

1장. 파이썬 설치 & 변수 & 연산자



프로그래밍이란?

- **프로그래밍(Programming)이란?**

- 컴퓨터 프로그램을 만드는 일
- 컴퓨터에게 원하는 작업을 수행하도록 명령을 내리는 과정

- **프로그램(Program)**

- 컴퓨터에게 일을 시키는 명령의 집합 또는 프로그래밍한 작업의 결과.

- **프로그래밍 언어의 종류**

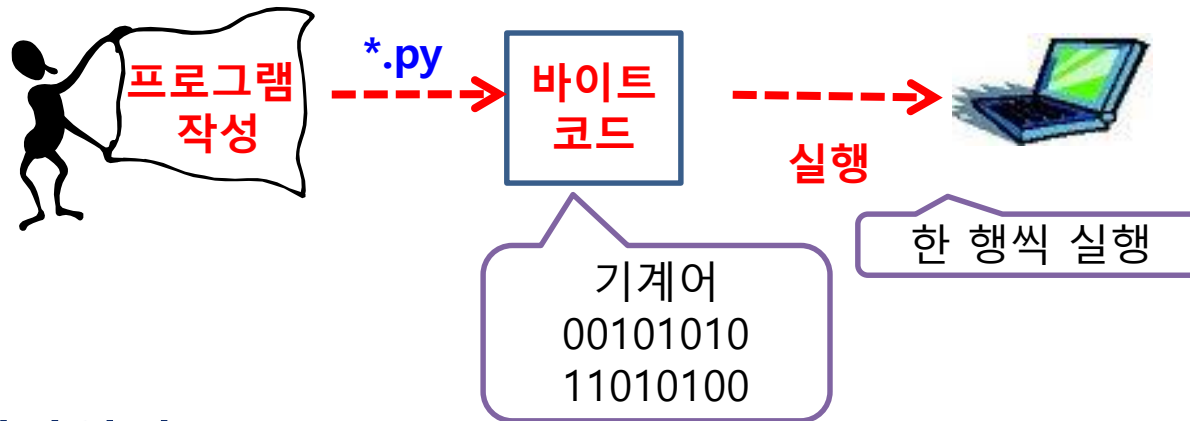
- C언어 , C++언어, Java, Python, JavaScript, C#

- **인터프리터, 컴파일러**

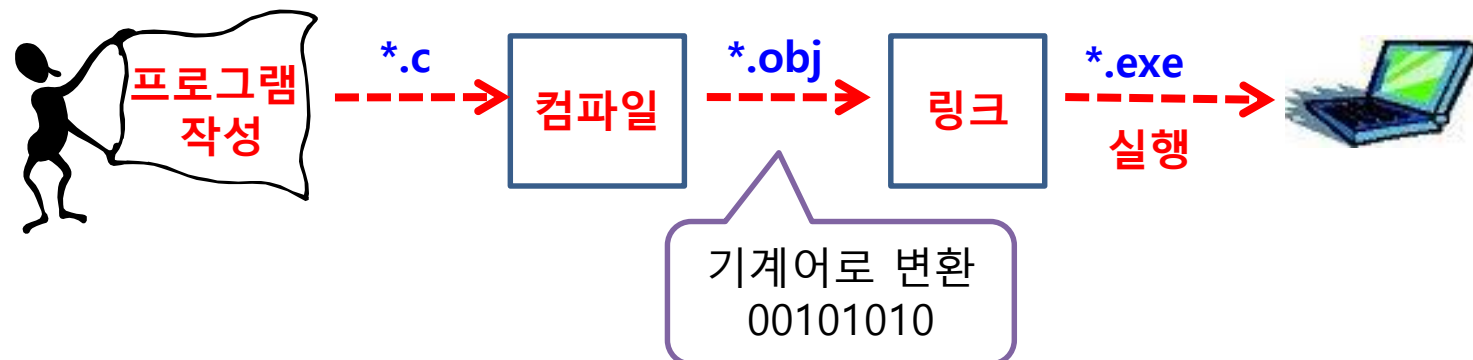
- 프로그램 언어를 컴퓨터가 알 수 있는 언어(기계어)로 바꿔 주는 프로그램
- **인터프리터(코드를 한 줄씩 변환하여 실행) – 파이썬, 자바스크립트**
- **컴파일러(전체 코드를 한 번에 변환후 실행) – C, C++, Java**

인터프리터와 컴파일러

➤ 인터프리터(Interpreter)



➤ 컴파일러(Compiler)



Python 언어

◆ 파이썬(Python) 창시

- 창시자 : 1990년 네델란드 암스테르담의 **귀도 반 로섬**
(이름의 유래 - 좋아하는 코미디 프로그램)
- 플랫폼에 독립적이고, 인터프리터 언어이며 객체지향 언어이다.

◆ 파이썬의 특징

- ✓ 사람이 사고하는 체계와 비슷하다.
- ✓ 문법이 간결하고 읽기 쉽다.
- ✓ 오픈 소스로 무료 - 누구나 자유롭게 사용하고 확장 가능
- ✓ 개발 속도가 빠르다.
- ✓ 다양하고 많은 라이브러리를 사용할 수 있다.

Python 언어

◆ 파이썬으로 할 수 있는 일

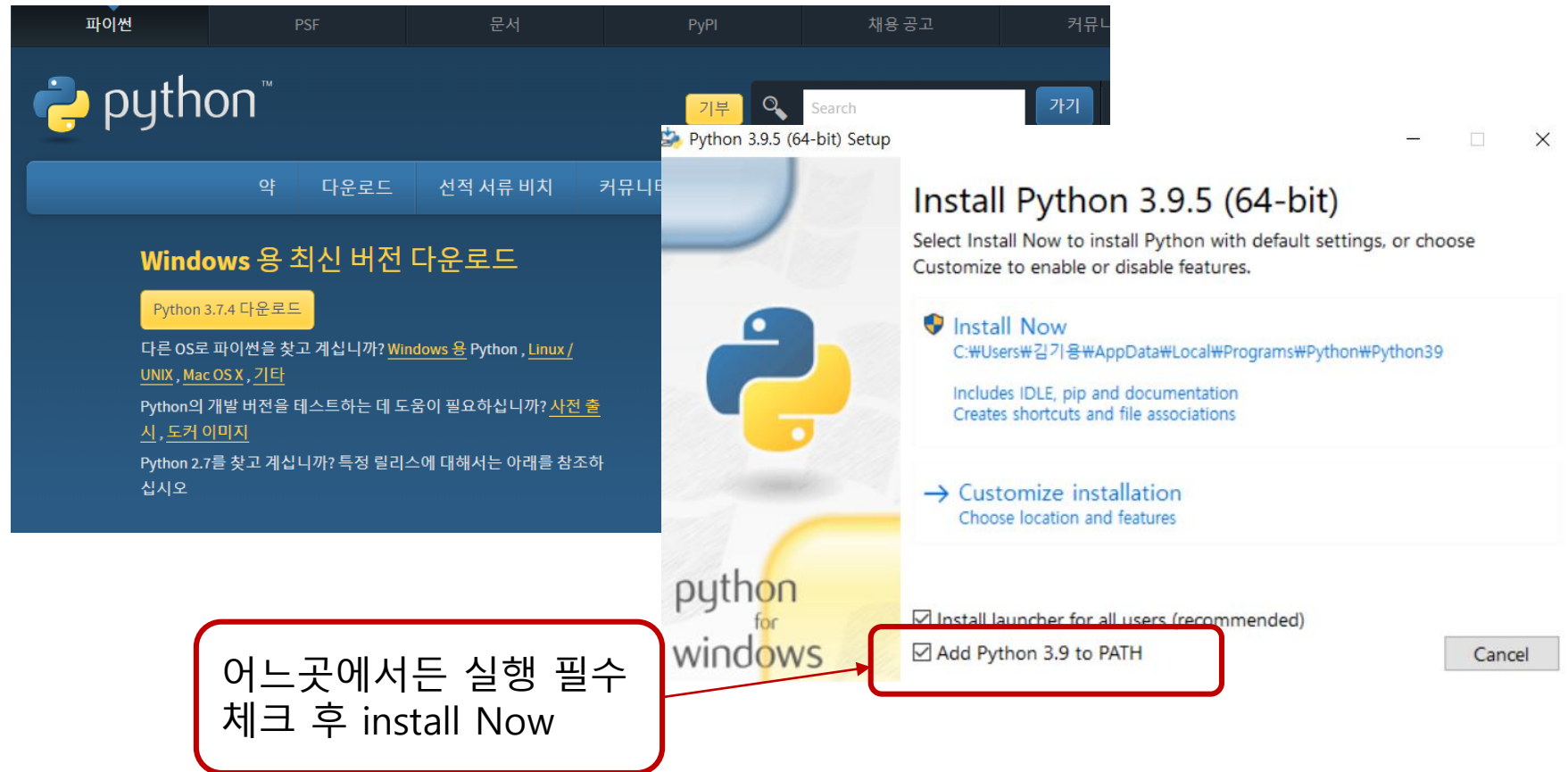
- **시스템 유틸리티 제작** – 윈도우 등에서 사용자에게 유용한 도구 (압축, 이미지뷰어 등)
- **웹 프로그래밍** – 웹 사이트 제작 등 웹 개발 (장고[Django] & 플라스크)
- **데이터 분석** – 데이터 수집, 분석 및 시각화(Pandas 모듈, matplotlib)
- **머신러닝(딥러닝)** – 인공지능 구현(텐서플로우, 케라스)
- **사물 인터넷** – IOT 구현(라즈베리파이[Raspberry Pi])

◆ 파이썬으로 할 수 없는 일

- **시스템 프로그래밍** – 운영체제 관련 제작(주로 C언어로 개발함)
- **모바일 프로그래밍** – 안드로이드, 아이폰 앱 등

파이썬 설치하기

- 파이썬 - www.python.org



파이썬 설치하기

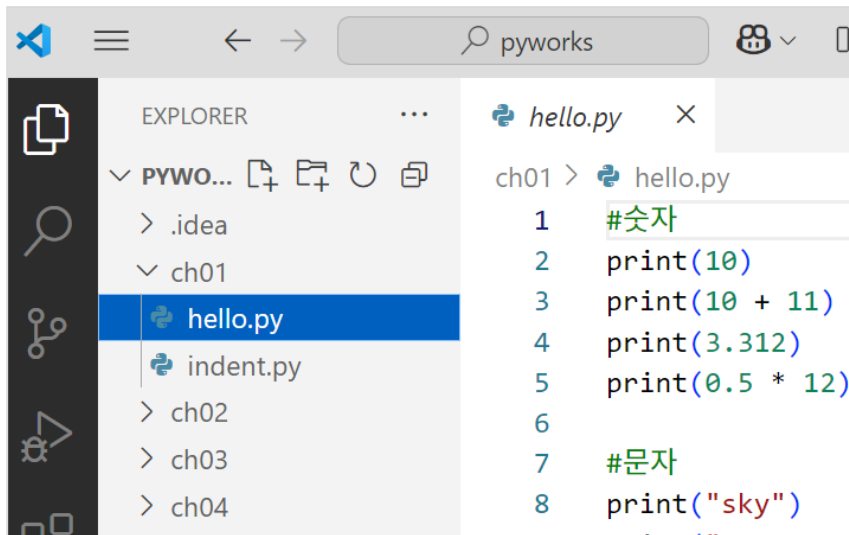
- 파이썬 설치 버전 확인 및 파이썬 실행하기

- 명령 프롬프트(cmd)

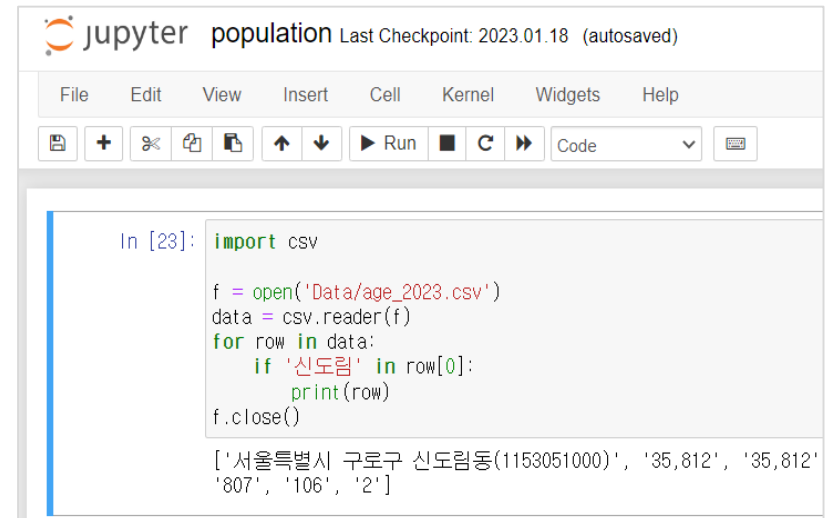
```
C:\Users\kiyon>python --version
Python 3.10.4

C:\Users\kiyon>python
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:13:41)
Type "help", "copyright", "credits" or "license" for more in
>>> 10 + 20
30
>>> "안녕하세요"
'안녕하세요'
>>> a = 10
>>> b = 20
>>> a + b
30
```

파이썬 추천 에디터(Editor)



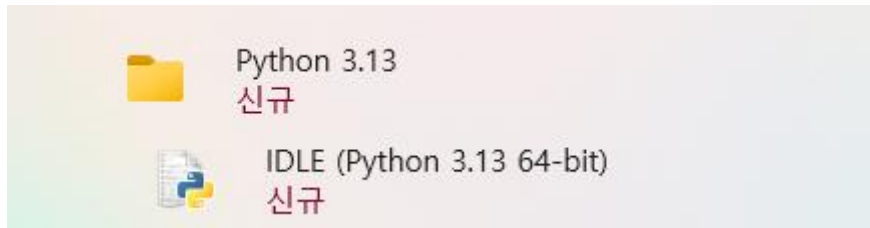
비주얼 스튜디오 코드(VS code)



주피터노트북(Jupyter Notebook)

파이썬 IDLE

■ 파이썬의 IDLE 실행



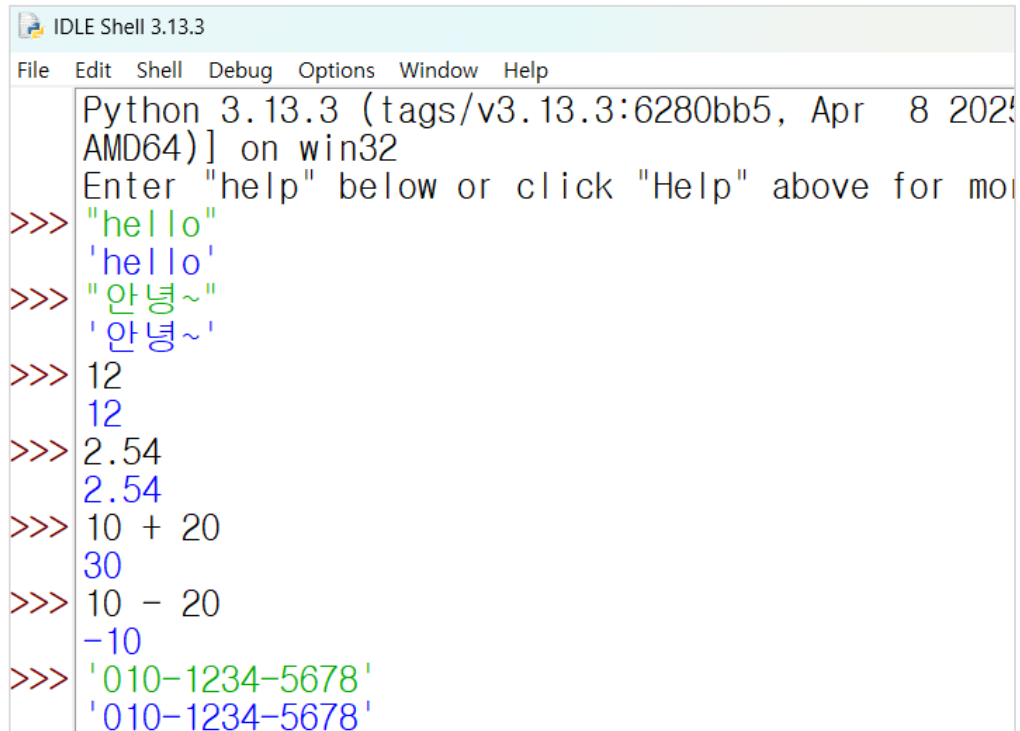
■ 파이썬의 셸

>>> 기호는 파이썬이 사용자가 입력하기를 기다리고 있다는 뜻이다.

사용자가 입력을 하면 바로 결과를 보여 주는데 이것을 대화형 셸(shell)이라고 함

파이썬 IDLE

■ 파이썬의 셸



```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 10:00:00) on win32
Enter "help" below or click "Help" above for more
>>> "hello"
'hello'
>>> "안녕~"
'안녕~'
>>> 12
12
>>> 2.54
2.54
>>> 10 + 20
30
>>> 10 - 20
-10
>>> '010-1234-5678'
'010-1234-5678'
```

기초 문법 - 오류

■ 구문 오류 및 변수 사용 오류

```
>>> 10 + 20
30
>>> 10 - 20
-10
>>> '010-1234-5678'
'010-1234-5678'
>>> 10 + 20 =
SyntaxError: cannot assign to expression
>>> hello
Traceback (most recent call last):
  File "<pyshell#8>", line 1, in <module>
    hello
NameError: name 'hello' is not defined. Did you mean: 'help'?
>>>
```

에러 : 구문 오류

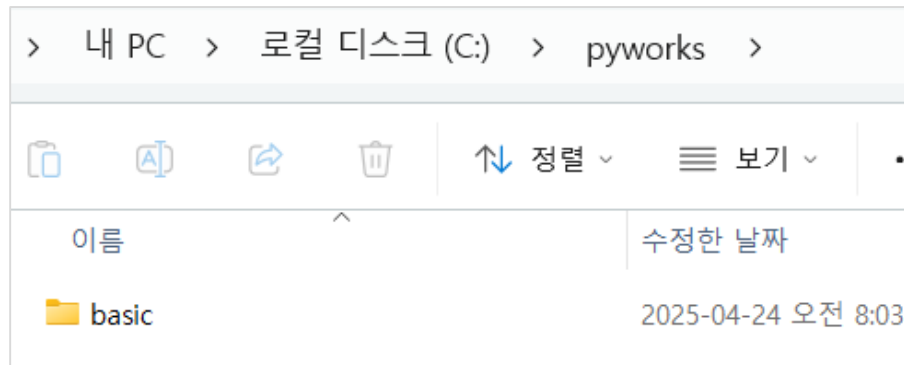
변수와 문자 구분

파이썬 IDLE - Editor

■ IDLE 에디터(Editor)

에디터는 프로그램을 작성하여 파일 형태로 저장하는 편집기이다.

1. File - > New File (새 파일)
2. 코드 작성
3. print.py로 저장 (pyworks 폴더 > basic 폴더 아래)
4. Run -> Run Module(F5) : 실행



파이썬 IDLE - Editor

■ 실습 예제

```
"""
주석(comment) - 코드 내용을 설명하기 위한 메모
- 한 줄 주석 : '#' 기호
- 여러 줄 주석 : 쌍따옴표 3번, 홑따옴표 3번 사용 가능
"""

# 출력 - print() 함수 사용
# 문자 출력 - '홑따옴표', '쌍따옴표' 모두 사용 가능
print('K')
print('hello')
print("아이스 아메리카노")
print('010-1234-5678')

# 숫자 출력 - 정수, 실수
print(12)
print(2.54)
print(0.756)
print("=====")

# 산술 계산
print(10 + 20)
print(50 - (5 * 6)) #20
print((50 - (5 * 6)) / 4) #5.0
```

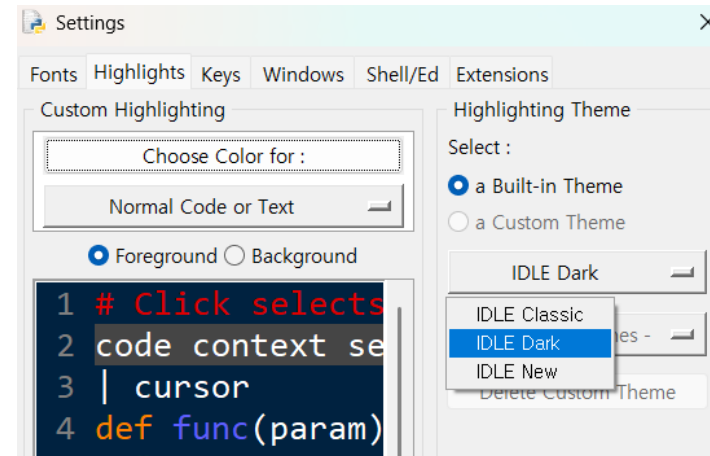
파이썬 IDLE – Editor

■ IDLE 에디터(Editor) 환경 설정

Options > Configure IDLE > Settings

- 글꼴 변경 : (Fonts) – consolas, size - 17
- 화면 색상 변경 : (Highlights) – IDLE Dark

```
# 출력 - print() 함수 사용
# 문자 출력 - '출따옴표',
print('K')
print('hello')
print("아이스 아메리카노")
print('010-1234-5678')
```



기초 문법

■ 기본 문법

- **자료형**을 사용하지 않는다. ($n = 10$, $\text{msg} = \text{'hello'}$)
- **주석(comment)** – 코드 내용을 설명하기 위한 메모
한 줄 주석 : '#' 기호
여러 줄 주석 : '''~''', (쌍따옴표 3번, 홑따옴표 3번 사용함)
- **들여쓰기(indent)**

4칸 들여쓰기

```
# 들여쓰기(indent)

n = 10
if n % 2 == 0:
    print("짝수")
else:
    print("홀수")
```

```
print('A')
print('B')
print('C')
```

1칸 들여쓰기로 예러

파이썬 파일 IDLE로 실행

◆ .py 파일 실행하기

1. 파일 > 단축 메뉴> Edit with IDLE > Edit with IDLE 3.13



2. IDLE 실행후 > 파일 > 열기(ctrl + O)

◆ 파이썬 Docs – python.org 사이트

▪ Documents > Python Docs > Tutorial(자습서)

Python 3.9.5 documentation

Welcome! This is the documentation for Python 3.9.5.

Parts of the documentation:

[What's new in Python 3.9?](#)

or all "What's new" documents since 2.0

[Tutorial](#)

start here

[Library Reference](#)

keep this under your pillow

[Language Reference](#)

describes syntax and language elements

[Python Setup and Usage](#)

how to use Python on different platforms

[Python HOWTOs](#)

In-depth documents on specific topics

[Installing Python Modules](#)

installing from the Python Package Index & other sources

[Distributing Python Modules](#)

publishing modules for installation by others

[Extending and Embedding](#)

tutorial for C/C++ programmers

[Python/C API](#)

reference for C/C++ programmers

[FAQs](#)

frequently asked questions (with answers!)

◆ 파이썬 Docs

▪ Tutorial(자습서) > Numbers

3.1.1. Numbers

The interpreter acts as a simple calculator: you can type an expression at it and it will write the value. syntax is straightforward: the operators `+`, `-`, `*` and `/` work just like in most other languages (for exam or C); parentheses `()` can be used for grouping. For example:

```
>>> 2 + 2
4
>>> 50 - 5*6
20
>>> (50 - 5*6) / 4
5.0
>>> 8 / 5 # division always returns a floating point number
1.6
```

변수(variable)

■ 변수란?

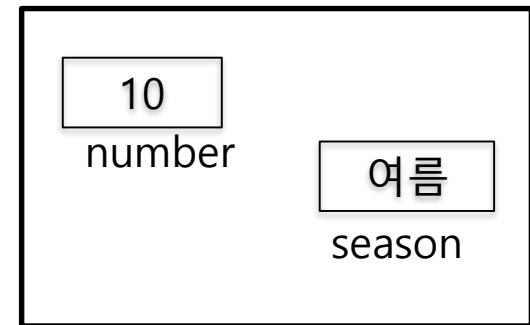
- 프로그램에서 사용되는 자료를 저장하기 위한 메모리 공간(영역)
- 프로그램 실행 중에 값 변경 가능, variable 이라 함

■ 변수의 선언 및 초기화

- 변수 선언은 어떤 타입의 데이터를 저장할 것인지 그리고 변수이름은 무엇인지를 결정한다. (자료형)은 생략함.

(자료형) 변수이름 = 초기값;

예) number = 10 , season = "여름"



변수(variable)

- 변수 사용 예제

```
# 상수 - 변하지 않는 수
print(4 + 5)
print(4 * 5)

# 변수 선언과 초기화
n1 = 4 #정수형 변수 n1에 4를 저장(기억)
n2 = 5

n1 = 10 #값을 변경(수정)

print("n1 + n2 =", n1 + n2)
print("n1 - n2 =", n1 - n2)
print("n1 * n2 =", n1 * n2)
print("n1 / n2 =", n1 / n2) #나누기는 소수로 출력됨

season = "여름" #문자형 변수 season에 "여름" 저장
print("계절 :", season)
```

변수(variable)

- 변수명(이름) 작성시 주의

- 변수 이름은 영문, 한글, 특수문자, 숫자의 결합으로 만든다.
- 변수 이름은 숫자로 시작할 수 없고, 공백이 있으면 안됨.
- 변수 이름은 대, 소문자를 구분함
- 예약어는 사용할 수 없음.(if, for, while 등)

```
# 변수이름 - 스네이크 표기법
name_of_fruit = "사과"
rate_of_birth = 0.75

print(name_of_fruit)
print(rate_of_birth)

# 변수 이름 작성시 오류
"""
3n = 100  #숫자로 시작 불가
sea son = "봄" #공백(문자)을 넣을 수 없다.
class = 3  #예약어(이미 지정된 명령어)는 사용 불가
"""
```

자료형(Type)

■ 자료형이란?

- 사용할 데이터의 종류에 따라 메모리 공간을 적절하게 설정해 주는 것
- 파이썬에서는 표기하지 않음 (예, num=10, name="홍길동", pi=3.14)

분류	자료형	설명	예
정수	int	소수점이 없는 수	-2, -1, 0, 1, 2
실수	float	소수점(.)이 있는 수, 부동소수점수라고도 불린다.	-3.5, 0.0, 1.25
문자열	str	알파벳과 다른 문자로 이루어진 문장	"a", 'hello', "비"
논리형	bool	참과 거짓을 표현	True, False(2가지 값만 있음)

type() 함수

- 자료형을 반환해주는 type() 함수

```
# 자료형 확인 - type()
# 숫자형
print(type(n1)) #int
print(type(1.609)) #float

# 문자형
print(type('B')) #str
print(type(season)) #str

# 논리형 자료형 - 결과값이 True / False임
print(4 > 5) # False
print(4 < 5) # True
print(type(4 > 5)) # bool
```

컴퓨터에서 데이터 표현하기

- 비트(binary digit)

bit(비트) : 컴퓨터가 표현하는 데이터의 최소 단위로 2진수 하나의 값을 저장할 수 있는 메모리의 크기

컴퓨터는 0과 1로만 데이터를 저장함(0-> 신호꺼짐, 1-> 신호켜짐)

- 비트로 표현할 수 있는 수의 범위

비트수	표현할 수 있는 범위(십진수)	
1bit	0, 1(0~1)	2^1
2bit	00, 01, 10, 11(0~3)	2^2
3bit	000, 001, 010, 011, 100, 101, 110, 111(0~7)	2^3

10진수를 2진수로 바꾸기

■ 진수 표현

10진수	2진수	16진수	10진수	2진수	16진수
1	0001	1	9	1001	9
2	0010	2	10	1010	A
3	0011	3	11	1011	B
4	0100	4	12	1100	C
5	0101	5	13	1101	D
6	0110	6	14	1110	E
7	0111	7	15	1111	F
8	1000	8	16	10000	10

자리 올림 발생

아스키 코드와 유니코드

- 숫자, 문자 표현 – 컴퓨터 내부에서는 숫자뿐만 아니라 문자도 2진수로 표현

- **아스키 코드(ASCII)**

- 숫자, 영문 알파벳 등 128개의 문자를 표기하도록 정한 코드값
- 1963년 미국 ANSI에서 표준화한 정보 교환용 7비트 부호 체계

- **유니코드(Unicode)**

- 전 세계 문자를 컴퓨터에서 일관되게 처리하기 위한 국제 표준. 각 문자에 고유한 숫자값이 할당되어 있음.
- 한글, 중국어 등 아스키 코드로 표현할 수 없는 문자 표기 가능
예) 한글은 약 11,000자
총 65536개의 문자 표현 가능

아스키 코드와 유니코드

■ 유니 코드 & 아스키 코드

문자	코드값	문자	코드값
	32	A	65
!	33	B	66
"	34	C	67
...
0	48	a	97
1	49	b	98
2	50	c	99

```
# ord(문자) - 문자를 유니코드(아스키) 값으로 변환
# chr(코드값) - 유니코드 값을 문자로 변환
print(ord('0'))    #48
print(ord('A'))    #65
print(ord('a'))    #97

print(chr(48))     #'0'
print(chr(65))     #'A'
print(chr(97))     #'a'
```

진수 표현

▪ 10진수, 2진수, 16진수

※ 10진수를 2진수로 바꾸기

$$10 = 1010(2)$$

8 4 2 1

$$1\ 0\ 1\ 0(1 \times 2^3 + 1 \times 2^1 \rightarrow 8 + 2)$$

진수 표현하기

```
num = 10
```

```
b_num = 0b1010
```

```
h_num = 0xA
```

```
print(num)
```

```
print(b_num)
```

```
print(h_num)
```

#진수 표현 함수

```
print(bin(10))
```

```
print(bin(65))
```

```
print(hex(10))
```

```
print(hex(65))
```

```
10 10 10
0b1010
0b1000001
0xa
0x41
```

자료형 변환(Type Conversion)

■ 형 변환

- 자료형은 각각 사용하는 메모리 크기와 방식이 다름
- **int**(문자) : 숫자로 변환함, **str**(숫자) : 문자로 변환함

```
# int(문자) - 문자를 정수로 변환
val_1 = "123"
val_1 = int(val_1)
print(val_1 + 10)

# 실수를 정수로 변환
val_2 = 3.1
val_2 = int(val_2)
print(val_2)

# str(숫자) - 숫자를 문자로 변환
val_3 = 123
#val_3 = str(val_3)
print(str(val_3))

# 정수를 실수로 변환
val_4 = 21
val_4 = float(val_4)
print(val_4)
```

```
133
3
123
21.0
```

문자열 다루기

▪ Text – 문자 쓰기

- 문자열 안에 따옴표 사용하기 – 서로 중복이 되지 않게 함
- 여러 줄로 출력하기 - 쌍따옴표 또는 홑따옴표 3개 사용("""~~""")
- 이스케이프 문자 사용하기

코드	내 용
\n	줄바꿈
\t	탭 – 문자열 간격
\'	어포스트로피(')

문자열 다루기

■ 문자열 예제

```
# 문자열 다루기
say1 = "'힘내세요!' \n라고 말했다."
print(say1)

say2 = "No, it doesn't"
print(say2)

say3 = 'No, it doesn\'t'
print(say3)

say4 = "문자 \"12\"를 숫자 12로 변환하세요"
print(say4)

# 문자열을 여러줄에 입력
table = """
상품명\t가격\t수량
키보드\t20000\t100
마우스\t25000\t100
모니터\t80000\t50
"""
print(table)
```

```
'힘내세요!'
라고 말했다.
No, it doesn't
No, it doesn't
문자 "12"를 숫자 12로 변환하세요
```

상품명	가격	수량
키보드	20000	100
마우스	25000	100
모니터	80000	50

항과 연산자

■ 항(operand)

- 연산에 사용되는 값

■ 연산자(operator)

- 연산에 사용되는 기호
예) $3 + 7$ (3과 7은 항, '+'는 연산자)



■ 연산자의 종류

구분	연산자	연산 예
대입연산자	'='	num = 10
산술연산자	+, -, *, /, //, %	3 + 7
비교연산자	>, >=, <, <=, ==, !=	7 == 3
논리연산자	and, or, not	7 >= 3 and 3 != 3
복합대입연산자	+=, -=, *=, /=	n += 1

대입 연산자

■ 대입 연산자

- 오른쪽의 값을 왼쪽의 변수에 대입
- '=' 연산자를 사용.

age = 20

user_id = "abc123"

- 연산의 순서

오른쪽 > 왼쪽

산술 연산 > 대입 연산

total = num1 + num2



'=' 은 대입 연산자

```
user_id = "smile"
```

```
password = "k1234"
```

```
print("user_id =", user_id)
```

```
print("password =", password)
```

```
print("user_id = " + user_id)
```

```
print("password = " + password)
```

대입 연산자 연습문제

■ 변수 값 교환하기

```
# 변수값 교환
x = 1
y = 2

print("==== 교환전 =====")
print("x =", x, ", y =", y)

# 교환
"""
temp = x
x = y
y = temp
"""

# 직접 교환
x, y = y, x

print("==== 교환후 =====")
print("x =", x, ", y =", y)
```

```
==== 교환전 =====
x = 1 , y = 2
==== 교환후 =====
x = 2 , y = 1
```

산술 연산자

■ 산술 연산자

연산자	연산 작업	설명
+	$n1 + n2$	더하기
-	$n1 - n2$	빼기
*	$n1 * n2$	곱하기
/	$n1 / n2$	나누기
//	$n1 // n2$	몫
%	$n1 \% n2$	나머지
**	$n1 ** n2$	거듭제곱

산술 연산자

■ 산술 연산자

```
n1 = 10  
n2 = 4
```

```
print("n1 + n2 =", n1 + n2)  
print("n1 - n2 =", n1 - n2)  
print("n1 * n2 =", n1 * n2)  
print("n1 / n2 =", n1 / n2) # 나누기  
print("n1 // n2 =", n1 // n2) # 몫  
print("n1 % n2 =", n1 % n2) # 나머지  
print("n1 ** n2 =", n1 ** n2) # 거듭제곱
```

```
n1 + n2 = 14  
n1 - n2 = 6  
n1 * n2 = 40  
n1 / n2 = 2.5  
n1 // n2 = 2  
n1 % n2 = 2  
n1 ** n2 = 10000
```

복합 대입 연산자

■ 복합 대입 연산자

연산자	연산 작업
+=	val += 10
-=	val -= 10
*=	val *= 10
/=	val /= 10
%=	val %= 10

```
val = 20
```

```
val += 10 # val = val + 10  
print(val)
```

```
val -= 10 # val = val - 10  
print(val)
```

```
val *= 10 # val = val * 10  
print(val)
```

```
val /= 10 # val = val / 10  
print(val)
```

```
val %= 10 # val = val % 10  
print(val)
```

비교 연산

■ 비교 연산자

연산자	의미	예	결과
<	보다 작다	9 < 10	True
>	보다 크다	9 > 10	False
<=	작거나 같다	9<=10	True
>=	크거나 같다	9>=10	False
==	같다	9==10	False
!=	같지 않다	9!=10	True
is	같다(객체)	a is b	False
is not	같지 않다(객체)	a is not b	True

비교 연산

■ 비교 연산자

```
x = 10
y = -10

print(x > 0)
print(y > 0)
print()

print(x > y)
print(x < y)
print()

print(x == 10)
print(x == y)
print(x != y)
print(x is y)
print(x is not y)
print()
```

```
True
False
```

```
True
False
```

```
True
False
True
False
True
```

논리 연산

■ 논리 연산자

연산자	규칙
x and y	x, y 가 모두 참이면 참, 나머지는 거짓
x or y	x, y 중 둘 중 하나가 참이면 참
not x	x가 참이면 거짓, 거짓이면 참

👉 **Python Docs > Library Reference > Built-in Types**

Operation	Meaning
<	strictly less than
<=	less than or equal
>	strictly greater than
>=	greater than or equal
==	equal
!=	not equal
is	object identity
is not	negated object identity

Boolean Operations — and, or, not ¶

These are the Boolean operations, ordered by ascending priority:

Operation	Result	Notes
x or y	if x is false, then y, else x	(1)
x and y	if x is false, then x, else y	(2)
not x	if x is false, then True, else False	(3)

논리 연산

■ 논리 연산자

```
# and - 2개의 조건이 모두 참일때 참이다.  
# or - 2개의 조건중 1개만 참이어도 참이다.  
print(x > 0 and y > 0) #False  
print(x > 0 or y > 0) #True  
print(not (y > 0))     #True
```

비트 연산

■ 비트 이동 연산자

연산자	규칙
$a \ll 2$	a를 왼쪽으로 2비트 이동
$b \gg 3$	b를 오른쪽으로 3비트 이동

원본 데이터 옮길 비트의 수

```
num = 5 #00000101
val_1 = (num) << (2) #00010100
print(val_1)

val_2 = (num >> 1) #00000010
print(val_2)
```

비트 연산

■ 비트 논리 연산자

연산자	규칙
10 & 13	10과 13의 비트 논리곱을 수행 (1010 & 1101)
10 13	10과 13의 비트 논리합을 수행 (1010 1101)
~10	1은 0으로 0은 1로 반전

```
num1 = 10    # 00001010
num2 = 13    # 00001101

val_3 = num1 & num2 # 00001000
print(val_3)

val_4 = num1 | num2 # 00001111
print(val_4)
```



```
10 : 0 0 0 0 1 0 1 0
& 13 : 0 0 0 0 1 1 0 1
-----
      0 0 0 0 1 0 0 0
```



```
10 : 0 0 0 0 1 0 1 0
| 13 : 0 0 0 0 1 1 0 1
-----
      0 0 0 0 1 1 1 1
```

문자열 연산

- 문자열 더해서 연결하기(Concatenation)

+ : 연결 연산자

- 문자열 곱하기

* : 곱하기 연산자

```
head = "Good"
tail = " Job!"
print(head + tail)
print(head * 3)

print('=' * 10)
print(head + tail)
print('=' * 10)
```

```
Good Job!
GoodGoodGood
=====
Good Job!
=====
```

파이썬의 입력 처리

➤ input() – 입력 함수

```
"""
print("문자 입력: ")
ch = input()
print(ch)

"""

ch = input("문자 입력: ")
print(ch)
```

문자 입력:
happy
happy

문자 입력: apple
apple

파이썬의 입력 처리

➤ input() – 입력 함수

```
# 정수에 1 더하기
num = input("정수 입력: ")
num = int(num) #문자를 정수로 변환
print(type(num))

print(num + 1)

# 몸무게의 2배 계산
weight = input("몸무게 입력: ")
weight = float(weight) #문자를 실수로 변환
print(type(weight))
```

파이썬의 입력 처리

➤ input() – 입력 함수

```
print("이름을 입력해 주세요:")  
name = input()  
print(name + '님 반갑습니다.')  
age = input('나이를 입력해 주세요: ')  
age = int(age)  
print("당신의 나이는 " + str(age) + "세 이군요!")
```

나이 계산 프로그램

나이 계산 프로그램

나이를 입력 받아 아래의 결과처럼 계산하는 프로그램을 작성하세요.

```
# 나이를 계산하는 프로그램
birth_year = input("태어난 연도를 입력하세요: ")

# 나이 = 현재년도 - 태어난년도
age = 2025 - int(birth_year) + 1 #문자를 정수로 변환
#print(birth_year, "년에 태어난 사람의 나이는", age, "세 입니다.")

# 출력할때는 숫자를 문자로 변환함
print(str(birth_year) + "년에 태어난 사람의 나이는 " +
      str(age) + "세 입니다.")

# f포맷 방식 - f다음에 쌍따옴표로 감싸고, 변수는 중괄호({})로 감싼다.
print(f"{birth_year}년에 태어난 사람의 나이는 {age}세 입니다.")
```


실습 문제 1 – 산술 연산

몫과 나머지 계산하기

30개의 빵을 4명이 나눠 가질때 몫과 나머지를 구하세요.

👉 실행 결과

빵의 개수	:	7
남은 빵의 개수	:	2

실습 문제 2 - 입력

사각형의 넓이 계산하는 프로그램

가로와 세로의 길이를 입력 받아 넓이를 계산하는 프로그램을 작성하세요.

☞ 실행 결과

가로의 길이 : 4
세로의 길이 : 5
가로 길이 : 4cm
세로 길이 : 5cm
면적 : 20cm