파이썬 프로그래밍

2. 프로그래밍 기초

00

Contents

❖ 수업 목표

- 변수에 대해 설명할 수 있고, 기본 자료형의 종류를 설명할 수 있다.
- 데이터를 입력 받고 출력할 수 있고, 연산자 종류와 사용법을 설명할 수 있다.

❖ 세부 목표

00

1. 프로그램, 데이터

❖ 프로그램이란?

- 컴퓨터가 무엇을 해야 할지 미리 작성해 놓은 계획
  - 문제 해결을 위해 필요한 데이터를 변수에 저장하고 처리방식(명령어)과 명령어들의 순서를 결정(알고리즘)하는 것의 집합

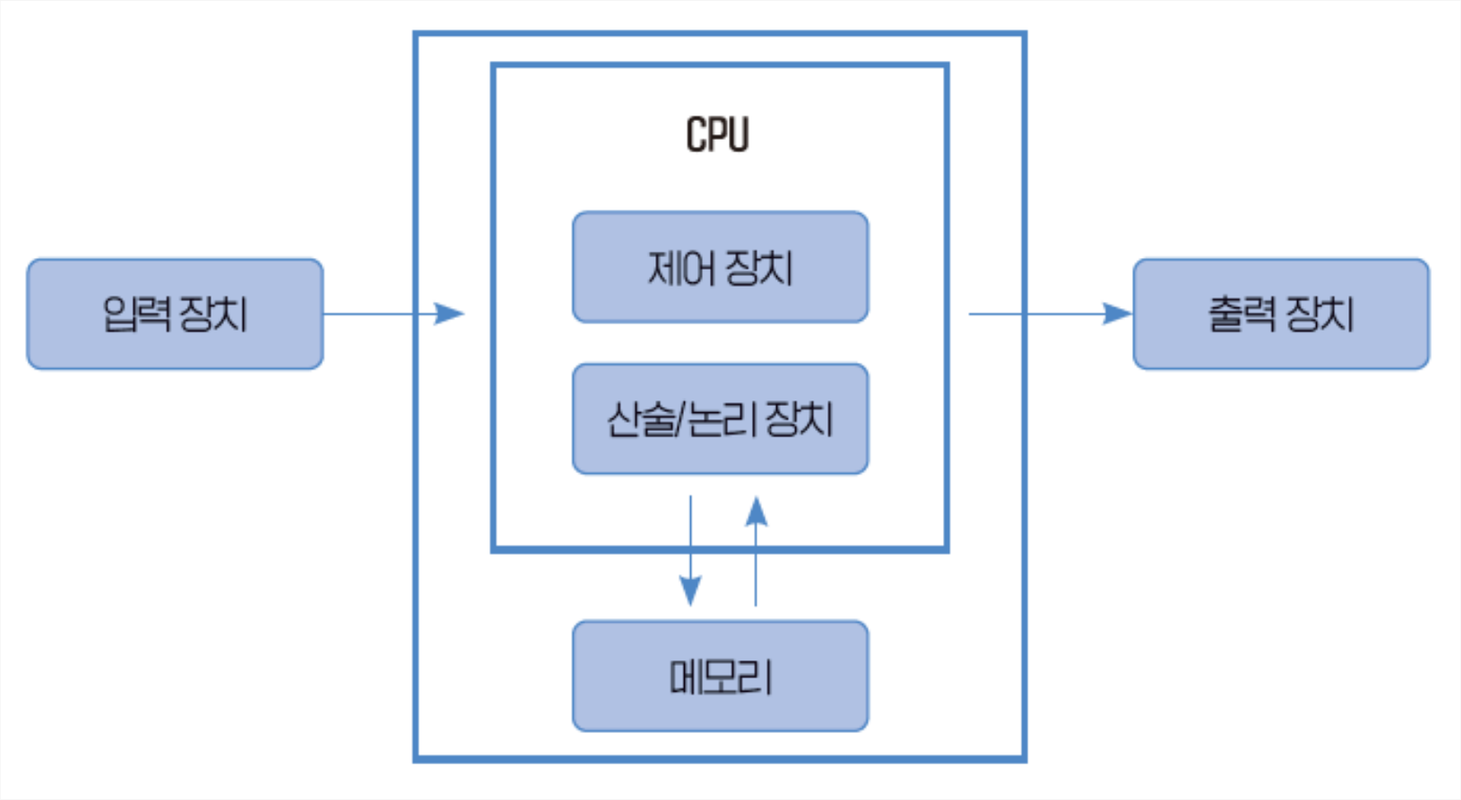
- 프로그램 = 데이터(data) + 명령어(instruction)

1. 프로그램, 데이터

❖ 컴퓨터 구조

▪ 폰 노이만 아키텍처

- 프로그램 실행시, 데이터(값)를 메모리(변수명으로 접근)에 저장
- 저장된 값을 순차적으로 CPU로 불러들여 명령어로 처리



1. 프로그램, 데이터

❖ 컴퓨터 프로그램 실행 과정

- (1) 키보드, 마우스 등 표준 입력 장치를 통해 정보 획득(입력)
- (2) 입력 받은 데이터(Data)를 메인 메모리에 저장
- (3) CPU가 데이터를 명령어 순서대로 처리
- (4) 처리 결과를 표준 출력 장치(모니터)에 표시(출력)

1. 프로그램, 데이터

❖ 데이터 (Data, 자료)

- 프로그램을 운용할 수 있는 형태로 기호화, 숫자화 한 자료
- 프로그램의 기본 재료

- **데이터 타입** (Data type, 자료형)
  - 자료를 종류에 따라 구분한 것

- **데이터 타입 종류**
  - 숫자형(정수, 실수), 문자열, 불리언, 리스트, 딕셔너리, 클래스 등

1. 프로그램, 데이터

❖ 숫자형

- 숫자로 이루어진 자료형

코드 2-1 다양한 숫자를 출력하는 코드

```
>>> print(1)
1
>>> print(-2)
-2
>>> print(3.14)
3.14
```

1. 프로그램, 데이터

❖ 문자열 (String)

- 문자를 나열한 자료형
- 작은따옴표, 큰따옴표 등으로 표현
- 따옴표 안에 문자, 숫자, 기호 등 넣음

코드 2-5 문자열을 출력하는 코드

```
>>> print('Hello World!')
Hello World!
>>> print('3.14')
3.14
>>> print('토끼야 안녕!')
토끼야 안녕!
>>> print("토끼야 안녕!")
토끼야 안녕!
```

2. 변수, 명령어

❖ 변수 (Variable)

- 값을 저장하는 공간
- 저장 위치는 컴퓨터의 RAM(주기억장치, 메인 메모리(Memory))
- 저장 위치를 숫자(주소)로 기억
- 숫자뿐 아니라 문자열 등 다양한 자료형 저장 가능

2. 변수, 명령어

❖ 변수에 이름 짓기

- 문자, 숫자, _ 사용 가능
- 대문자와 소문자 구분
- 공백 사용 불가
- 숫자로 시작 불가
- 미리 예약된 이름(Keyword) 사용 불가
  - import keyword
  - print(keyword.kwlist)

- 가능한 예
  - i, my_int, count, numbers
- 불가능한 예
  - 3rd_num, my string, if, for

2. 변수, 명령어

❖ 명령어

- 데이터가 저장된 변수들을 논리적 순서에 따라 처리하는 과정
  - 조건문, 반복문, 함수

- 데이터를 다루기 위해 입출력 함수의 사용
  - 출력 명령: print() 함수
  - 입력 명령: input() 함수

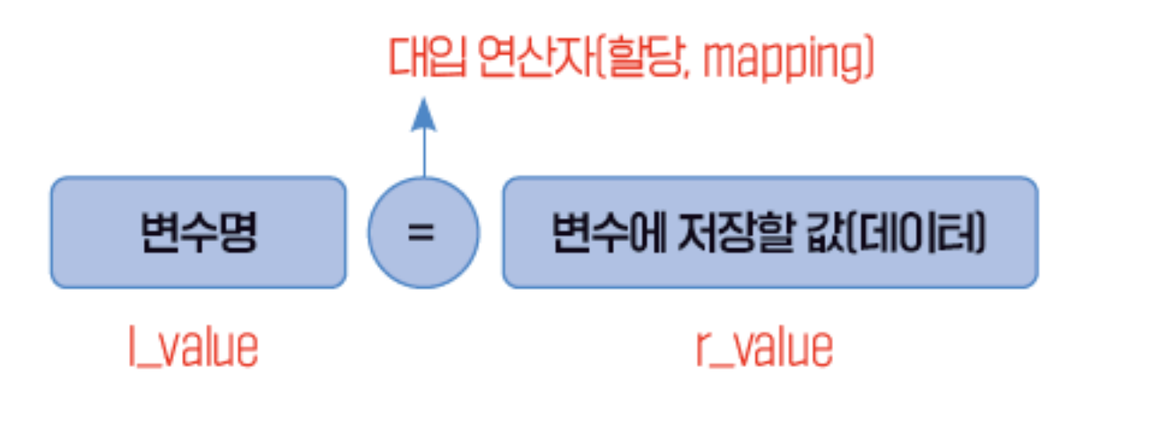
  - ※ 함수(Function): 특정 동작(기능)을 수행하는 코드 집합

3. 변수와 데이터

❖ 변수의 데이터 할당

▪ 저장을 위해 사용하는 연산자

- = : 대입(할당, assignment)연산자



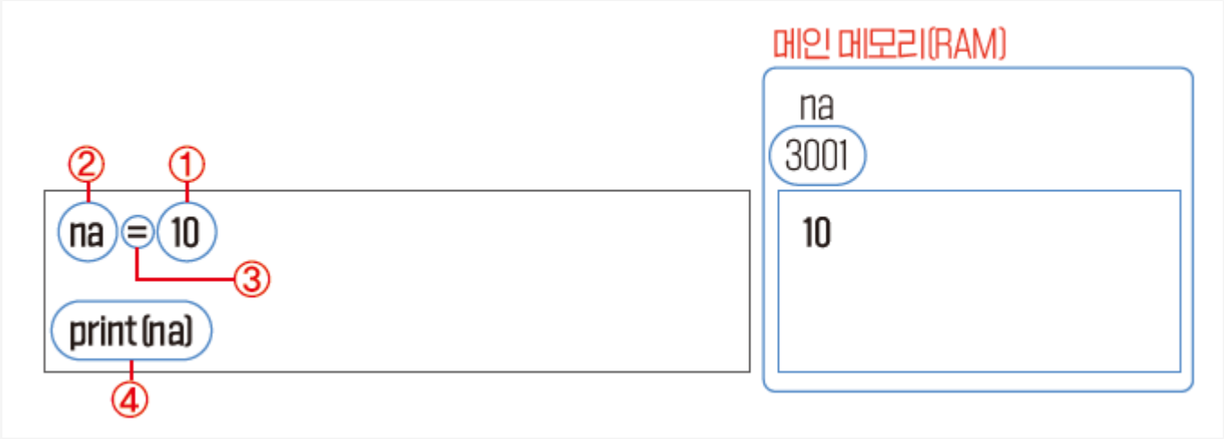
- 변수명: 대입 연산자 왼쪽 위치 (Left Value)
- 저장값: 대입 연산자 오른쪽 위치 (Right Value)

- 만약, 대입연산자 없이 변수가 사용된다면?
  - r_value(저장값)으로 해석함

3. 변수와 데이터

❖ 변수의 데이터 할당 과정

- 예) 파일명 [first.py] 로 프로그램을 작성 후 저장하고 실행



- ① 데이터 ‘10’의 저장을 위해 메인메모리(기억장소) 공간 확보 및 저장
- ② 기억 장소에 붙여줄 이름(변수명)을 ‘na’로 정의
- ③ 대입 연산자는 메모리 주소 3001번지와 변수명을 맵핑(연결(1:1), 할당)
- ④ print(na)를 실행. 표준 출력 함수 print()는 na 변수를 r_value로 해석하고 na 변수 값 10을 화면에 출력

3. 변수와 데이터

▪ 변수 값 저장 및 출력

코드 2-9 변수에 저장한 문자열을 출력하는 코드

```
>>> rainbow = '빨주노초파남보'
>>> print(rainbow)
빨주노초파남보
```

변수에 저장된 값은 언제든지 변할 수 있음

코드 2-10 변수에 저장한 값을 변경하는 코드

```
>>> count = 0
>>> print(count)
0
>>> count = 1
>>> print(count)
1
>>> count = count + 1
>>> print(count)
2
```

3. 변수와 데이터

❖ 변수의 데이터형 구별

- 컴퓨터가 기본적으로 구별하는 데이터형: 기본 데이터형
- 파이썬에서 각 변수의 데이터형은 r_value에 따라 결정

syntax : 숫자형 변수 정의

정수형 : int 실수형 : float 변수명 = 변수에 저장할 값(데이터)	na = 10 # -----> ① nb = 20.2 # ---> ②

- ① r_value가 10으로 정수이므로 na는 정수형 변수
- ② r_value가 20.2로 실수이므로 nb는 실수형 변수

3. 변수와 데이터

❖ 변수의 데이터형 구별

syntax : 문자형 변수 정의

문자형 : string **변수명** = **“변수에 저장할 값”**(데이터)	sa = "python" # -----> ①

- ① **r_value가 문자열이므로 sa는 문자열(String) 변수.**
  - **따옴표(작은따옴표, 큰따옴표 모두 가능)로 감싸 문자열의 시작과 끝을 표시**

4. 데이터 입력과 출력

❖ 표준 입/출력 장치와 함수

- 표준 입/출력 장치로부터 입력과 출력을 할 수 있도록 도와주는 표준 입출력 함수

syntax : 표준 입출력 장치와 함수

표준 입력 장치 : 키보드	표준 입력 함수 : input( )
표준 출력 장치 : 모니터	표준 출력 함수 : print( )

- 데이터 입력(키보드) → 변수 저장 후 명령어 처리 → 데이터 출력(화면)

- 표준 입출력 함수는 사전에 기본적으로 정의되어 있음
- 사전 정의된 함수 사용 시, 정해진 규칙(syntax, 문법)에 따라 사용해야 함

4. 데이터 입력과 출력

❖ 표준 출력 함수

syntax : 표준 출력 함수

• **표준 출력 함수** : print(인수) • **함수명** : print • **인수** : 모든 데이터형 가능, 변수 • **반환값(Return Value)** : 없음	na = 10 sa = "python" print(na) print(sa)

- **반환값 : 함수가 동작을 수행한 후 되돌려 주는 결과 값**
- **출력 함수는 표시를 위한 서식을 제공하지만 쉼표(,)로 여러 값 표현 가능**
- **print(변수)에서 대입 연산자 없이 사용되는 변수는 r_value이므로 변수에 저장된 값을 의미**
- **※인수(argument): 함수의 괄호 안에 작성되는 변수**

4. 데이터 입력과 출력

- **표준 출력 함수 예시 코드**

```
na=10
nb=20.2
sa="python"
print("na변수값", na)
print("nb변수값",nb)
print("sa변수",sa)
```

〈화면 출력〉

```
na변수값 10
nb변수값 20.2
sa변수 python
```

```
nc=30
nd=40
print("nc=", nc,"nd=", nd)
nd=nc
print("nc=", nc,"nd=", nd)
```

〈화면 출력〉

```
nc=30 nd=40
nc=30 nd=30
```

4. 데이터 입력과 출력

❖ 표준 입력 함수

syntax : 표준 입력 함수

- **표준 입력 함수** : input("인수")
- **함수명** : input
- **인수** : input("화면에 표시될 문자열")
- **반환값** : 입력한 값이 문자열 데이터형 (str)으로 반환됨. 정수값 같은 숫자형 데이터를 입력해도 문자열로 형 변환되어 반환됨. 이때 시각적으로 반환된 문자열이 input() 함수 위에 살포시 올라와 있다고 생각하자.

```
sa=input("문자열을 입력하세요.") ①
print(sa) ②
```

- ① input("문자열을 입력하세요.")에서 큰따옴표 사이의 문자열은 프로그램 실행 시 사용자가 입력해야 할 정보에 대한 설명을 작성함
  - input( ) 함수만 사용하면 커서만 깜박거리기 때문에 그 다음 단계에서 해야 할 일에 대한 설명이 필요함
- ② 키보드로 'python' 을 입력하면 입력한 문자열이 sa 변수에 할당됨

4. 데이터 입력과 출력

▪ 표준 입력 함수 예시 코드

```
# pinput.py로 아래 프로그램을 저장하고 실행한다.
print("첫 번째 정수를 입력하세요") # 20을 입력
ra=input( ) # ----------------------------------> ①
rb=input("두 번째 정수를 입력하세요") # 5를 입력 # ---> ②
rc=ra+rb # ------------> ③
print(ra, "+", rb, "값은", rc, "이다")
```

- **얼마가 출력될 것이라고 예상하는가?**
  - **연산자 +는 피연산자의 값에 따라 2가지 기능으로 동작**
  - **(1) 피연산자가 모두 숫자이면 우리가 생각하고 있는 덧셈 연산 수행**
  - **(2) 피연산자가 모두 문자열이면 문자열과 문자열을 연결하는 연결자 역할 수행**

5. 연산자와 주석

❖ 연산자 (Operator)

- 연산할 때 사용하는 기호

- \+ (더하기)
- \- (빼기)
- * (곱하기)
- / (나누기)

코드 2-2 더하기, 빼기, 곱하기, 나누기를 하는 코드

```
>>> print(1 + 2)
3
>>> print(3 - 2)
1
>>> print(2 * 4)
8
>>> print(6 / 3)
2.0
```

더하기 / 빼기 / 곱하기 / 나누기

- ** (제곱)
- // (몫)
- % (나머지)

코드 2-3 제곱, 몫, 나머지를 구하는 코드

```
>>> print(5 ** 2)
25
>>> print(5 // 2)
2
>>> print(5 % 2)
1
```

5. 연산자와 주석

❖ 주석 (Comment)

- 사람을 위해 남기는 설명
  - 컴퓨터가 기계어로 번역하는 과정에서 제외함
- # 사용
  - 뒤의 문장을 처리하지 않고 넘어감

코드 2-12 주석으로 내용을 설명하는 코드

```
>>> coffee = 4100  # 커피의 가격
>>> juice = 4600  # 주스의 가격
>>> tea = 3900  # 홍차의 가격
```

  - 특정 문장 잠시 제외하기도 가능

코드 2-13 주석으로 문장을 제외하는 코드

```
>>> print('토끼야 안녕!')
토끼야 안녕!
>>> # print('토끼야 안녕!')
>>>
```

5. 연산자와 주석

❖ 주석 (Comment)

- 여러 줄 주석 처리
  - 시작 따옴표 3개와 종료 따옴표 3개로 감싸기
    - 작은따옴표, 큰따옴표 모두 가능

  - 예시 1) '''이것은
    주석'''
  - 예시 2) """이것은
    주석"""

6. 자료형 변환

❖ 자료형 변환

- input() 함수로 정수 데이터를 받아들여도 반환 값은 문자열 타입
- 입력 값으로 산술 연산 시, 문자열 타입을 정수 타입으로 형 변환 필요

- 자료형 변환 함수

syntax : 자료형 변환

- **정수형으로 변환하기** : int( )
- **실수형으로 변환하기** : float( )
- **문자형으로 변환하기** : str( )
- **데이터형 확인하기** : type( ), 사용하는 데이터나 변수의 데이터형을 확인하기 위해 제공하는 내장 함수이다.

6. 자료형 변환

❖ 자료형 변환

▪ 자료형 변환 코드

- input() 함수로 받아들인 데이터를 각각의 변수에 저장
- 저장한 변수의 데이터형을 int형으로 변환하여 산술 연산 실행

```
print("첫 번째 정수를 입력하세요.") # 20을 입력하세요.
ra=input()
rb=input("두 번째 정수를 입력하세요.") # 5를 입력하세요.

# 형 변환하기
ra=int(ra)
rb=int(rb)
rc=ra+rb
print(ra, "+", rb, "결괏값은", rc, "이다.")
rd=ra  -  rb
print(ra, "-", rb, "결괏값은", rd, "이다.")
```

6. 자료형 변환

❖ 자료형 변환

▪ type( ) 함수로 데이터형 확인

- 각 변수의 데이터형 확인

```
print("첫 번째 정수를 입력하세요") # 20을 입력하세요.
ra=input( )
rb=input("두 번째 정수를 입력하세요") # 5를 입력하세요.
print(type(ra))
print(type(rb))

# 형 변환하기
ra=int(ra)
rb=int(rb)
print(type(ra))
print(type(rb))
rc=ra+rb
print(ra, "+", rb, "=", rc, "이다.")
rd=ra  -  rb
print(ra, "-", rb, "=", rd, "이다.")
```

00

7. 연습문제

❖ 아래와 같은 출력 결과가 나오도록 빈칸에 연산자를 넣으시오

```
>>> print(3 ▢ 1 ▢ 2)
2
>>> print(3 ▢ 1 ▢ 2)
4
>>> print(3 ▢ 1 ▢ 2)
1
```

❖ 다음 코드의 출력 결과를 적으시오

```
>>> num1 = 9
>>> num2 = 2
>>> print(num1 // num2)
▢
>>> print(num1 % num2)
▢
```

7. 연습문제

❖ 출력 결과가 다른 하나를 고르시오

- print(8 / 2)
- print(2 * 2)
- print(8.0 / 2)
- print(8.5 // 2)

❖ 딸기 11개를 3명의 친구에게 나눠줍니다. 나눠주고 남는 딸기의 개수를 출력하는 코드로 올바른 것을 고르시오

- print(11 / 3)
- print(11 % 3)
- print(11 // 3)
- print(11 ** 3)

7. 연습문제

❖ 자료형 변환

- 다음 코드의 출력 결과를 예상해보고 에러 발생 시 수정해보자.

〈실수형〉

```
a=input("실수를 입력하세요.")
b=input("두 번째 실수를 입력하세요.")
result=a+b
print(a, "+", b, "=", result)
result=a - b
print(a, "-", b, "=", result)
```

```
실수를 입력하세요10
두 번째 실수를 입력하세요7
10+7=107

---------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_20484\4092767159.py in <module>
      3 result=a+b
      4 print(a, "+", b, "=", result)
----> 5 result=a  -  b
      6 print(a, "-", b, "=", result)

TypeError: unsupported operand type(s) for -: 'str' and 'str'
```

7. 연습문제

❖ 산술 연산자 활용

syntax : 산술 연산자(덧셈) 코드

```
na = 20
nb = 5

# na 변수의 값과 nb 변수의 값을 더하기 실행 순서
result = na + nb # --------> ① 더하기 연산자 ② 대입 연산자
       ②   ①
# 화면에 결괏값 출력하기
print(na, "+", nb, "=", result) # ----> ③ print 문 변경

#출력결과: 20 + 5 = 25
```

- ① 대입 연산자의 오른쪽의 연산을 먼저 실행. na 변수의 값과 nb 변수의 값 연산을 실행한 결과 값 계산 수행
- ② 대입 연산자의 오른쪽 값(r_value)이 대입 연산자의 왼쪽 변수(l_value)에 저장됨

5. 연산자, 주석(과제)

❖ 산술 연산자와 출력 함수 활용

- ③ print 명령문 실행 시 아래와 같이 출력되도록 print 문을 변경 작성해보자. 쉼표(,)를 사용해 복잡해 보이는 화면 출력도 모두 가능함

```
na 값 20 더하기 nb 값 5의 결괏값은 25다.
```

7. 연습문제

❖ 산술 연산자 활용

- 연산 프로그램 작성
- 산술 연산자(덧셈) 코드에 이어서 뺄셈, 곱셈, 나누기 연산을 추가하고 그 결과를 화면에 출력하는 프로그램을 작성해 보자.

```
10 – 20.2 = xx이다.
10 * 20.2 = xx이다.
10 / 20.2 = xx이다.
```

```
result=na - nb
print(na, "-", nb, "=", result,"이다.")
result=na * nb
print(na, "*", nb, "=", result,"이다.")
result=na / nb
print(na, "/", nb, "=", result,"이다.")
```

Notice

❖ 과제

- 1. 연산자 입력 문제 풀기
- 2. 예시 코드 출력 결과 예측하기
- 3. 자료형 변환 프로그램 작성하기
- 4. 산술 연산자 활용 프로그램 작성하기
- 5. 출력함수 활용 프로그램 작성하기
- 6. 자료형 변환 프로그램 작성하기

❖ 다음 수업 내용

- 제어문과 조건식
  - 제어문, 조건식, 비교연산자, 논리연산자, 연산자 우선순위, if문