파이썬과 함께하는 이세계 코딩 생활

2주차 : 변수부터 산술 연산까지~

오늘 배울 내용

- 변수
- 연산자
- 입출력
- 산술 연산

변수 (variable)

- 변수란 무엇인가?
  - 데이터를 저장하는 **메모리 공간**의 이름
  - 변수에 데이터를 저장하면 (== 변수명을 붙여 놓으면) 나중에 그 데이터에 다시 접근할 수 있다.
  - 수학에서 나오는 변수의 개념과 유사하다.
    - y = x+3 라는 식이 있을 때, x, y 는 변하는 값을 가진다.
    - 마찬가지로 파이썬에서의 변수도 변하는 값을 가진다.

표준 입출력 함수, 주석 처리

- 입력 : input()
  - input() 은 문자열을 입력 받는다.
  - 숫자를 받더라도 문자열 취급을 하니까,
    사용자로부터 직접 입력을 받은 후에 사용하려면 int() 를 취해줘야 한다.

- 출력 : print()
  - print() 는 출력하는 함수이다.
  - 출력하고 기본적으로 한 줄 바꾼다. end = ‘ ‘ 라고 쓰면 공백으로 대체 가능

- 주석 : #
  - 주석 처리한다.
  - 인터프리터가 해석하지 않는다.

표준 입출력 함수, 주석 처리

코드

```
a = input() # a 입력
print(a) # a 출력
type(a) # a의 자료형 출력
```

예시1)

```
>> 100
100
str
```

코드

```
a = int(input()) # a 입력
print(a, end= ' ') # a 출력
type(a) # a의 자료형 출력
```

예시2)

```
>> 100
100 int
```

자료형(data type)
: 문자형(str), 숫자형(int, float) 등이 있다! 자세한 건 다음주에 계속!

변수의 사용

처음 사용

```
s = "포자춤~ 포자춤~ 기쁨의~ 포자춤~"
print(s)
```

```
포자춤~ 포자춤~ 기쁨의~ 포자춤~
```

재사용

```
s = "포자춤~ 포자춤~ 기쁨의~ 포자춤~"
s = "보라버섯~ 보라버섯~ 기쁨의~ 보라버섯~"
print(s)
```

```
보라버섯~ 보라버섯~ 기쁨의~ 보라버섯~
```

변수를 재사용하면, 기존에 저장한 값은 없어지고 새로운 값이 저장된다.

변수를 왜 써야하는가!

- 이 문장들을 출력하려고 할 때

```
포자춤~ 포자춤~ 기쁨의~ 포자춤~
포자춤~ 포자춤~ 기쁨의~ 포자춤~
보라버섯~ 보라버섯~ 기쁨의~ 보라버섯~
보라버섯~ 보라버섯~ 기쁨의~ 보라버섯~
```

그냥 출력

```
print("포자춤~ 포자춤~ 기쁨의~ 포자춤~")
print("포자춤~ 포자춤~ 기쁨의~ 포자춤~")
print("보라버섯~ 보라버섯~ 기쁨의~ 보라버섯~")
print("보라버섯~ 보라버섯~ 기쁨의~ 보라버섯~")
```

변수 사용해서 출력

```
s1 = "포자춤~ 포자춤~ 기쁨의~ 포자춤~"
s2 = "보라버섯~ 보라버섯~ 기쁨의~ 보라버섯~"

print(s1)
print(s1)
print(s2)
print(s2)
```

지금은 간단한 텍스트라 그냥 출력하면 되지만, 중요한 값이거나 데이터의 양이 엄청 많다면....? 🤔

변수명 규칙

변수명 ⊂ 식별자

1. 변수명은 영어 대소문자, 숫자, 언더바(_)로만 작성할 수 있다.
2. 변수명은 숫자로 시작할 수 없습니다. 즉, 반드시 영문자나 언더바(_)로 시작해야 한다.
3. 변수명은 대소문자를 구분한다.
4. 변수명에는 파이썬에서 미리 정의된 예약어(reserved words)는 사용할 수 없다.
   (ex. for, if, while, True, False)

특수문자(&, *, (,), %, $, #, @, , !), 공백, 한글 사용 금지

- num (O)
- 1class (X)
- _2team (O)
- For (O)

파이썬 예약어

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

기본 연산자

- 산술 연산자
- 비교 연산자
- 할당 연산자
- 논리 연산자
- 연산자 우선순위

- 비트 연산자
- 멤버 연산자
- 식별 연산자

산술 연산자

연산자	설명
+	더하기
-	빼기
*	곱하기
/	나누기
%	나머지
**	제곱
//	몫

코드

```
print(1+2)
print(1-2)
print(1*2)
print(2/1)
print(5%2)
print(3**2)
print(5//2)
```

결과

```
3
-1
2
2.0
1
9
2
```

← 실수형으로 출력

int()

- 숫자나 문자열을 정수로 바꿔주는 함수

코드

```
print(int(2/1))
```

결과

```
2
```

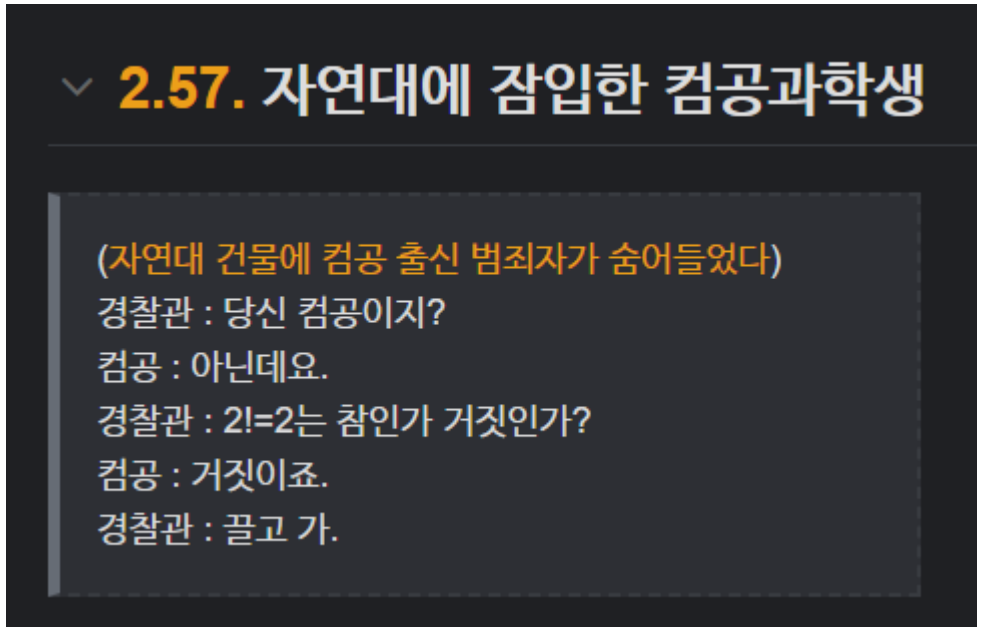
```
help(int)
```

```
Help on class int in module builtins:

class int(object)
 |  int([x]) -> integer
 |  int(x, base=10) -> integer
 |
 |  Convert a number or string to an integer, or return 0 if no arguments
 |  are given.  If x is a number, return x.__int__().  For floating point
 |  numbers, this truncates towards zero.
 |
 |  If x is not a number or if base is given, then x must be a string,
 |  bytes, or bytearray instance representing an integer literal in the
 |  given base.  The literal can be preceded by '+' or '-' and be surrounded
 |  by whitespace.  The base defaults to 10.  Valid bases are 0 and 2-36.
 |  Base 0 means to interpret the base from the string as an integer literal.
 |  >>> int('0b100', base=0)
 |  4
  …
```

비교 연산자

연산자	수학에서는 이렇게
==	=
!=	≠
>	>
<	<
>=	≥
<=	≤



수학에서 2!=2는 2!와 2가 서로 같다는 뜻이니까 참이지만,
코딩할때 2!=2는 2와 2가 서로 다르다는 뜻이므로 거짓임

연산자 = 는 같다는 뜻이 아니라, 할당 연산자이다!

할당 연산자 (== 대입 연산자)

연산자	변수 a 와 b로 예시	풀어서 설명
=	a = b	b를 a에 할당
+=	a += b	a = a+b
-=	a -= b	a = a-b
*=	a *= b	a = a*b
/=	a /= b	a = a/b
%=	a %= b	a = a%b
**=	a **= b	a = a**b

계산한 결과를 a에 할당한다고 생각하면 마음 편함

논리 연산자

- 논리 연산이란?
  - 참, 거짓 이렇게 2가지 값 만을 가지고 행하는 연산!

연산자	뭐라고 부르는지	예시	설명
and	논리곱	a and b	둘 다 True 면 True
or	논리합	a or b	둘 중 하나만 True 면 True
not	부정	not a	아니면 True

논리 연산자

```
True and True

True and False

False and False
```

```
True or False

False or False

True or True
```

```
not True

not False
```

논리 연산자

```
True and True
>> True
True and False
>> False
False and False
>> False
```

```
True or False
>> True
False or False
>> False
True or True
>> True
```

```
not True
>> False
not False
>> True
```

논리 연산자 + 비교 연산자

```
a = 1
b = 2

a == 1 and a < b

>> True
```

a 가 1 이고, a 가 b보다 작다.

```
a = 1
b = 2

a < b or b > 8

>> True
```

a 가 b보다 작거나, b가 8보다 크다.

오늘 배운 연산자들 우선순위

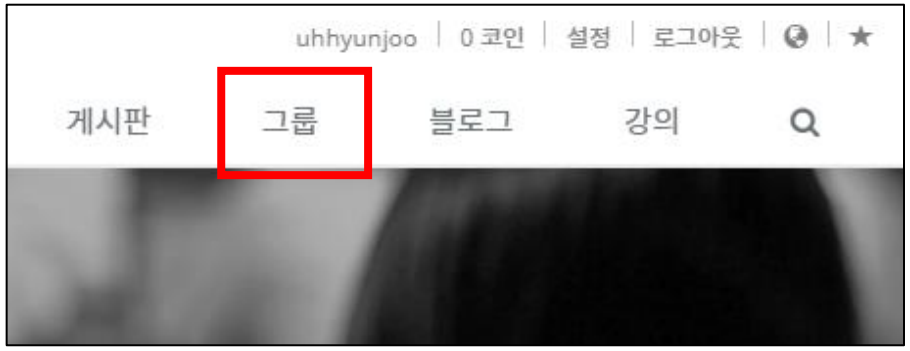
연산자	설명	비고
**	지수	
+ -	단항 플러스, 단항 마이너스	-2, +a, -b
* / % //	곱하기, 나누기, 나머지, 몫	
+ -	덧셈과 뺄셈	
<= < > >=	비교 연산자	
<> == !=	평등 연산자	
= %= /= //= -= += *= **=	할당 연산자	
not or and	논리 연산자	

한 줄에 숫자 여러 개 입력 받기

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

백준으로 파이썬 맛보기

- 2주차 문제집



문제 번호	제목	정보	맞은 사람	제출	정답 비율
2557	Hello World	성공 분류	126977	403972	42.167%
1000	A+B	성공 다국어 디버그 분류	117259	378021	43.786%