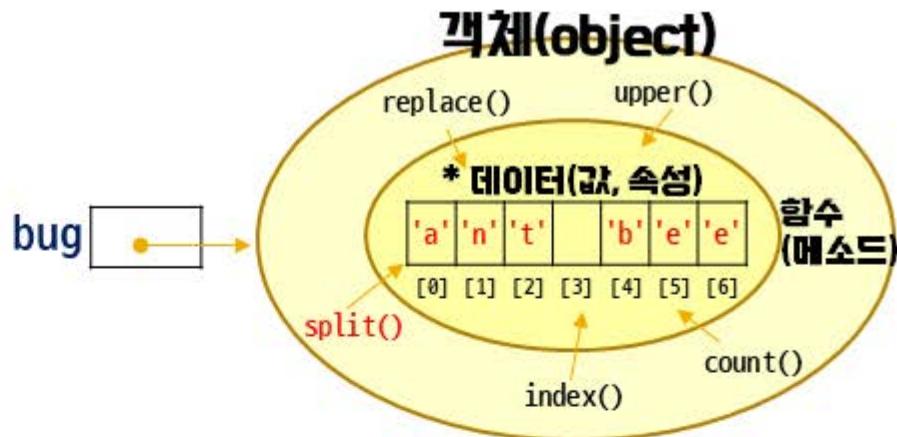
b [] → 'k' 'i' 'w' 'i'
[0] [1] [2] [3]

```

bug = "ant bee"

a, b = bug.split()
      "ant bee"
      "ant" 과 "bee"

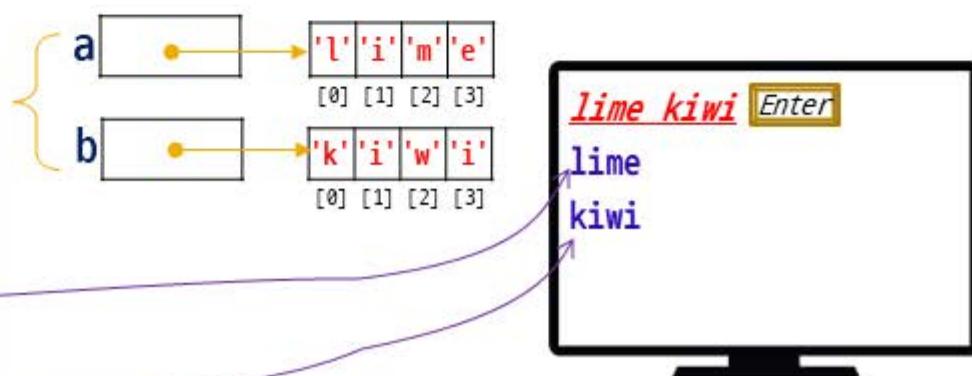
```



```

a, b = input().split()
      "lime kiwi"
      "lime kiwi".split()
      "lime" 과 "kiwi"
print(a)
print(b)

```



여러개 값을 1개씩 함수에 적용해주는 **map()** 함수

123 -35

a, b = input().split()

"123", "-35"



a, b = int(input().split()) (오류)

"123" 과 "-35"

a = int("123")
a, b = int("123", "-35") (X)

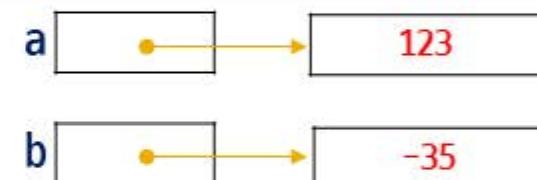
a, b = map(int, input().split())

"123" 과 "-35"

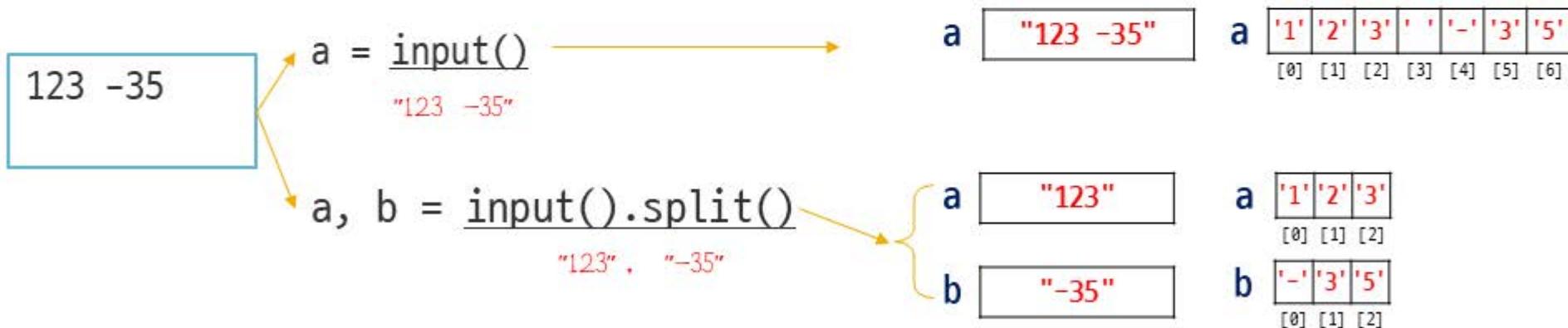
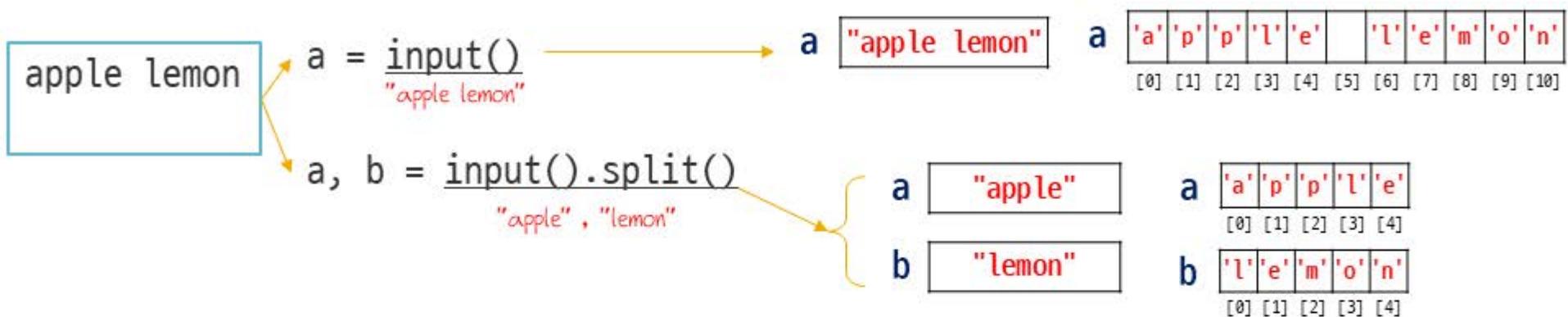
mapping int("123") int("-35")

123

-35



입력함수 `input()`, 문자열 분리 메소드함수 `split()`



아래 코드의 차이점

```
a = "123"
```

```
r1 = int(a[0]) + int(a[1]) + int(a[2])
```

```
r2 = int(a[0] + a[1] + a[2])
```

```
print(r1, r2)
```



문제해결과제



나머지 연산자 : %(modulo, MOD)

$$\begin{array}{r} q \\ \hline b \overline{) a} \\ b q \\ \hline r \end{array}$$

$$a = b q + r \quad (0 \leq r < b)$$

$$q = a // b$$

$$r = a \% b$$

0 % 5	$\rightarrow 0$
1 % 5	$\rightarrow 1$
2 % 5	$\rightarrow 2$
3 % 5	$\rightarrow 3$
4 % 5	$\rightarrow 4$
5 % 5	$\rightarrow 0$
6 % 5	$\rightarrow 1$
7 % 5	$\rightarrow 2$
8 % 5	$\rightarrow 3$
9 % 5	$\rightarrow 4$
10 % 5	$\rightarrow 0$
11 % 5	$\rightarrow 1$
...	

최소동전개수

```
money=int(input())
```

```
a=(money//500)
```

```
money2=(money-500*a)
```

```
b=(money2//100)
```

```
money3=(money2-100*b)
```

```
c=(money//50)
```

```
money4=(money3-50*c)
```

```
d=(money4//10)
```

```
print(a+b+c+d)
```

최소동전개수

```
money=int(input())
```

```
a=(money//500)
```

```
money2=(money-500*a)
```

```
b=(money2//100)
```

```
money3=(money2-100*b)
```

```
c=(money//50)
```

```
money4=(money3-50*c)
```

```
d=(money4//10)
```

```
print(a+b+c+d)
```

```
money=int(input())
```

```
a=(money//500)
```

```
b=(money%500//100)
```

```
c=(money%500%100//50)
```

```
d=(money%500%100%50//10)
```

```
print(a+b+c+d)
```

최소동전개수

```
money=int(input())  
  
a=(money//500)  
  
b=(money%500//100)  
  
c=(money%500%100//50)  
  
d=(money%500%100%50//10)  
  
print(a+b+c+d)
```

```
money=int(input())  
  
a=(money//500)  
  
b=(money%500//100)  
  
c=(money%100//50)  
  
d=(money%50//10)  
  
print(a+b+c+d)
```

최소동전개수 구하기

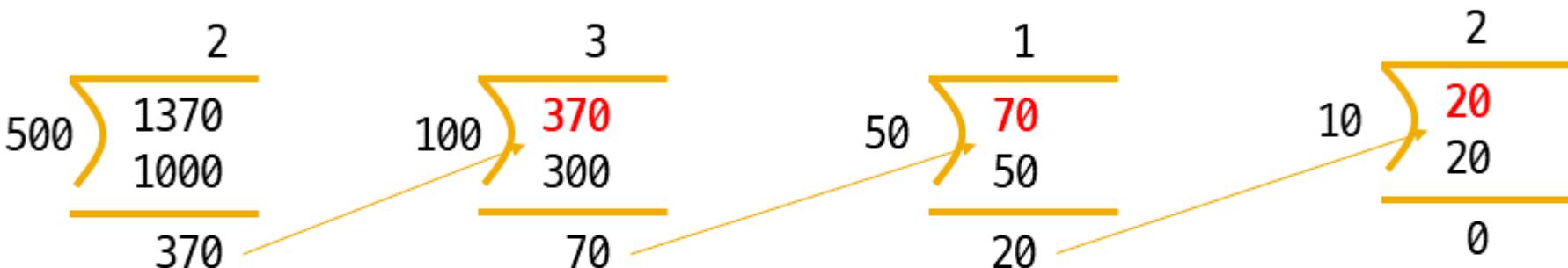
- ▶ 거스름돈(1370원이라면) → 최소동전 개수? 8개

500원

100원

50원

10원



$$c500 = a // 500$$

$$c100 = (a \% 500) // 100$$

$$c50 = (a \% 100) // 50$$

$$c10 = (a \% 50) // 10$$

=====

a%500%100

=====

a%500%100%50

a - 500*

W요일에서 D일 후는 무슨 요일1?

일	월	화	수	목	금	토
0	1	2	3	4	5	6

(W + D) % ?

0	1	2	3	4	5	6
0	1	2	3	4	5	6
7	8	9	10	11	12	13
0	1	2	3	4	5	6
14	15	16	17	18	19	20
0	1	2	3	4	5	6
21	22	23	24	25	26	27
0	1	2	3	4	5	6

$$r = a \% b$$

$$(0 \leq r < b)$$

W요일에서 D일 후는 무슨 요일2?

일	월	화	수	목	금	토	(W + D) % 7
0	1	2	3	4	5	6	

일	월	화	수	목	금	토
1	2	3	4	5	6	7

0	1	2	3	4	5	6
1	2	3	4	5	6	7
7	8	9	10	11	12	13
1	2	3	4	5	6	7

W요일에서 D일 후는 무슨 요일2?

일	월	화	수	목	금	토
0	1	2	3	4	5	6
??	??			(W - 1 + D) % 7 + 1		

일	월	화	수	목	금	토
1	2	3	4	5	6	7

0	1	2	3	4	5	6
1	2	3	4	5	6	7
7	8	9	10	11	12	13
1	2	3	4	5	6	7

1년 ~ N년까지 윤년은 몇번?

윤년 : 2월이 29일까지 있는 년도, 윤년인 해는 1년이 366일이다.

윤년의 기준

가. 4의 배수는 윤년

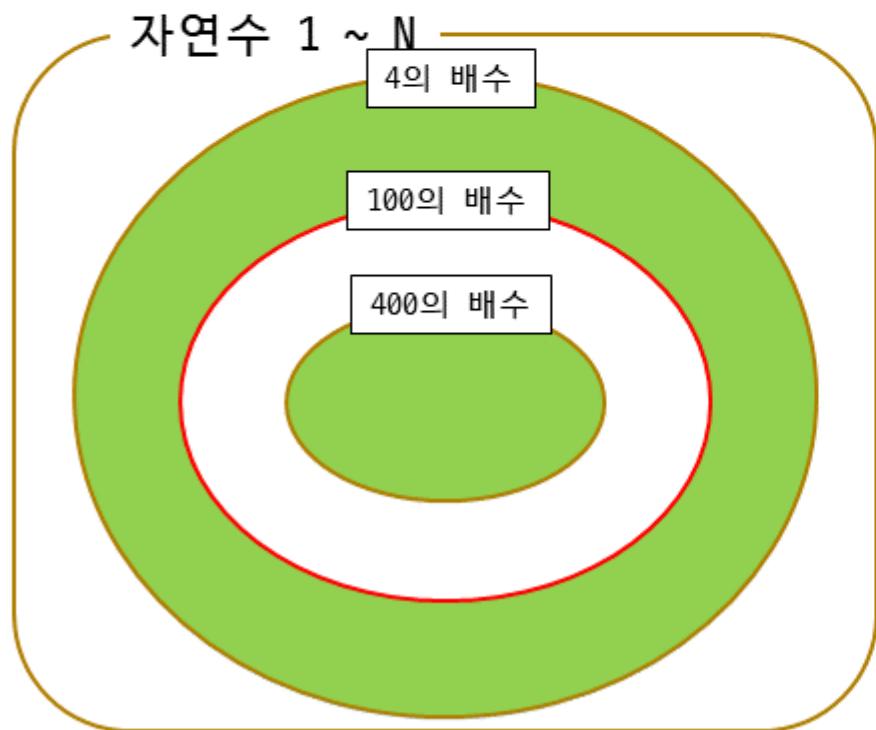
$$4\text{의 배수} = \{4, 8, 12, 92, 96, 100, 104, 108, \dots\}$$

나. 100의 배수는 윤년 아님

$$100\text{의 배수} = \{100, 200, 300, 400, \dots\}$$

다. 400의 배수는 윤년

$$400\text{의 배수} = \{400, 800, 1200, 1600, \dots\}$$



$$N\text{년까지 } 4\text{의 배수} = N // 4$$

$$N\text{년까지 } 100\text{의 배수} = N // 100$$

$$N\text{년까지 } 400\text{의 배수} = N // 400$$

문자(열) 데이터 다루기 (문자상수 and 문자변수)

분리(split) – 특정 문자를 기준으로 분리

"apple kiwi".split() (1개)가

"apple" "kiwi" (2개)로 분리

자르기(slicing) – 특정 위치로 문자(열) 분리

"apple kiwi"[0:5] "apple kiwi"[6:10]

'a'	'p'	'p'	'l'	'e'	' '	'k'	'i'	'w'	'i'
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

```
a = input()  
"apple kiwi"
```

```
print(a)
```

apple kiwi Enter

Apple kiwi

```
a, b = input().split()
```

"apple kiwi"

"apple"

a ← "apple"
b ← "kiwi"

```
print(a)
```

```
print(b)
```

apple kiwi Enter

apple

kiwi

```
kor, eng = input().split()
```

kor ← "85"

eng ← "90"

```
sum = kor + eng
```

```
print(sum)
```

85 90 Enter

8590

```
kor, eng = map(int, input().split())
```

kor ← int("85") ← "85"

eng ← int("90") ← "90"

a, b = int("10", "20") # 오류

85 90 Enter

175 Enter

```
sum = kor + eng
```

```
print(sum)
```

```
s = input() s 'H' 'e' 'l' 'l' 'o' 'W' 'o' 'r' 'l' 'd'  
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]  
[-11] [-10] [-9] [-8] [-7] [-6] [-5] [-4] [-3] [-2] [-1]
```

```
print(s[0], s[10])
```

Hello World Enter

H d

d d

Hello Hello

World World

HloWrd

```
print(s[0:5], s[:5])
```

```
print(s[6:11], s[6:])
```

```
print(s[::-2])
```

```
w, h = 70.0, 170.5
```

```
bmi1 = w/((h/100)*(h/100))
```

```
bmi2 = w/((h*h)/10000)
```

24.079600278635372

24.079600278635375

24.080

24.080

```
print(bmi1) # 24.079600278635372
```

```
print(bmi2) # 24.079600278635375
```

```
print("%.3f"%(bmi1)) # 소수점 이하 3자리까지 출력
```

```
print("%.3f"%(bmi2))
```

실습) 아래 코드를 실행하시오.

문자(열) 분리 : 특정문자를 기준으로 분리
형식 : 문자데이터.split()

s = "Hello World"

s [H][e][l][l][o] [W][o][r][l][d]
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]

a, b = s.split() # 변수값 분리
print(a, "그리고", b)

a [H][e][l][l][o] b [W][o][r][l][d]
[0] [1] [2] [3] [4] [0] [1] [2] [3] [4]

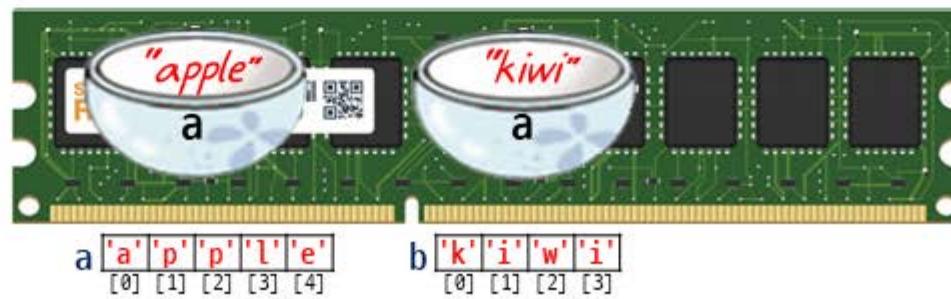
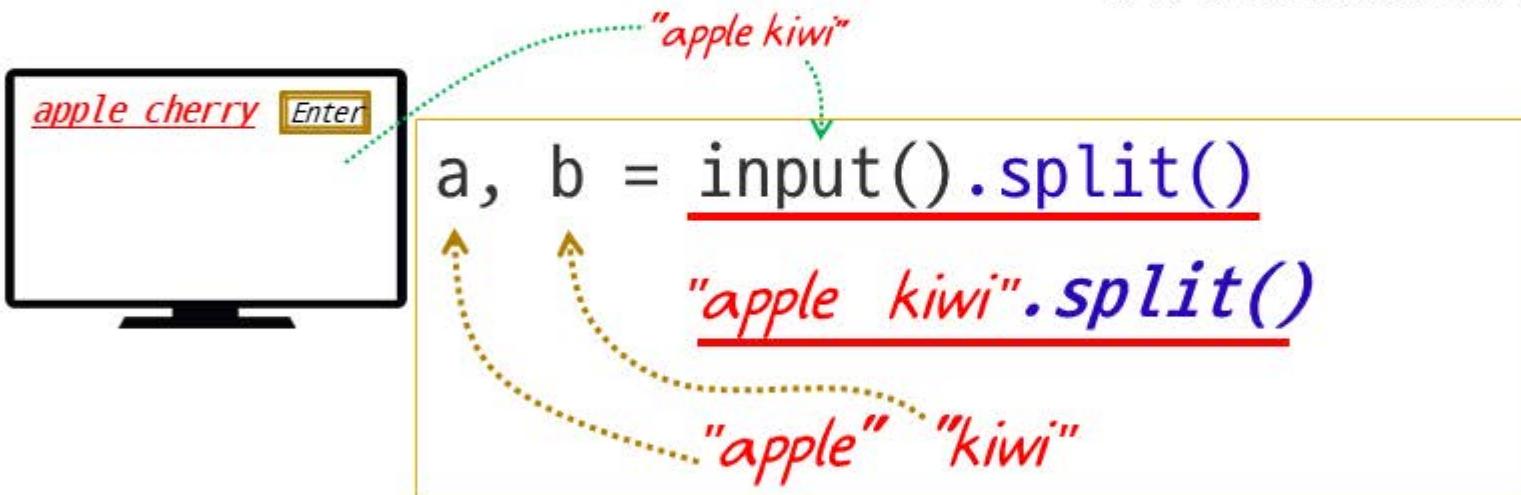
c, d = "apple kiwi".split() # 상수값 분리
print(c, "그리고", d)

a [a][p][p][l][e][k][i][w][i]
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9]
c [H][e][l][l][o] d [k][i][w][i]
[0] [1] [2] [3] [4] [0] [1] [2] [3]

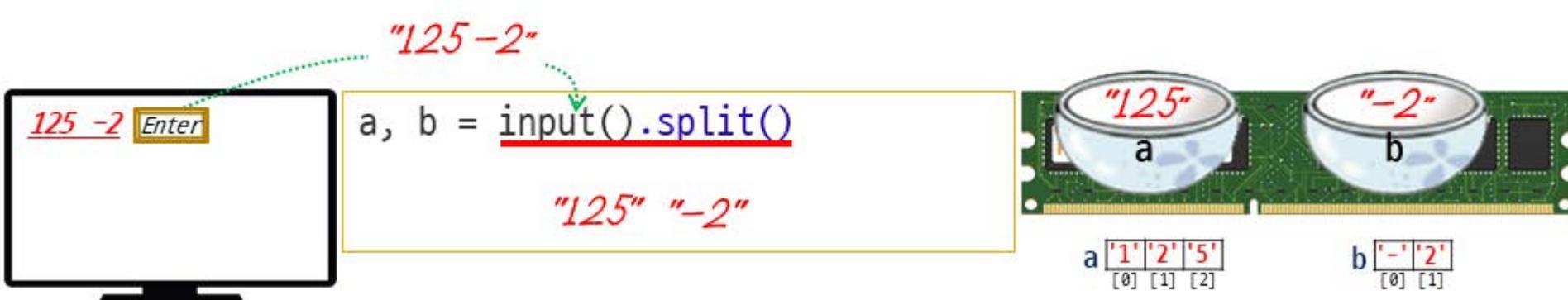
e, f, g = "2022-3-21".split('-') # (-으로 분리)
print(e, "그리고", f, "그리고", g)

e [2][0][2][2][-][3][-][2][1]
[0] [1] [2] [3] [4] [5] [6] [7] [8]
f [3] g [2][1]
[0] [0] [1]

한줄에 입력된 다수 데이터 분리(split) 하기



split() 메소드



map() 함수

"125 -2 83"

오류

125 -2 83

Enter

a, b, c = int(input().split())

"125", "-2", "-83"

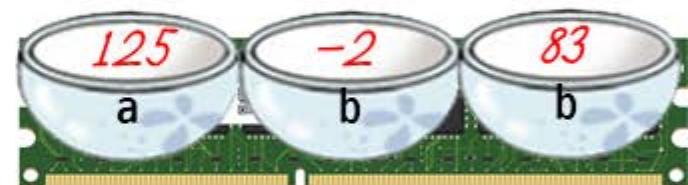
int(x) : 1개의 값만 형변환, int("125", "-2", "-83")

a, b, c = input().split()

a = int(a)

b = int(b)

c = int(c)



"125 -2 83"

125 -2 83

Enter

a, b, c = map(int, input().split())

a ← int("125") "125"

b ← int("-2") "-2"

c ← int("83") "83"
mapping

실습) 아래 코드를 실행하시오.

슬라이싱 : 문자데이터에서 a번부터 b-1번까지 잘라내기

형식 :

문자[a] : a번째 문자 자르기

문자데이터[a:b] : a~b-1번째 문자 자르기

```
s = "Hello World"
```

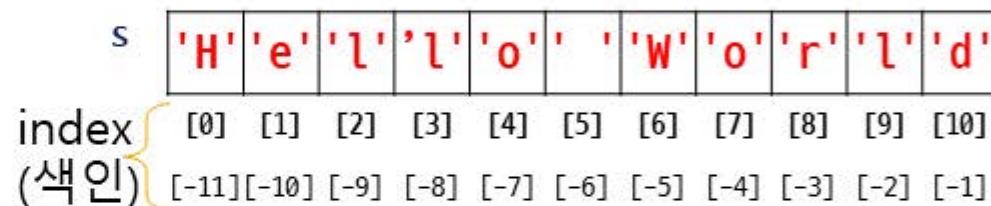
```
print(s[0], s[6])
```

```
print(s[0:5], s[:5])
```

```
print(s[6:11], s[6:])
```

```
print(s[10], s[-1])
```

```
print(s[::-2])
```



```
H W  
Hello Hello  
World World  
d d  
HloWrld
```

문자데이터 분리, 문자데이터 슬라이싱

분리 형식 : 문자데이터.split()

s = "Hello World"

s [H][e][l][l][o] [W][o][r][l][d]
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]

a, b = s.split()

a [H][e][l][l][o] b [W][o][r][l][d]
[0] [1] [2] [3] [4] [0] [1] [2] [3] [4]

~~a, b = 123456.split()~~ 오류(숫자는 split() 불가)

슬라이싱 형식 : 문자데이터[0:5]

a = "12345"[0:3]

a [1][2][3]
[0] [1] [2]

~~b = 12345[0:3]~~ 오류(숫자는 슬라이싱 불가)

분리(split()) or 자르기(slicing)

031225-4123456

4

a, b = input().split()

a ['0' | '3' | '1' | '2' | '2' | '5']
[0] [1] [2] [3] [4] [5]

b ['4' | '1' | '2' | '3' | '4' | '5' | '6']
[0] [1] [2] [3] [4] [5] [6]

print(_____)

031225-4123456

4

a, b = input()

a ['0' | '3' | '1' | '2' | '2' | '5' | '-' | '4' | '1' | '2' | '3' | '4' | '5' | '6']
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13]

print(_____)

분리(split()) or 자르기(slicing)

2023-03-20

03 20

a, b, c = input().split()

a

'2'	'0'	'2'	'3'
[0]	[1]	[2]	[3]

 b

'0'	'3'
[0]	[1]

 c

'2'	'0'
[0]	[1]

print(b, c)

2023-03-20

03 20

a = input()

a

'2'	'0'	'2'	'3'	'-'	'0'	'3'	'-'	'2'	'0'
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

print(a[5:7], a[8:10])

분리(split()) or 자르기(slicing)

20230320

03 20

```
a, b, c = input().split()
```

```
print(b, c)
```

20230320

03 20

```
a = input()
```

```
a ['2' '0' '2' '3' '0' '3' '2' '0'  
[0] [1] [2] [3] [4] [5] [6] [7]
```

```
print(a[???:??], a[???:??])
```

031225-4123456

4

a, b = input().split()

a ['0' '3' '1' '2' '2' '5'
[0] [1] [2] [3] [4] [5]

b ['4' '1' '2' '3' '4' '5' '6'
[0] [1] [2] [3] [4] [5] [6]

print(_____)

031225-4123456

4

a, b = input()

a ['0' '3' '1' '2' '2' '5' '-' '4' '1' '2' '3' '4' '5' '6'
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13]

print(_____)

$f(x)$

함수 (Function)

print(x)

function : (사람사물의) 기능, 행사, 의식, (수학의) 함수

어떤 특정 기능을 갖는 명령어 집합

* 함수의 예

`print()` 데이터를 출력하는 기능

```
print("Hello World")
```

`input()` 데이터를 입력하는 기능

```
a = input()
```

`int(), float(), str()` 데이터를 형변환하는 기능

```
a = int("2")
```

`round()` 데이터를 반올림하는 기능

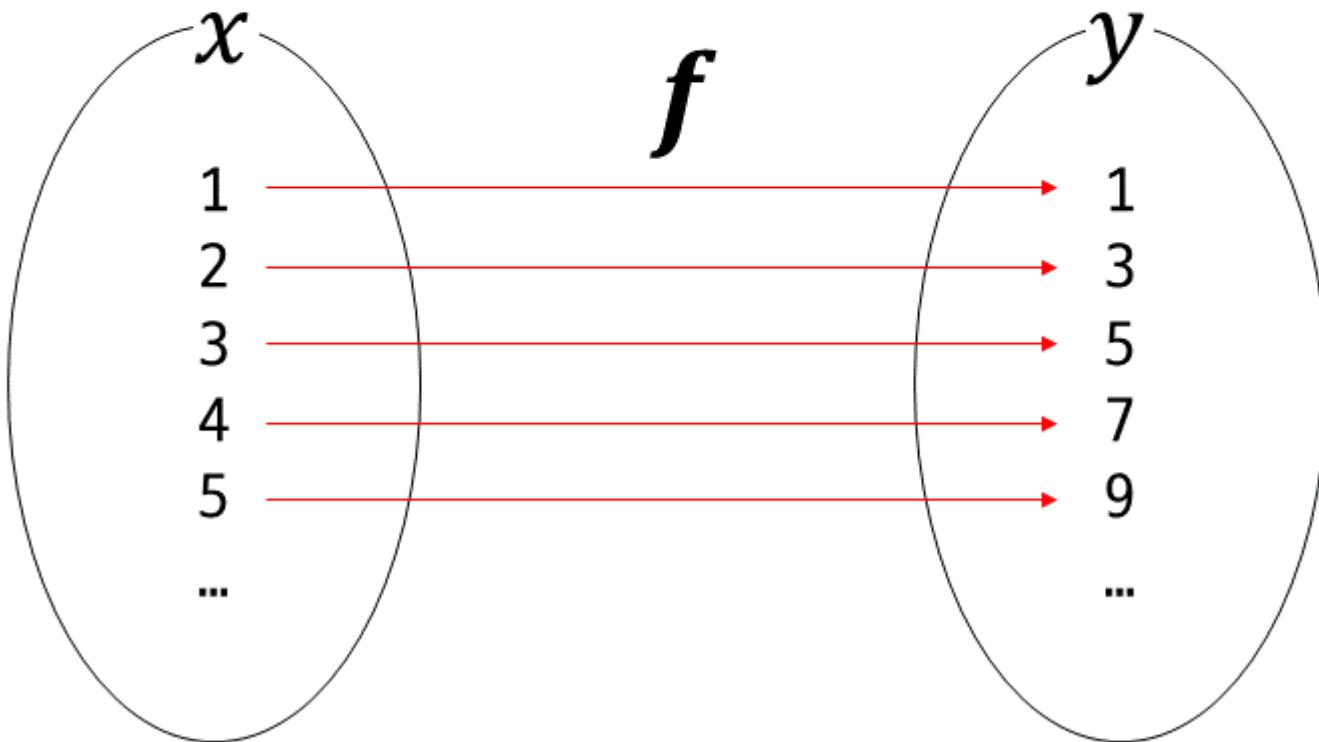
```
a = round(1.256, 2)
```

`len()` 데이터의 길이/개수를 구하는 기능

```
a = len("hello")
```

$f(x)$ 를 정의하시오...

$$f(x) = ?$$

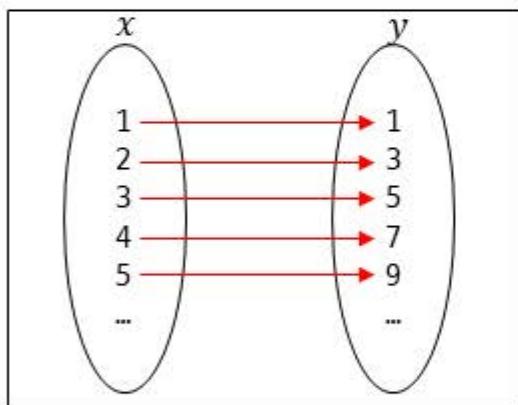


x 번째 홀수를 구하는 함수

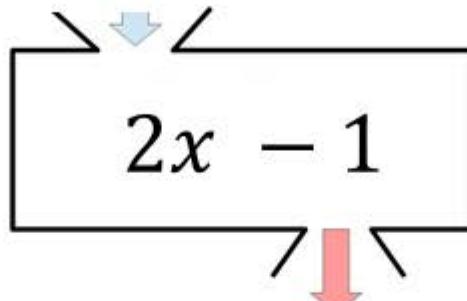
수학에서
함수의 정의

$$y = f(x) = 2x - 1$$

x 는 자연수

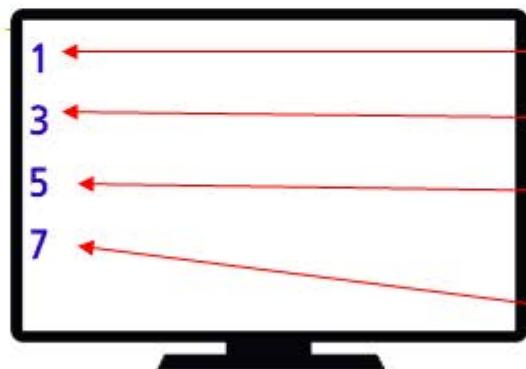


입력 x



파이썬에서
함수의 정의

```
def f(x) :  
    ???  
    ???
```



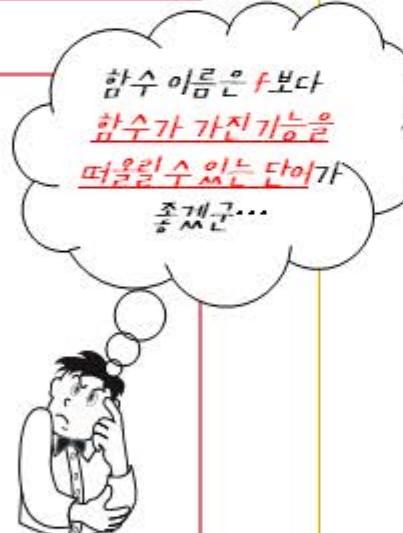
```
print( f(1) )
print( f(2) )
print( f(3) )
a = f(4)
print( a )
```

x 번째 홀수를 구하는 함수의 이름은???

무의미한 이름은
피할 것

```
def f(x) :  
    y = 2 * x - 1  
    return y
```

```
print( f(1) )  
print( f(2) )  
print( f(3) )  
print( f(4) )  
print( f(5) )  
  
a = f(99)  
print( a )
```



기능과 관련된
이름으로

```
def odd(x) :  
    y = 2 * x - 1  
    return y
```

```
print( odd(1) )  
print( odd(2) )  
print( odd(3) )  
print( odd(4) )  
print( odd(5) )  
  
a = odd(99)  
print( a )
```

식별자(identifier)의 이름규칙
* 식별자: 변수명, 함수명 등

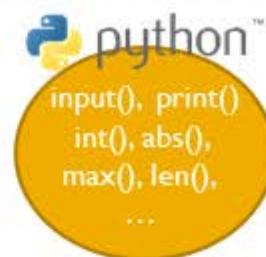
- 알파벳, 숫자, _의 조합
- 숫자로 시작할 수 없음
- 예약어 사용 금지



파이썬에서 함수의 종류

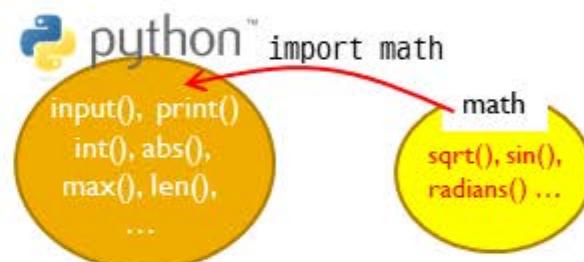
▶ 내장함수(built-in 함수)

- ▶ 이미 완성되어 있는 함수
- ▶ 파이썬 내에 이미 포함되어 있어서 제약없이 사용



▶ 외부함수

- ▶ 이미 완성되어 있는 함수
- ▶ 파이썬 내에 포함되어 있지 않아서 포함시켜서 사용



▶ 사용자 정의 함수

- ▶ 사용자가 직접 정의해서 사용하는 함수

```
def f(x) :  
    y = 2 * x - 1  
    return y
```

▶ 메소드(함수)

- ▶ 특정개체에서만 사용할 수 있는 함수

내장함수(built-in function)

내장함수 : 이미 정의된 함수, 파이썬 내에 포함된 함수

* 입력 / 출력 함수

`input()` : 입력 가능, `print()` : 출력 가능

* 형(type) 변환 함수

`int()` : 정수로 형변환 가능, `float()` : 실수로 형변환 가능, `str()` : 문자로 형변환 가능

* 수치처리 함수

`abs()` : 절대값 가능, `round()` : 반올림 가능, `max()` : 최대값 가능, `min()` : 최소값 가능,

* 문자(열) 함수

`len()` : 문자열 길이 구하는 기능, `eval()` : 문자로 된 수식을 수치계산 하는 기능

`ord()` : 문자의 유니코드 번호 구하는 기능, `chr()` : 숫자에 해당되는 유니코드 문자를 구하는 기능

* 기타 : `map()`, `type()`, `id()`, `dir()`, `list()` 등 주요 함수들이 더 있음...

```
x = 2.782
```

```
print( int(x) )  
print( abs(-15) )
```

```
print( max(15, 7, 16, 3, 8) )  
print( min(10, -7, 6, 32) )
```

```
print( round(0.3456, 1), round(0.3456, 2) )
```

```
print( eval("2 * int(x) + 3") )  
print( len("Hello World") )  
print( len( str(x) ) )
```

```
print( ord('A'), ord('\n'), chr(97), chr(44032) )
```

```
2  
15  
16  
-7  
0.3 0.35  
7  
11  
5  
65 10 a 가
```

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	.
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	,	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j

```
name = "홍길동"  
age = 17  
height = 158.6
```

```
print(type(name), type(age), type(height))
```

```
print(type("충곽"), type(29), type(3.14))
```

```
<class 'str'> <class 'int'> <class 'float'>  
<class 'str'> <class 'int'> <class 'float'>
```

외부함수(module), 표준 모듈, 외부 모듈

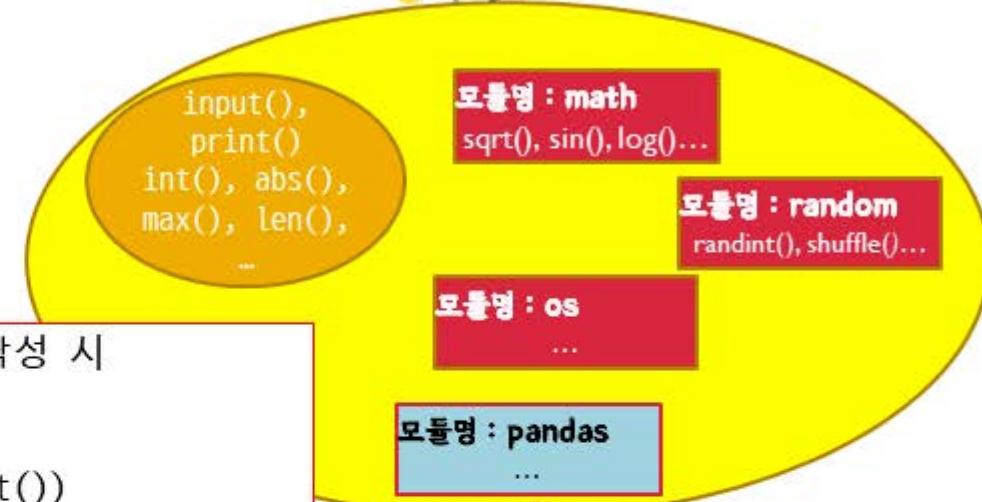
- * module : 함수, 클래스 등을 모은 파일

(모듈을 묶어서 패키지(package)라 함.)

가. 표준모듈(standard module)

- * 파이썬에 내장된 모듈
- * import 해야 사용 가능
- * ex) math, random, os ...

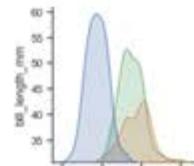
```
# 프로그램 작성 시  
import math  
  
a = int(input())  
  
r = math.sqrt(a)  
  
print(r)
```



pip install pandas
외부모듈 pandas 설치

나. 외부모듈(3rd party module)

- * 파이썬에 설치 안됨, 필요시 설치(pip)해서 사용
- * 설치 후 import 해야 함.
- * ex) pandas, seaborn, matplotlib, scikit-learn ...



모듈명 : pandas
각종 객체, 함수 등으로 구성

참고



기본 파이썬 + 데이터분석에 필요한
외부모듈이 한꺼번에 설치됨

내장(built-in)함수 – 이미 정의된 함수

▶ 내장함수 : 이미 정의되어 있어서 사용만 하면 되는 함수

- ▶ `print()` : 데이터를 출력하는 기능
- ▶ `input()` : 데이터를 입력하는 기능
- ▶ `int(), float(), str()` : 데이터를 형변환하는 기능
- ▶ `round(실수, 자리수)` : 실수를 반올림하는 기능
- ▶ `eval(문자계산식)`
- ▶ `len(문자열 or 리스트)` : 문자열의 길이, 리스트의 원소 개수 기능
- ▶ `chr(정수)` : ASCII숫자 → 문자
- ▶ `ord(문자)` : ASCII문자 → 숫자
- ▶ 예)

~~def print(x) :~~
~~블라블라…~~
~~여서구저여구…~~
~~x를 출력해라…~~

```
x = 2.782
print( int(x) ) # 소수점 이하 버림(x - 2)
print( abs(-15) ) # 절대값
print( round(0.3456, 1), round(0.3456, 2) ) #반올림

print( eval("2 * int(x) + 3") ) # 연산식 계산
print( len("Hello World") ) # 문자열의 길이
print( len( str(x) ) ) # 문자열의 길이

print( ord('A'), chr(97) )
```

2
15
0.3 0.35
7
11
5
65 a

문자함수

- ▶ `ord(c)` : 문자 c의 ASCII 번호
- ▶ `chr(n)` : 자연수 n에 해당되는 ASCII 문자

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	'
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	.	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	:	91	5B	[123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	-	127	7F	[DEL]

4. 문자함수

- `ord(c)` : 문자 c의 ASCII 번호
- `chr(n)` : 정수 n에 해당되는 ASCII 문자

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	10진수 번호	16진수 번호	문자	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	'
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42		98	62	b
3	3	[END OF TEXT]	35	23	#	67	43		99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44		100	64	d
5	5	[ENQUIRY]	37	25	%	69	45		101	65	e
6	6	[ACKNOWLEDGE]	38	26	?	70	46		102	66	f
7	7	[BELL]	39	27		71	47		103	67	
8	8	[BACKSPACE]	40	28		72	48		104	68	
... 중간 생략 ...											
24	18	[CANCEL]	56	3A	:	91	5D		127	7F	[DEL]
25	19	[END OF MEDIUM]	57	3B	;	92	5E				
26	1A	[SUBSTITUTE]	58	3C	<	93	5F				
27	1B	[ESCAPE]	59	3D	=	94					
28	1C	[FILE SEPARATOR]	60	3E	>	95					
29	1D	[GROUP SEPARATOR]	61	3F	?	96					
30	1E	[RECORD SEPARATOR]	62			97					
31	1F	[UNIT SEPARATOR]	63			98					

문자 'A'의 ASCII 번호가 궁금해?

`ord('A')`라고 하면 돼...

그럼 65번이라고 알려줄꺼야...

100번 문자가 알고 싶을 땐,
`chr(100)`라고 하면 되겠군...

그럼 'd'가 나오겠지???



문자열 객체의 메소드(함수)

메소드 : 특정 객체에 속한 함수

- **메소드(Method)**
 - 특정 객체에 포함된 함수
 - 파이썬에서는 모든 데이터가 객체
- **메소드 함수의 사용 예**
객체.메소드함수()
- **문자(열) 객체의 메소드함수**
 - "Hello World".split() # 문자열 분리 --> Hello와 World로
 - "Hello".upper() # 문자열 대문자로 --> HELLO
 - "Hello".count('l') # 검색문자 개수 --> 2
 - "Hello".index('e') # 검색문자 위치 --> 1
 - "Hello".replace('l', 'L') # 다른 문자로 대치 --> HeLLo

문자(열)
객체
string

count()
index()
join()
replace()
find()
split()
upper()
lower()

...

함수의 종류

▶ 내장함수(built-in)

- ▶ `print()`, `input()`, `int()`, `abs()`, `len()` ...

```
a = -15  
r = abs(a)  
print(r)
```

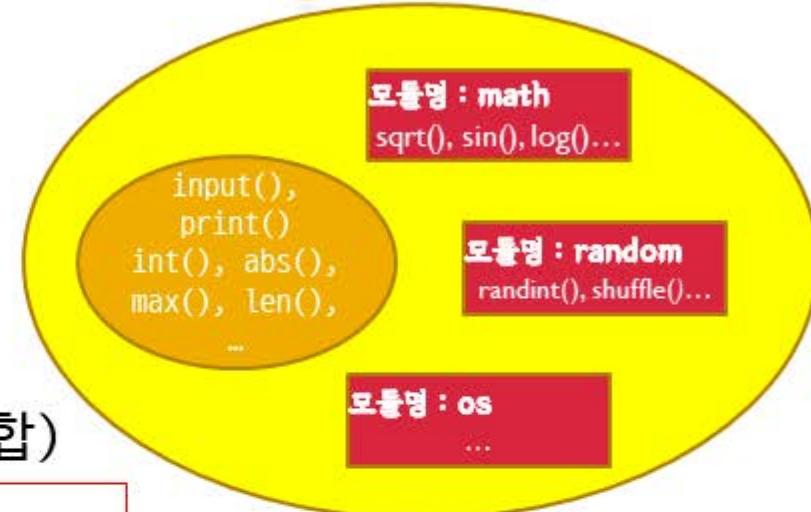
▶ 외부 모듈 함수(모듈은 관련된 함수 집합)

```
import math  
r = math.sqrt(16) # 4.0 → math 모듈에 포함된  
print(r)           sqrt() 함수로 연산하라
```

▶ 메소드(method) 함수 - 특정 객체에 속한 함수

- ▶ 사용형식 : `객체.메소드함수()`

```
a, b = "Hello World".split()  
print(a)                         → "Hello World"가 Hello//World  
print(b)                         split() 함수로 문자열을 수행하라
```



파이썬의 모든 값(데이터)는 객체

x = 10

pi = 3.14

s = "Hello"

정수형 객체

int

bit_length()

to_bytes()

bit_count()

.....

예시)

x.bit_count()

(10).bit_count()

문자(열)형 객체

<str>

count() find()

split() upper()

index() lower()

replace()

....

예시)

s.count()

"Hello".upper()

실수형 객체

<float>

as_integer_ratio()

hex() fromhex()

....

예시)

pi.hex()

(3.14).hex()

문자(열)의 (메소드)함수

- ▶ 메소드(method) 함수 – 특정 객체에 속해, 객체 내에서만 사용하는 함수
 - ▶ 사용형식 : 객체명.메소드함수()

```
s = "Hello World"
```

```
print(s.upper())
```

```
print(s.count('l'))
```

```
print(s.index('r'))
```

```
print(s.replace('l', 'L')) # 'l' → L 대체
```

```
a, b = s.split()
```

```
print(a)
```

```
print(b)
```

대문자로

'l'의 개수?

'r'의 위치?

공백을 기준으로 문자 분리

S

'H'	'e'	'l'	'l'	'o'	' '	'W'	'o'	'r'	'l'	'd'
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]

HELLO WORLD

3

8

HeLLo WorLd

Hello

World

문자(열)의
메소드(함수)

count()
index()
split()
replace()
find()
upper()
lower()

함수와 (메소드)함수

▶ 내장함수(built-in)

▶ `print()`, `input()`, `int()`, `abs()`, `len()` ...

▶ 모듈함수

```
import math  
r = math.sqrt(16) # 4.0  
print(r)
```

▶ 메소드(method) 함수 – 특정 객체에 속한 함수

▶ 사용형식 : `객체.메소드함수()`

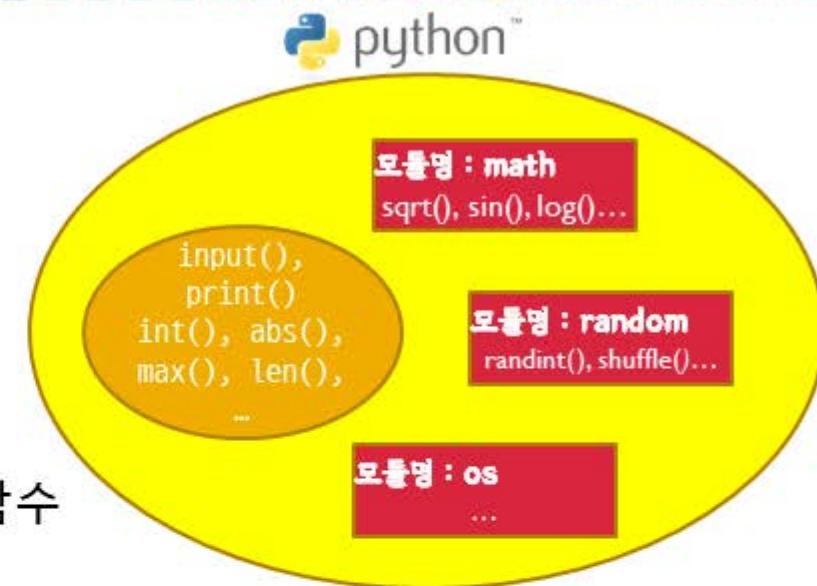
▶ 문자(열) 객체의 메소드(method) 함수 – 문자(열) 객체에서만 사용하는 함수

▶ 사용형식 : `문자(열).메소드함수()`

```
s = "Hello World"  
a, b = s.split()
```

```
print(a)  
print(b)
```

```
print(s.count('l')) # 'l'의 개수?  
print(s.index('r')) # 'r'의 위치?
```



s [H] [e] [l] [l] [o] [] [W] [o] [r] [l] [d]
[0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10]

```
Hello  
World  
0  
2
```

문자(열)의 (메소드)함수

▶ 메소드(method) 함수-특정 객체에 속해, 객체 내에서만 사용하는 함수

▶ 사용형식 : 객체명.메소드함수()

```
s = "Hello World"
```

```
s1, s2 = s.split() # 분리
```

```
print(s1)
```

```
print(s2)
```

```
print(str.upper())
```

대문자로

```
print(str.count('l'))
```

'l'의 개수?

```
print(str.index('r'))
```

'r'의 위치?

```
print(str1[4], str2[0:2]) # 슬라이스(자르기)
```



'H'	'e'	'l'	'l'	'o'	' '	'W'	'o'	'r'	'l'	'd'
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]



'H'	'e'	'l'	'l'	'o'
[0]	[1]	[2]	[3]	[4]



'W'	'o'	'r'	'l'	'd'
[0]	[1]	[2]	[3]	[4]

```
Hello  
World  
HELLO WORLD  
3  
8  
o Wor
```

문자(열)
string

count()
index()
join()
replace()
find()
split()
upper()
lower()

문자 데이터의 분리 메소드함수 – split()

- ▶ 문자열을 분리할 때 사용하는 함수

문자데이터 .split()

* 문자데이터 : 문자변수, 문자상수

▶ 사용방법

- ▶ 문자데이터 .split() → 문자데이터를 공백으로 분리

예) a, b = "Buso Mountain".split()
 # "buso", "Mountain"
 문자데이터(문자 상수)

- ▶ 문자데이터 .split('분리기호') → 문자데이터를 분리기호로 분리

예) h, m, s = "12:01:52".split(':') # "12", "01", "52"

예) m, d = "12-25".split('-') # "12", "25"

예) a, b, c = "12,13,14".split(',') # "12", "13", "14"

예) y, m, d = "2020,5,01".split(',') # "2020", "5", "01"