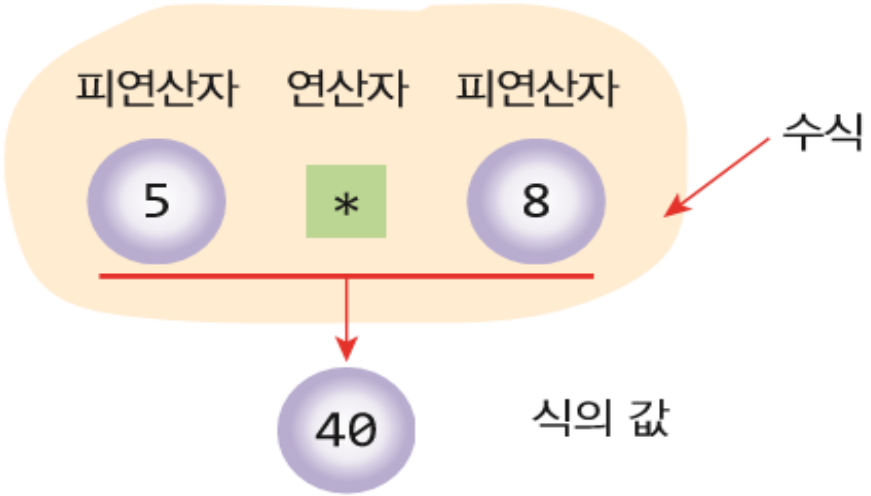
연산자로 계산하자

목차

연산자와 피연산자

1. 수식(expression) = 피연산자들과 연산자들의 모임
2. 연산자(operator): 연산을 나타내는 기호
3. 피연산자(operand): 연산의 대상이 되는 값



산술 연산자

덧셈, 뺄셈, 곱셈, 나눗셈, 나머지 연산 등

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

산술 연산자

```
a = 2
b = 3

print("a + b =" , a + b)
print("a - b =" , a - b)
print("a * b =" , a * b)
print("a / b =" , a / b)
print("a // b =" , a // b)
print("a % b =" , a % b)
print("a ** b =" , a ** b)      # 거듭제곱연산자
```

```
a + b = 5
a - b = -1
a * b = 6
a / b = 0.6666666666666666
a // b = 0
a % b = 2
a ** b = 8
```


Day_03_01.py

도전

사용자로부터 (키보드로 부터, 표준입력 방식)으로
2자리 정수를 입력 받아 (예 12)
원래의 값 (예 12) 을 꺼꾸로 뒤집어 2자리의 숫자 (예 21) 를 만들고

원래의 값 (예 12) 과 뒤집힌 숫자 (예 21)의 합을 구하는
프로그램을 작성하시요

```
n = int(input("두자리 정수 입력:" ))

a = n//10
b = n%10

reverse = b*10 + a

sum = n + reverse

print("원본 데이터 :", n)
print("역순 데이터 :", reverse)
print("두 수의 합 :", sum)
```

```
n = input("두 자리 숫자 입력:")

r = int(n[1])*10+int(n[0])*1

print(n)
print(r)
sum = int(n) + r_n

print(sum)
```

n(0)	n(1)
3	4

도전

키보드로 5글자를 입력 받아 꺼꾸로 출력하는
프로그램을 작성하시요

예) 입력받은 5글자 : 23456
거꾸로 출력 : 65432

data(0)	data(1)	data(2)	data(3)	data(4)
2	3	4	5	6

입력받은 5글자 : korea
거꾸로 출력 : aerok

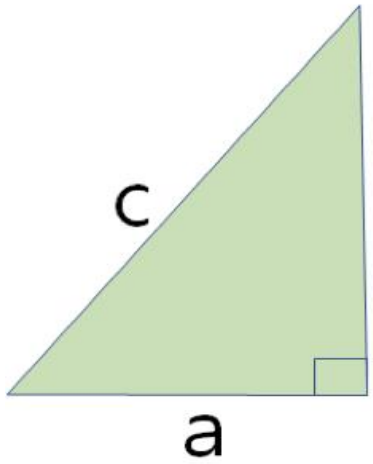
data(0)	data(1)	data(2)	data(3)	data(4)
k	o	r	e	a

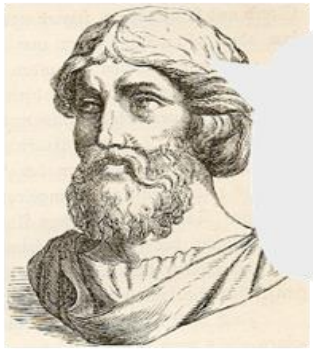
도전

```
data = input("5 글자 입력:")

r_data = data[4]+data[3]+data[2]+data[1]+data[0]

print(data)
print(r_data)
```

data(0)	data(1)	data(2)	data(3)	data(4)
k	o	r	e	a

도전





Pythagoras (Πυθαγόρας)

기원전 570년–기원전 495년

$$c^2 = a^2 + b^2$$

제가 발견한 정리라고 합니다.
정말일까요?
저의 제자가 발견했다고도 하죠.

$$c = \sqrt{a^2 + b^2} = (a^2 + b^2)^{\frac{1}{2}}$$

```
직각삼각형의 밑변를 입력하시오:6
직각삼각형의 높이를 입력하시오:8
빗변의 길이는 : 10.0
```

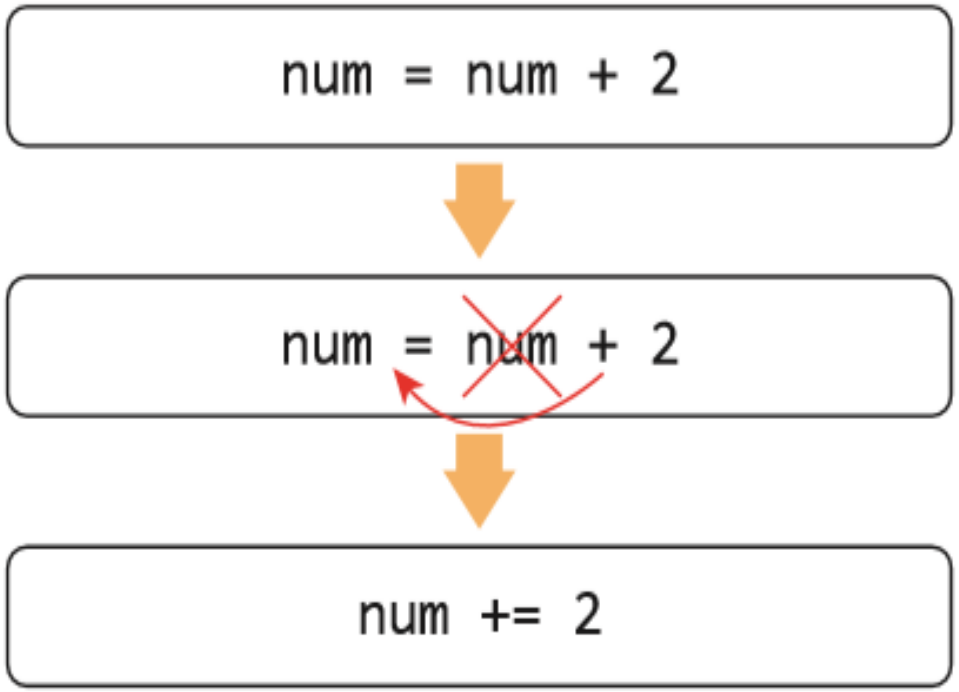
도전

```
bottom = float(input('밑변:'))
height = float(input('높이:'))

h = (bottom **2  + height ** 2) **0.5
print('빗변의 길이는 :', h)
```

$$c = \sqrt{a^2 + b^2} = (a^2 + b^2)^{\frac{1}{2}}$$

복합 (할당) 연산자

대입(할당) 연산자와 다른 연산자를 결합하여 간략하게 표현

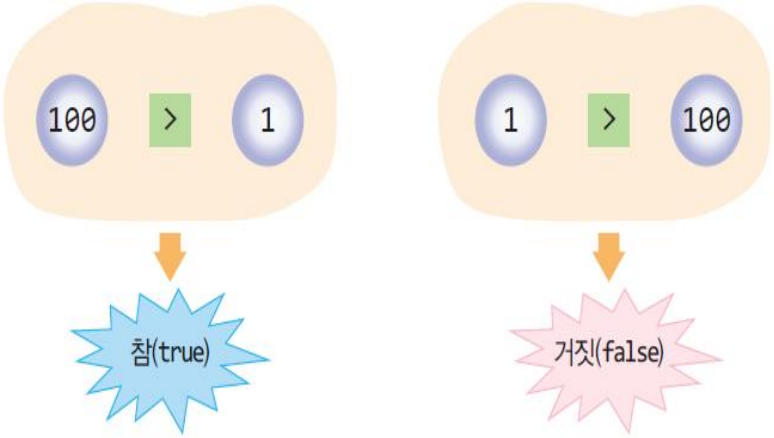


복합 연산자	의미
x += y	x = x + y
x -= y	x = x - y
x *= y	x = x * y
x /= y	x = x / y
x %= y	x = x % y

비교 연산자

◆ '크다' 혹은 '작다'와 같은 비교 연산은 수치 데이터를 담고 있는 두 개의 피연산자를 대상으로 크기 관계를 살펴본다.

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



```
a = 10
b= 7
print(a, b)
print("a >  b : " , a >  b)
print("a >=  b : " ,a >= b)
print("a <  b : " ,a <  b)
print("a <=  b : " ,a <= b)
print("a ==  b : " ,a == b)
print("a !=  b : " ,a != b)
print('-'*20)
```

```
============================
10 7
a > b :  True
a >= b :  True
a < b :  False
a <= b :  False
a == b :  False
a != b :  True
============================
```

논리 연산자

- 부울값을 가진 데이터에 대해서 적용할 수 있는 연산이 논리 연산

연산자	의미
x and y	x와 y중 거짓(False)이 하나라도 있으면 거짓이 되며 모두 참(True)인 경우에만 참이다.
x or y	x나 y중에서 하나라도 참이면 참이 되며, 모두 거짓일 때만 거짓이 된다.
not x	x가 참이면 거짓, x가 거짓이면 참이 된다.

논리 연산자

```
a = 15
print( (a >= 10) and (a<=19) )

print( (a <= 10) or (a<=19) )

print(not(a>=10))
```

```
True
True
False
```

부울 (bool) 자료형

- True 와 False 값을 가지는 자료형
- 부울형이 아니 자료형의 데이터도 부울형으로 변환 가능
- 0이거나 비어있는 것은 False
- 0이 아닌 값이나 무엇인가 들어 있는 것은 True

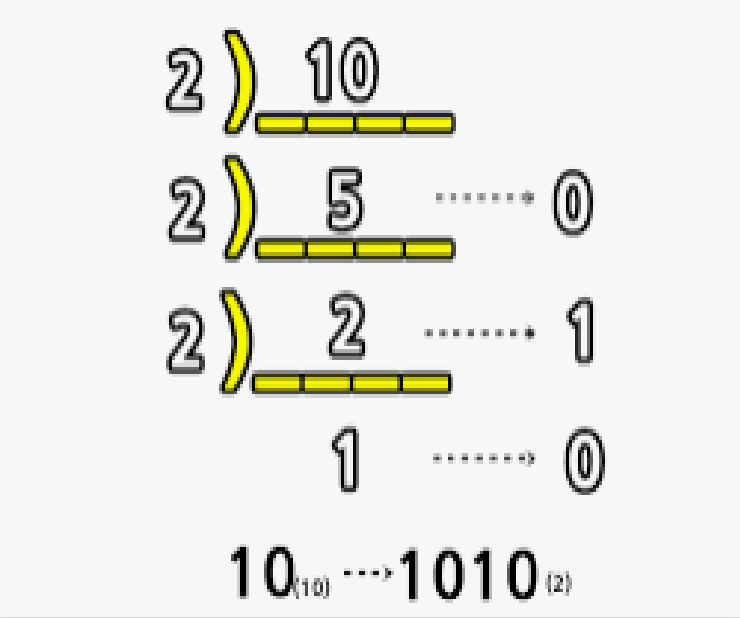
```
print(bool(0))                                  False
print(bool(99))                                 True
print(bool(-12.3))                              True
print(bool(''))                                 False
print(bool('abc'))                              True
```

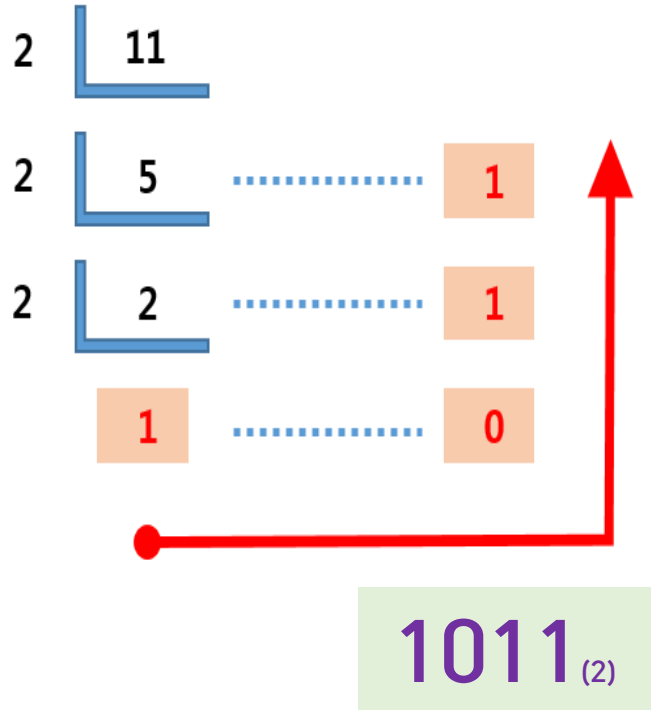
비트 연산자

- 비트bit 단위로 처리가 이루어지는 연산
- 어떤 정수의 이진수 값을 확인하고 싶으면 bin() 함수를 사용

연산자	의미	설명
&	비트 단위 AND	두 개의 피연산자의 해당 비트가 모두 1이면 1, 아니면 0
|	비트 단위 OR	두 피연산자의 해당 비트 중 하나라도 1이면 1, 아니면 0
^	비트 단위 XOR	두 개의 피연산자의 해당 비트의 값이 같으면 0, 아니면 1
~	비트 단위 NOT	0은 1로 만들고, 1은 0으로 만든다.
<<	비트 단위 왼쪽으로 이동	지정된 개수만큼 모든 비트를 왼쪽으로 이동시킨다.
>>	비트 단위 오른쪽으로 이동	지정된 개수만큼 모든 비트를 오른쪽을 이동시킨다.

비트 연산자





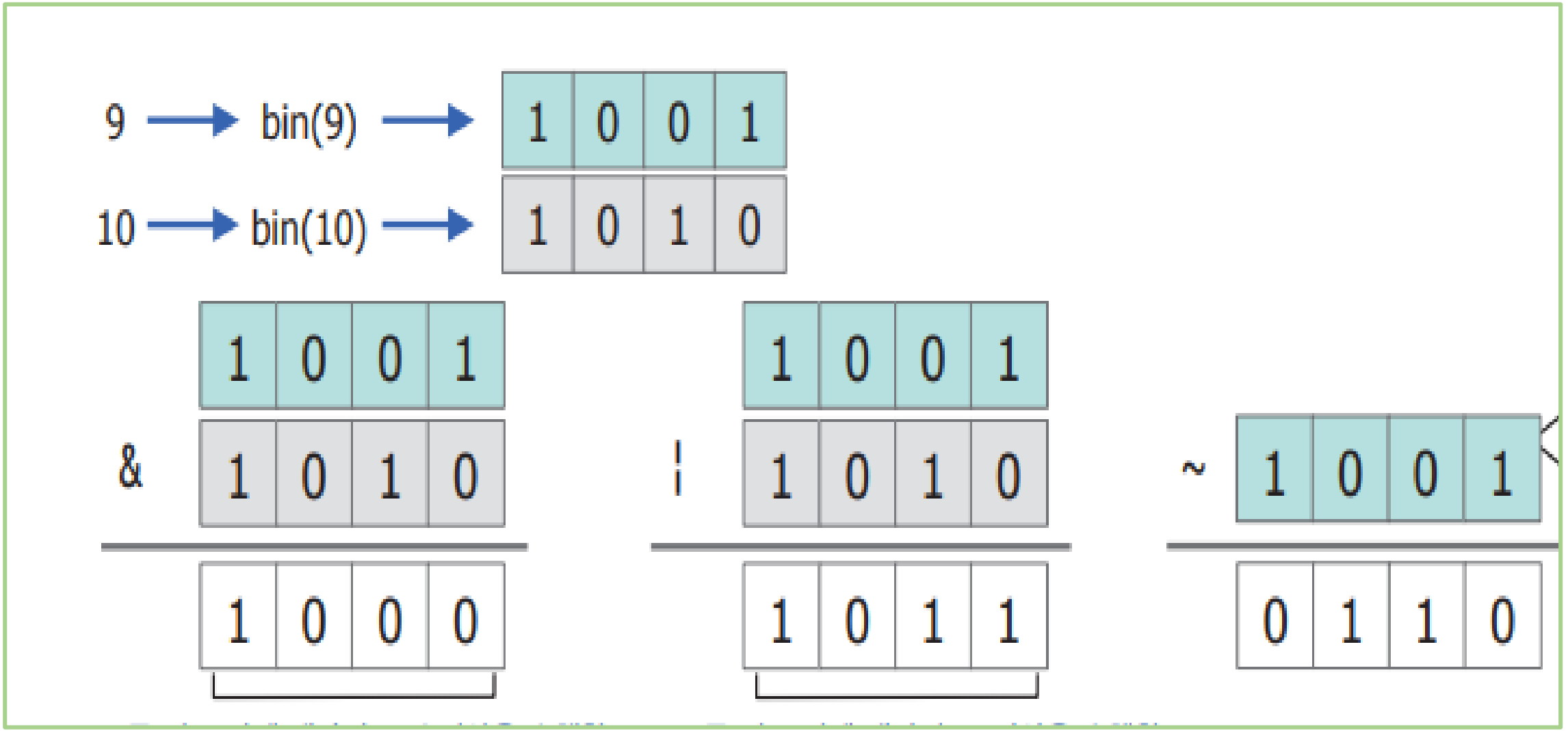
비트 연산자

2347			
2	3	4	7
1000	100	10	1
10^3	10^2	10^1	10^0

1	0	1	0	1	0
x	x	x	x	x	x
2^5	2^4	2^3	2^2	2^1	2^0
32	0	8	0	2	0

2347=2*1000 + 3*100+4*100+7*1

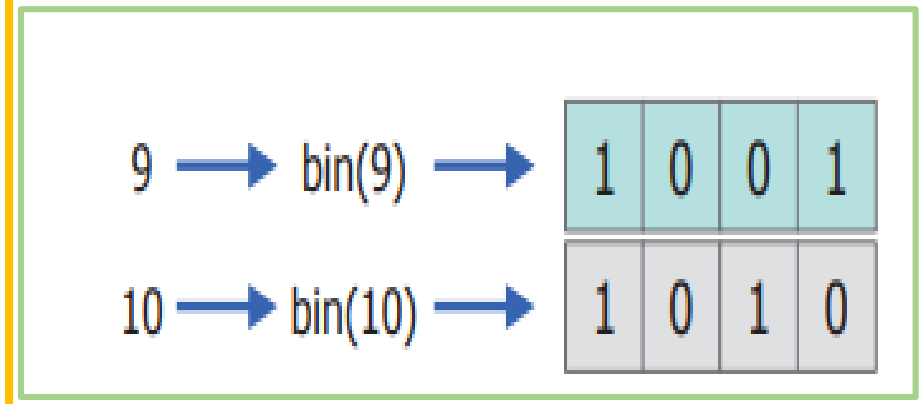
비트 연산자


9 → bin(9) → 1 0 0 1
10 → bin(10) → 1 0 1 0
1 0 0 1
& 1 0 1 0
1 0 0 0
1 0 0 1
| 1 0 1 0
1 0 1 1
~ 1 0 0 1
0 1 1 0

비트 연산자

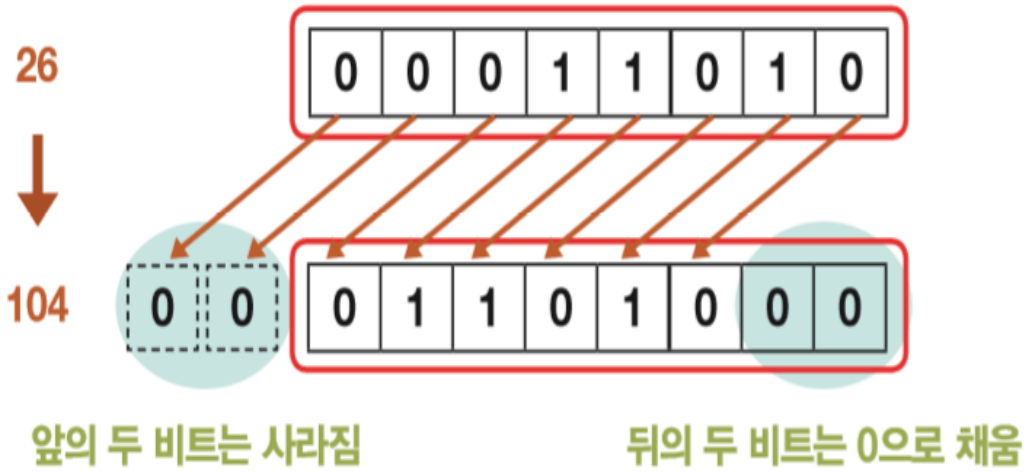
```
print(bin(9))
print(bin(10))
print(bin(9 & 10))
print(bin(9 | 10))
print(bin(9 ^ 10))
print(9 ^ 10)
```

```
0b1001
0b1010
0b1000
0b1011
0b11
3
```

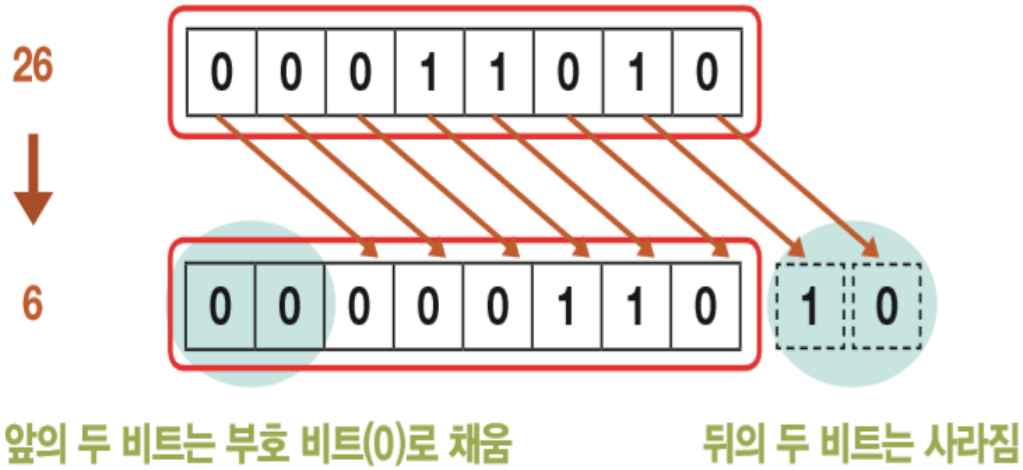


쉬프트shift 연산

- << 연산자는 지정된 수만큼 모든 비트를 왼쪽을 이동시키는 연산자
- >> 연산자는 지정된 수만큼 모든 비트를 오른쪽으로 이동시키는 연산자



26<<2



26>>2

쉬프트shift 연산

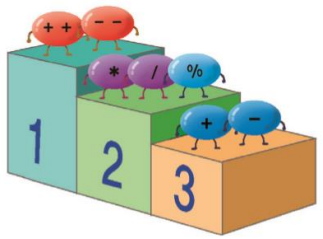

```
print(4<<1)
print(4<<2)
print(2<<4)
print(16>>2)
print(9>>2)
```

```
8
16
32
4
2
```

	129	64	32	16	8	4	2	1
4	0	0	0	0	0	1	0	0
2	0	0	0	0	0	0	1	0
16	0	0	0	1	0	0	0	0
9	0	0	0	0	1	0	0	1

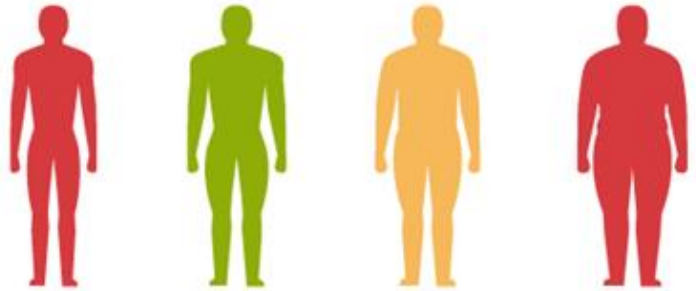
연산자 우선순위

연산자	설명
()	괄호 연산자로 가장 높은 우선 순위 연산자
**	지수 연산자
~ , + , -	단항 연산자(예: -10, +20)
* , / , % , //	곱셈, 나눗셈, 나머지 연산자
+ , -	덧셈, 뺄셈
>> , <<	비트 이동 연산자
&	비트 AND 연산자
^ , |	비트 XOR 연산자, 비트 OR 연산자
<= , < , > , >=	비교 연산자
== , !=	동등 연산자
= , %= , /= , //= , -= , += , *= , **=	할당 연산자, 복합 할당 연산자
is , is not	아이덴티티 연산자
in , not in	소속 연산자
not , or , and	논리 연산자



도전

사용자로부터 신장과 체중을 입력받아 BMI 값을 출력하는 프로그램을 작성하여 보자.



$$BMI = \frac{(\text{weight in kilograms})}{\text{height in meters}^2}$$

BMI Chart

BMI less than 18.50	Underweight
BMI 18.50 - 24.99	Healthy weight
BMI 25.00 - 29.99	Overweight
BMI 30 or more	Obese

몸무게를 kg 단위로 입력: 85.0
키를 미터 단위로 입력: 1.83
당신의 BMI= 25.381468541909282

도전

```
weight = float(input("몸무게를 kg 단위로 입력: "))
height = float(input("키를 미터 단위로 입력: "))

bmi = (weight / (height**2))

print("당신의 BMI=", bmi)
```

$$BMI = \frac{(\text{weight in kilograms})}{\text{height in meters}^2}$$

```
몸무게를 kg 단위로 입력: 85.0
키를 미터 단위로 입력: 1.83
당신의 BMI= 25.381468541909282
```

round() 함수

```
no1 = 1.457
no2 = 1.557

print("%s : 반올림 후 데이터 값  %s" % (no1, round(no1)))
print("%s : 반올림 후 데이터 값 %s" %  (no2, round(no2)))

print("%s : 반올림 후 데이터 값 %s" % (no1, round(no1,1)))
print("%s : 반올림 후 데이터 값 %s" % (no2, round(no2,2)))
```

round (number , ndigits)

```
1.457 : 반올림 후 데이터 값  1
1.557 : 반올림 후 데이터 값 2
1.457 : 반올림 후 데이터 값 1.5
1.557 : 반올림 후 데이터 값 1.56
```

리스트[]

◆ 여러 개의 자료를 모아서 하나의 묶음으로 저장

◆하나의 변수에 여러 개의 값을 저장

◆ 리스트 내의 개별 데이터 - 항목, 요소

```
city_list = [ '광주',  ' 서울',  ' 수원', 제주' ]
```

index

city_list[0]	city_list[1]	city_list[2]	city_list[3]
광주	서울	수원	제주

range()

- range(start, stop, step)
- start에서 시작하여 (stop-1)까지 step 간격으로 정수들을 생성
- stop 값은 반드시 지정 (start, step은 생략 가능)

```
print(list(range(0,5,1)))
print(list(range(0,5)))
print(list(range(5)))
print(list(range(1,11,2)))
```

random 모듈

◆ 난수와 관련한 함수를 제공

```
import random

print(random.random())
print(random.randint(1,7))

print(random.randrange(7))
print(random.randrange(1,7))
print(random.randrange(0,10,2))
```

```
# 0 이상 1 미만의 임의의 실수를 반환

# 이 함수는 매번 다른 실수를 반환

# 1 이상 7 이하(7을 포함)의 임의의 정수를 반환

# 0 이상 7 미만(7을 포함하지 않음)의 임의의 정수를 반환

# 1 이상 7 미만(7을 포함안함)의 임의의 정수를 반환

# 0, 2, 4, 8 중(10은 포함 안함) 하나를 반환함
```