

01

왜 파이썬인가?



1 문법이 쉽고, 프로그램의 작성이 간단함

2 유지 보수 및 관리가 용이함

3 실무에서 많이 사용되어 활용함

4 학습 환경이 풍부함

5 라이브러리가 풍부하여 확장성이 좋음

2 파이썬으로 무엇을 할 수 있는가?



할 수 있는 일

시스템 유틸리티
제작

임베디드 :
IOT 분야

GUI
프로그래밍

웹
프로그래밍

수치 연산
프로그래밍

데이터 분석

2 파이썬으로 무엇을 할 수 있는가?



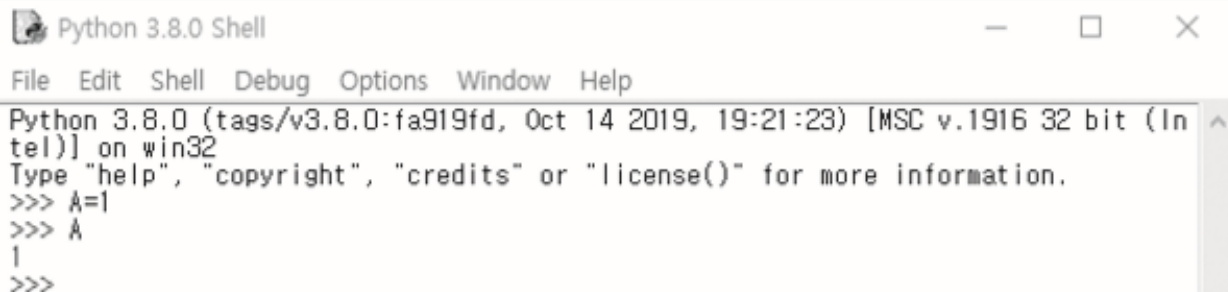
할 수 없는 일

시스템과
밀접한 프로그래밍
영역



모바일
프로그래밍

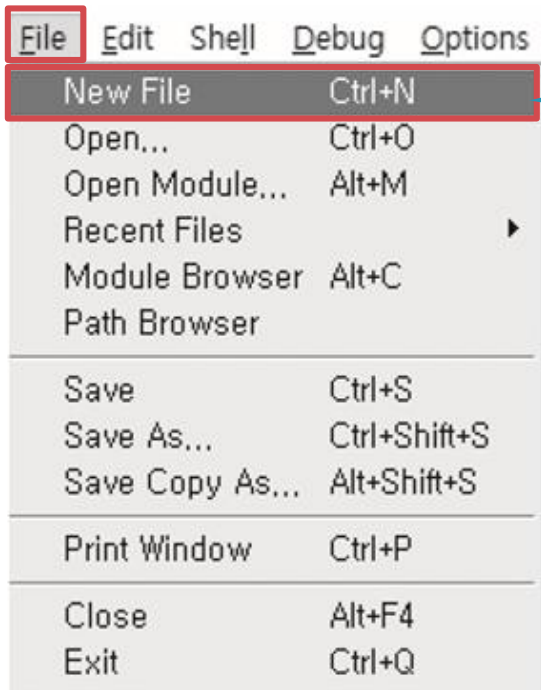
✓ IDLE 메뉴를 실행하면 IDLE Shell창이 먼저 나옴



```
Python 3.8.0 Shell
File Edit Shell Debug Options Window Help
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:21:23) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> A=1
>>> A
1
>>>
```

Shell창에서는 파이썬을
실행할 수 있지만, 저장은 할 수 없음

✓ 프로그램은 IDLE 편집창에서 작성함



편집창 실행

✓ 프로그램 실행

- IDLE 편집 창 메뉴에서 [Run → Run Module] 선택
(단축키 : F5)



02 변수와 연산자



변수 (Variable)

쉽게 변하는 수



프로그램이 동작하면서 어떤 상황 혹은 상태에 따라 변화하는 어떤 자료(데이터)를 담아 두기 위해 사용하는 개념

☑ 변수는 컴퓨터에서 값을 저장하는 기억장치(메모리) 공간

☑ '변수명 = 값' 형식으로 사용

특징

일시적으로 자료를 저장하는 공간임

변수에 저장된 값은 변할 수 있음

변수에는 숫자, 문자열 등 모든 자료형을
저장할 수 있음

변수에는 다른 변수의 값도 저장할 수 있음

변수는 사용되기 전에 반드시 할당되어 있어야 함

1

영문자, 숫자, 언더바(_)로 구성될 수 있으며
첫 글자는 반드시 영문자/언더바(_)로 시작함

2

공백이 들어가면 안 됨

3

대문자와 소문자를 구별함



Name과 name은 다른 변수!

4

파이썬에서 다른 용도로 사용되는 예약어는
변수명으로 사용할 수 없음

“ 변수에 들어 있는 **자료의 값에 따라**
변수의 형태가 결정됨 ”

✓ 파이썬은 다른 프로그래밍 언어와는 달리 값을 할당하면
그때 타입이 결정됨

✓ `type(변수이름)`을 통해 변수의 타입을 알아볼 수 있음

3 다양한 타입의 변수

기본 자료형의 종류

자료형	의미	예시
int	integer, 정수	n=100
float	float, 부동 소수점	n=95.5
str	string, 문자열	n='Kim' n="Kim"
bool	boolean	n=True n=False

4 다양한 연산자



산술 연산자



대입 연산자



연산자	의미
=	왼쪽변수에 오른쪽 값을 할당
+=	왼쪽변수에 오른쪽 값을 더하고 결과를 왼쪽변수에 할당
-=	왼쪽변수에 오른쪽 값을 빼고 결과를 왼쪽변수에 할당
*=	왼쪽변수에 오른쪽 값을 곱하고 결과를 왼쪽변수에 할당
/=	왼쪽변수에 오른쪽 값을 나누고 결과를 왼쪽변수에 할당
...	...

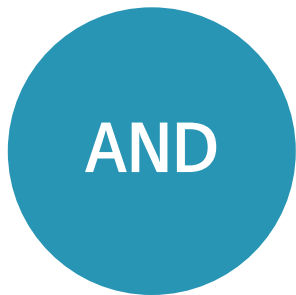


비교(관계) 연산자

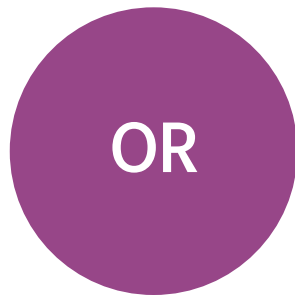
연산자	의미
==	값이 동일하다
!=	값이 동일하지 않다
>	왼쪽 값이 오른쪽 값보다 크다
>=	왼쪽 값이 오른쪽 값보다 크거나 동일하다
<	왼쪽 값이 오른쪽 값보다 작다
<=	왼쪽 값이 오른쪽 값보다 작거나 동일하다

논리 연산자

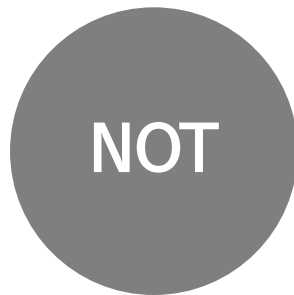
논리  연산




둘 다 참일 때만 참



둘 중 하나만
참이어도 참



논리 상태를
반전

4 다양한 연산자

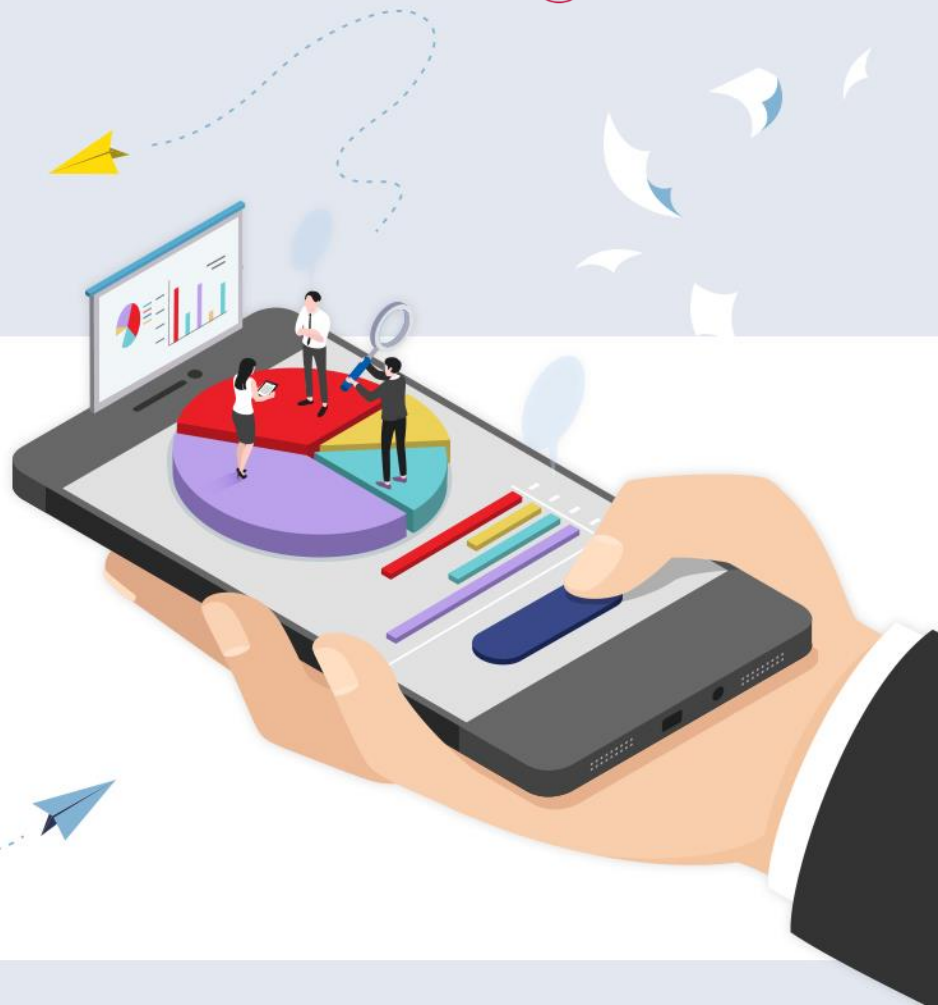


문자열 연산자

연산자	의미
+	문자열 더하기
*	문자열 곱하기

03

기본자료형과 입출력 함수



“ 자료형 = 데이터형 ”

자료의 종류를 구분해 놓은 것

정수형

age=21

실수형

score=95.5

문자열형

name='python'
name="python"

불(bool)형

work=True
work=False

input() 함수

- 함수 내에 안내할 문자열을 포함시켜 사용
- 파이썬은 이 함수에 의해 입력 받은 모든 자료를 문자열로 저장
- 다양한 자료형 변환 함수와 함께 사용됨

안내 메시지를 출력하고 사용자가
입력한 값을 문자열 형태로 받음

```
name = input('이름 입력:')
```

변수

print() 함수

✓ 파이썬을 이용하여 자료를 출력할 때 사용

✓ 파이썬의 출력형태

- 콤마(,)로 구분하여 출력하는 형태
- % 형식지정자를 이용하는 형태
- format() 함수를 이용하는 형태

04

컬렉션 자료형



컬렉션 자료형

여러 개의 값을 하나의 변수에 담을 수 있으며
변수 안에 공간을 여러 개 가지며,
변수 안에 서로 다른 공간을 찾는 방법

1 컬렉션 자료형의 개념과 필요성



컬렉션 자료형 종류	생성방법
리스트 (List)	[]
튜플 (Tuple)	()
딕셔너리 (Dictionary)	{키:값}
세트 (Set)	{ }



문법

☑ 대괄호 [] 안에 서로 다른 자료형의 값을 콤마(,)로 구분해 하나 이상 저장할 수 있는 컬렉션 자료형

- 요소(Element)
 - 대괄호 []에 넣는 자료
 - 순서를 가지고 있고 인덱스를 사용하여 참조 가능



문법

**리스트 문법****리스트명 = [값1, 값2, 값3,]**



인덱싱 및 슬라이싱

인덱싱(Indexing)

및

슬라이싱(Slicing)

무엇인가 ‘가리킨다’

무엇인가 ‘잘라낸다’



조작함수

- ☒ 리스트변수 이름 뒤에 마침표(.)를 붙인 다음
함수 이름을 사용



조작함수

함수	설명	사용법
append()	리스트에 요소를 마지막 위치에 새로 추가	리스트.append(값)
insert()	리스트의 해당 위치에 요소를 새로 삽입	리스트.insert(위치,값)
sort()	오름차순정렬 내림차순정렬	리스트.sort() 리스트.sort(reverse=True)
count()	해당 요소의 개수를 반환	리스트.count(찾을값)
pop()	리스트 제일 뒤의 항목을 빼내고, 빼낸 항목은 삭제 제거할 위치에 있는 요소를 제거	리스트.pop() 리스트.pop(위치)
remove()	해당 요소를 찾아 삭제	리스트.remove(삭제할값)

문법

✓ () 안에 서로 다른 자료형의 값을 콤마(,)로 구분해
하나 이상 저장할 수 있는 컬렉션 자료형

✓ 0부터 시작하는 인덱스를 이용해 접근할 수 있고
한 번 저장된 요소는 변경할 수 없음

튜플 문법

튜플명 = (값1, 값2, 값3,)

문법

- ✓ {}안에 키:값 형식의 항목을 콤마(,)로 구분해 하나 이상 저장할 수 있는 컬렉션 자료형
- ✓ 키를 먼저 지정하고 :(콜론)을 붙여서 값을 표현
 - 키와 값은 1:1 대응관계

딕셔너리 문법

딕셔너리명 = {key1:value1, key2:value2, key3:value3,...}

조작함수

함수	설명	사용법
get()	항목접근하기	딕셔너리.get(key)
pop() del ()	항목 꺼내고 삭제하기 항목삭제하기	딕셔너리.pop(key) del(딕셔너리[key])
items()	딕셔너리에 저장된 항목	딕셔너리.items()
keys()	딕셔너리에 저장된 키	딕셔너리.keys()
values()	딕셔너리에 저장된 값	딕셔너리.values()

문법

- ✓ 집합에 관련된 것을 쉽게 처리하기 위해 만든 자료형으로
중복을 허용하지 않는 컬렉션 자료형
- ✓ {}안에 항목을 콤마(,)로 구분해 하나 이상 저장할 수 있는
컬렉션 자료형

세트 문법

세트명 = {}

05 선택문



조건식이 만족하면 한 번 실행하고,
만족하지 않으면 해당 구간을 건너뛰게 됨



파이썬의 **선택문**

파이썬의 선택문



if문을 이용하여 조건식의 결과가
True 인지, False 인지에 따라
코드를 실행하거나 실행하지 않는 선택적 구조

예 if, if~else, if~elif~else 등

규칙



if문의 조건식들은 논리 연산이 가능한 문장으로 참/거짓 판별이 가능하도록 작성해야 함

규칙



if문의 조건식 끝에는 항상 :(콜론)이 있어야 함

규칙



실행 코드는 반드시 공백(스페이스바 4칸 또는 Tab)으로 들여쓰기(indent)를 하여
if문에 포함되는(중속) 코드로 작성해야 함

선택문을 표현하기 위해 필요한 조건식은
관계연산자와 논리연산자를 이용하여 표현

조건식이 만족

True

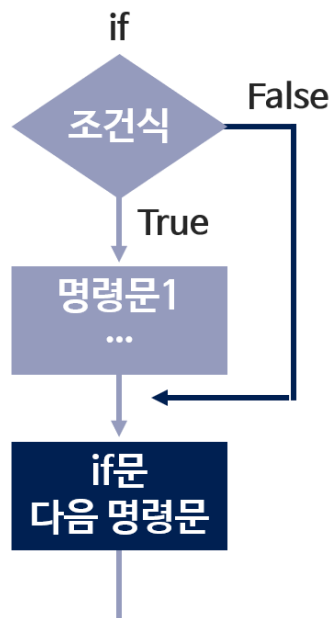
조건식이 불만족

False

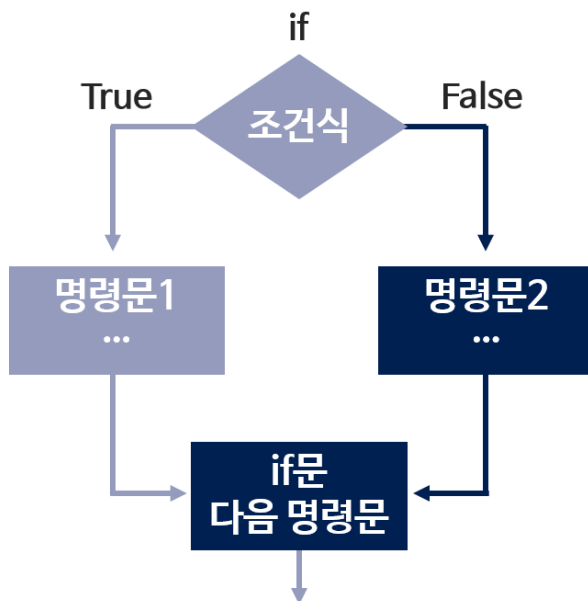
조건식의 예

- **total >= 100**
 - total 변수에 저장된 자료가 100 이상이면 만족하는 조건식
- **6 <= age < 60**
 - age 변수에 저장된 자료가 6 이상, 60세 미만이면 만족하는 조건식
- **num % 3 == 0**
 - num 변수에 저장된 자료를 3으로 나눈 나머지가 0 이면 만족하는 조건식
- **first != second**
 - first 변수와 second 변수에 저장된 자료가 다르면 만족하는 조건식
- **kor >= 90 and eng >= 90**
 - kor 변수에 저장된 자료가 90 이상이고, eng 변수에 저장된 자료가 90 이상이면 만족하는 조건식

if문



if~else문



if~elif~else문

