

# 파이썬을 계산기 처럼 이용

# 간단한 사칙연산

## 사칙연산

- 파이썬에서 사칙 연산을 하려면 더하기(+), 빼기(-), 곱하기(\*), 나누기(/)

기호를 이용

```
[ ] 1 + 1
```

```
[ ] 5 - 2
```

- 파이썬에서는 숫자와 연산자 사이의 공백은 무시하므로 이 공백은 있어

도 되고 없어도 됨

```
[ ] 15 * 2
```

```
[ ] 10 / 2
```

- 파이썬에서 나눗셈 연산은 실수로 처리하기 때문

# 간단한 사칙연산

---

## 사칙연산

- 정수와 실수
- 정수 : 자연수( $1, 2, 3 \dots$ )와 자연수의 음수, 그리고 0을 이뤄진 수의 집합
- 실수 : 유리수와 무리수의 이뤄진 집합
- 프로그램에서는 숫자에 소수점이 없으면 정수(int)이고 소수점이 있으며 실수(float)

# 간단한 사칙연산

---

## 사칙연산

- 정수 연수만 할 수 있는 것이 아니라 실수 연산도 할 수 있음

```
[ ] 1.2 + 5.3
```

```
[ ] 3.5 - 5.0
```

```
[ ] 1.4 * 2
```

```
[ ] 5 / 2
```

# 간단한 사칙연산

## 사칙연산

- 연산 기호가 두 개 이상일 경우 일반적인 연산 규칙

'괄호 안 계산 -> 지수 계산 -> 곱셈과 나눗셈 계산 -> 덧셈과 뺄셈 계산' 순서로 연산

- 같은 순위의 연산일 경우 왼쪽에서 오른쪽 순서로 계산, 중복된 괄호가 있을 경우 안쪽 괄호부터 계산을 따름
- 연산에서 괄호는 '('와 ')'를 이용
- 괄호는 사용할 때는 왼쪽 괄호와 오른쪽 괄호 쌍이 꼭 맞아야 함

# 간단한 사칙연산

---

## 사칙연산

### ◦ 복합연산

```
[ ] 2 + 3 * 4
```

```
[ ] 3 / 2 * 4 - 5 / 2
```

```
[ ] 10 / 5 + (5 - 2) * 2
```

```
[ ] (5 * 4 - 15) + ((5 - 2) * (9 - 7))
```

# 간단한 사칙연산

---

## type() 함수

- 자료의 형식(데이터 타입)을 알려주는 type()이라는 함수
- type() 함수의 인자로 정수를 입력하면 int를, 실수를 입력하면 float를 결과로 돌려줌
- type() 함수에 정수와 실수를 입력

```
[ ] type(3)
```

```
[ ] type(1.2)
```

# 거듭 제곱과 나머지

## 거듭 제곱

- 숫자A를 n번 곱하는 거듭제곱(Power)

```
[ ] 2 * 2 * 2 * 2 * 2
```

- 파이썬에서는 거듭제곱을 위한 연산자 \*\*

(두 별표 사이에는 공백이 없어야 함)

```
[ ] 2 ** 5
```

- 정수뿐만 아니라 실수도 거듭제곱

```
[ ] 1.5 ** 2
```



# 거듭 제곱과 나머지

---

## 거듭 제곱

- 거듭제곱의 지수도 정수일 필요가 없음
- 파이썬에서 제공하는 수학(math) 라이브러리를 이용하면 거듭제곱, 거듭제곱근 외에 다양한 수학 함수를 입력함

```
[ ] 2 ** (1/2)
```

# 거듭 제곱과 나머지

## 나머지

- 나머지를 구하기 위해 파이썬에서는 퍼센트 기호(%)를 이용
- 퍼센트 기호(%)를 나머지 연산자(Modulo operator) 라고 함

```
[ ] 13 % 5
```

- 몫은 정수 나누기 연산자(//)로 구할 수 있음

```
[ ] 13 // 5
```

# 산술 연산자

## 산술 연산자

### ○ 산술 연산자

연산자 기호	의미	예	결과
+	더하기	5+2	7
-	빼기	5-2	3
*	곱하기	5*2	10
/	나누기	5/2	2.5
**	거듭제곱	5**2	25
%	나머지	5%2	1
//	몫	5//2	2

# 과학적 표기법

## 과학적 표기법

- 연산할 때 아주 큰 수나 작은 수를 다뤄야 할 때 특히 과학이나 공학 분야

```
[ ] 3 * 10 ** 8
```

- 거듭제곱 연산자로 큰 수를 입력할 수 있지만 10의 거듭제곱(즉,  $10^n$ )의 경우  $e$ n으로 편리하게 입력, 여기서  $n$ 은 정수로 양수, 음수, 0
- 10의 거듭제곱 표시를 위한  $e$ n 앞에는 항상 숫자가 있어야 함

```
[ ] 3e8
```

# 과학적 표기법

---

## 과학적 표기법

- 파이썬에서 과학적 표기법으로 출력할 때는 e 다음에는 지수가 양수냐 음수냐에 따라서 양수 기호(+)나 음수 기호(-)가 들어감

```
[ ] 1e15
```

```
[ ] 1e16
```

```
[ ] 1e-4
```

```
[ ] 1e-5
```

# 진수 표현과 변환

## 과학적 표기법

- 파이썬은 10진법 외에 2진법, 8진법, 16진법으로 숫자를 입출력하며 변환하는 방법을 제공
- 10진수 외에 2진수, 8진수, 16진 수를 입력하기 위해서는 숫자 앞에 각각 '0b', '0o', '0x'를 붙임

```
[ ] 17
```

```
[ ] 0b10001
```

```
[ ] 0o21
```

```
[ ] 0x11
```

# 진수 표현과 변환

## 과학적 표기법

- 파이썬에서는 2진수, 8진수, 16진수 형태로 숫자를 입력해도 기본적으로 10진수로 출력
- 파이썬에서 10진수를 2진수, 8진수, 16진수로 변환하는 함수가 있고 각각의 함수는 `bin()`, `oct()`, `hex()`

```
[ ] bin(17)
```

```
[ ] oct(17)
```

```
[ ] hex(17)
```

# 진수 표현과 변환

---

## 과학적 표기법

- 출력 결과는 숫자가 아니라 문자 열이라는 것임
- 출력 결과는 작은 따옴표(' ') 안에 있음
- 출력 결과가 숫자가 아니라 문자열임을 나타냄
- `bin()`, `oct()`, `hex()` 함수를 이용한 출력 결과는 산술 연산에 이용할 수 없음,
- 진수 변환은 연산이 모두 끝난 후에 해야 함



# 진수 표현과 변환

## 과학적 표기법

- 서로 다른 진수의 연산 결과를 10진수, 2진수, 8진수, 16진수로 변환해 출력

```
[ ] 0b10 * 0o10 + 0x10 - 10
```

```
[ ] bin(0b10 * 0o10 + 0x10 - 10)
```

```
[ ] oct(0b10 * 0o10 + 0x10 - 10)
```

```
[ ] hex(0b10 * 0o10 + 0x10 - 10)
```

# 논리 연산 및 비교 연산

---

## 논리 연산

- 어떤 조건을 만족하는 참(True)과 만족하지 않는 거짓(False)을 이용해 연산하는 논리 연산(logical operation), 논리 연산은 불린 연산(Boolean operation)이라고도 함
- 파이썬에서 논리 연산을 위한 데이터 타입은 불(bool)임
- 불 데이터 타입에는 논리 참(True) 혹은 논리 거짓(False)이 있음
- 참(True) 혹은 거짓(False)을 입력할 때 참은 True, 거짓은 False를 입력

# 논리 연산 및 비교 연산

---

## 논리 연산

- 'true'나 'False'처럼 따옴표를 사용하면 안 됨,
- 따옴표를 이용해 입력하면 불 데이터가 아니라 문자열 데이터로 인식함
- True를 true나 TRUE로 쓰거나 False를 false나 FALSE로 쓸 수 없음

```
[ ] print(True)
```

```
[ ] print(False)
```

# 논리 연산 및 비교 연산

---

## 논리 연산

- 문자열을 출력하는 `print('True')`나 `print('False')`와 달리, 논리 연산의 결과를 출력한 것
- 불 데이터인 `True`나 `False`의 데이터 타입은 `type()` 함수를 이용해 확인

```
[ ] type(True)
```

# 논리 연산 및 비교 연산

---

## 논리 연산

- 불 데이터의 경우 논리 연산만 할 수 있음
- 논리 연산에는 논리곱(and), 논리합(or), 논리 부정 (not) 등
- 논리곱(and)은 두 개의 불 데이터가 모두 참일 때만 참이고 나머지는 거짓
- 논리합(or)은 두 개의 불 데이터 중 하나라도 참이면 참이고 둘 다 거짓이면 거짓
- 논리부정(not)은 하나의 불 데이터가 참이면 거짓이고 거짓이면 참

# 논리 연산 및 비교 연산

## 논리 연산

### ○ 논리 연산자

논리연산자	의미	활용 예	설명
and	논리곱	A and B	A와 B 모두 참일 때만 참이고 나머지는 거짓
or	논리합	A or B	A와 B 중 하나라도 참이며 둘 다 거짓일 때 거짓
not	논리부정	Not A	A가 참이며 거짓이고, 거짓이면 참

# 논리 연산 및 비교 연산

## 논리 연산

- 논리 연산자를 이용한 진리표

A	B	A and B	A or B	not A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

```
[>] print(True and False)
      print(True or False)
      print(not True)
```

# 논리 연산 및 비교 연산

## 비교연산자

- 두 개의 숫자를 비교하는 비교 연산

비교 연산자	의미	활용 예	설명
==	같다	$a == b$	a는 b와 같다
!=	같지 않다	$a != b$	a는 b와 같지 않다
<	작다	$a < b$	a는 b보다 작다
>	크다	$a > b$	a는 b보다 크다
<=	작거나 같다	$a \leq b$	a는 b보다 작거나 같다
>=	크거나 같다	$a \geq b$	a는 b보다 크거나 같다



# 논리 연산 및 비교 연산

---

## 비교연산자

- 비교 연산자를 이용해 연산
- 비교 연산의 결과는 불 데이터 형식으로 나옴

```
[ ] print(5 == 3)
    print(5 != 3)
    print(5 < 3)
    print(5 > 3)
    print(5 <= 3)
    print(5 >= 3)
```

# 논리 연산 및 비교 연산

## 논리 연산과 비교연산자 혼합연산

- 비교 연산자의 우선순위가 논리 연산자의 우선순위보다 높음
- 비교 연산자와 논리 연산자가 함께 있으면 비교 연산을 먼저 함

```
[ ] print( 1 > 0 and -2 < 0)
```

- 비교 연산인 '1>0 과 '-2<0'을 먼저 수행하는데 결과는 모두 True
- 그 후 이 연산의 결과를 이용해 논리곱{and) 연산을 수행하는데 결과는 True

# 논리 연산 및 비교 연산

## 논리 연산과 비교연산자 혼합연산

- 괄호와 연산이 함께 있으면 괄호의 우선순위가 높음
- 괄호가 여러 겹 있을 때는 가장 안쪽 괄호부터 먼저 계산함

```
[ ] print((3 < 0) and ((-5 > 0) and (1 > 5)))
```

```
[ ] print((3 > 0) or ((-5 > 0) and (1 > 5)))
```

```
[ ] print(((3 > 0) or (-5 > 0)) and ((4 > 8) or (3 < 0)))
```

# 정리

---

## 정리

- 파이썬을 이용해 계산기처럼 연산하는 방법
- 간단한 사칙연산부터 거듭제곱과 과학적 표기법
- 10진수, 2진수, 8진수, 16진수의 표현과 변환 방법
- 논리 연산 및 비교 연산