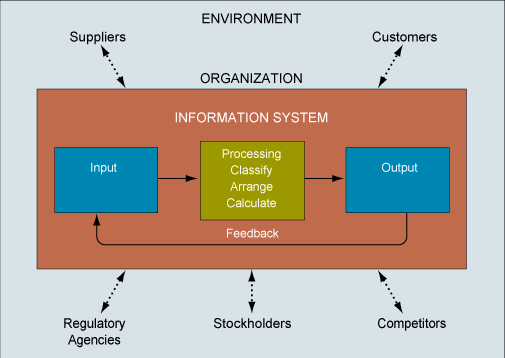
**2강. 프로그램의 입출력 다루기 – 변수와 자료형**

**2.1 입력-처리-출력이 갖춰지면 정보시스템이 된다.**

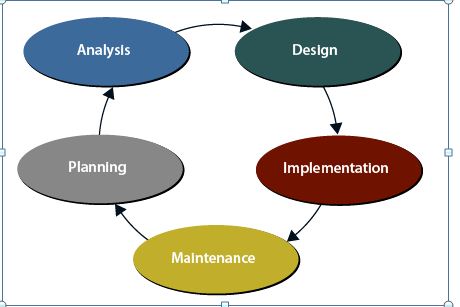
유용한 프로그램이 갖는 공통적인 특징은 데이터를 입력받아, 처리하여, 새로운 출력을 만들어낸다는 것이다.

데이터를 입력받아 원하는 방식으로 가공하고 처리한 후 사용자가 원하는 정보를 만들어 내는 컴퓨터와 프로그램을 묶어서 정보시스템(information system)이라고 부른다.

예를 들어, 비디오게임을 할 때를 가정해 보자. 조이스틱이나 마우스를 이용해서 게이머가 입력을 하면, 플레이스테이션과 같은 게임기가 입력값에 따라 캐릭터의 위치를 변경하는 등의 처리를 하게 되고, 디스플레이에 처리된 결과가 반영되어 캐릭터가 다른 곳에 나타나게 된다.



정보시스템은 다음과 같이 계획 -> 분석 -> 설계 -> 구현 -> 유지보수의 단계를 순환하게 되며, 프로그래밍은 구현단계의 주 업무가 된다.



퀴즈) 1강의 실습4)에서 만든 “PrintTest.py” 프로그램의 입력, 처리, 출력 부분은 각각 무엇인가요?

퀴즈) 1강의 실습5)에서 만든 “NumGuess.py” 프로그램의 입력, 처리, 출력 부분은 각각 무엇인가요?

입출력을 원활히 하려면 프로그램에 ‘변수(variable)’를 잘 활용해야 한다.

**2.2 메모리에 이름을 붙여 ‘변수’를 정의하자**

컴퓨터는 입력을 받으면 내부 어딘가에 저장을 해두고 처리해야 한다. 일차적으로 보관하는 장소는 컴퓨터의 메모리(memory) 영역이다. 메모리도 역시 ON/OFF 스위치들로 구성되어 있다.

메모리는 입력값을 스위치들의 나열, 즉 비트(bit)들의 나열로 만들어 저장한다. 여기서, 메모리에 값을 ‘저장’한다는 것은 스위치를 특정 패턴으로 바꾼 후 그 상태가 유지되도록 둔다는 것이다.

또한, 메모리에 저장된 값은 필요할 때 읽어서 활용할 수 있다.

또한, 해당 메모리에 저장된 값을 다른 값으로 변경할 수도 있다.

메모리의 특정한 위치에 값을 저장하고, 이를 읽어들이고, 다른 값으로 변경하고자 한다면 프로그램에서 그 위치를 계속 알고 있어야 한다.

파이썬 프로그램에서는 메모리의 특정 위치에 ‘이름’을 붙여주고, 이름을 이용하여 값을 저장하고 읽고 변경할 수 있게 해준다.

실습1) 다음을 IDLE에서 실행하세요.

>>> Professor = "JinJin"

>>> print Professor

*(결과값을 적어보세요)* ***JinJin***

퀴즈) 실습1)에서 저장하고자 한 데이터는 무엇인가요? 저장을 위해 메모리에 붙인 이름은 무엇인가요? 메모리에 들어있는 데이터를 읽어내어 화면에 보여주기 위해 print 명령어는 어떤 이름을 사용했나요? 실습1)의 실행 결과로 화면에 나타난 값은 무엇인가요?

위 실습1)의 프로그램의 >>> Professor = "JinJin" 부분을 설명하자면 다음과 같다.

- “JinJin”이라는 문자데이터를 저장했다가 출력하기 위해서는 메모리가 필요하다.

- 그래서, Professor라는 이름을 붙여 메모리를 확보하고자 한다.

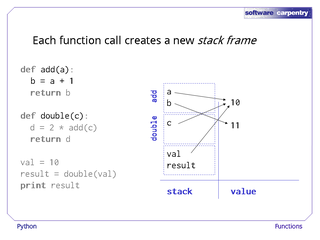
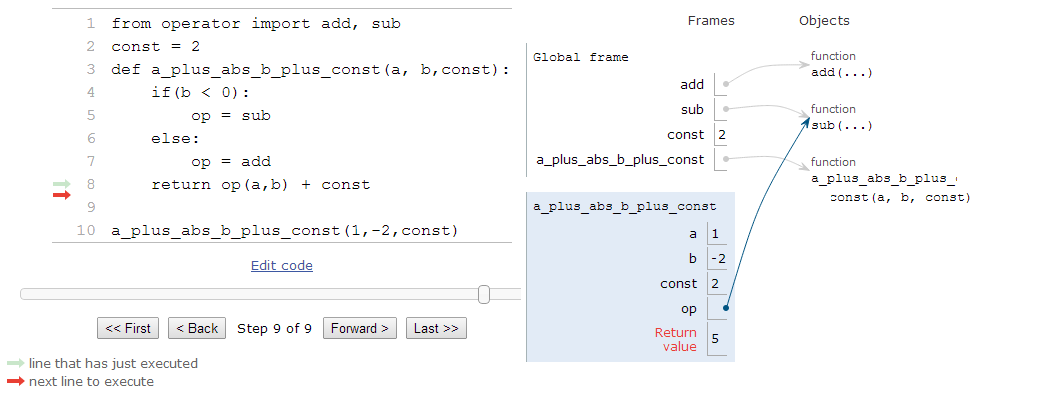
- 그리고, 그 메모리에 데이터를 저장하기 위해 등호(=)를 사용했다.

파이썬에서 등호(=)의 역할은 할당 혹은 지정(assign)을 의미한다. 파이썬은 등호가 나오는 명령어를 만나면 등호의 오른쪽을 먼저 처리한 후, 그 결과를 왼쪽에 적용한다.

먼저, 컴퓨터 메모리의 어느 곳엔가 “JinJin”이라는 문자열을 저장한다. 그리고, 그 값이 저장된 메모리의 위치값, 즉 주소값을 알고 있다.

메모리의 위치값은 매우 복잡한 값이다. 따라서, 이 값을 프로그래머가 기억하여 코딩하기엔 부적절하다. 파이썬은 위치값 대신 프로그래머가 지어준 Professor라는 이름을 사용하여 메모리의 특정 위치를 지정하도록 해준다. Professor라는 이름을 지어주는 것은 어떤 물건이나 장소에 라벨이나 태그, 혹은 스티커를 붙이는 것과 유사하게 생각하면 된다.

메모리에 이름붙인 것을 변수(variable)라고 한다.



**2.3 파이썬이 변수를 인식하는 방식**

변수의 이름은 키워드(keyword)나 지정어(reserved word)여서는 안된다. 파이썬은 프로그램을 수행하다가 어떤 이름을 만났을 때, 이것이 키워드나 지정어와 같은 경우 그 정해진 용도로 사용하려한다.

실습2) 키워드를 변수명으로 사용하려 하는 경우를 테스트해봅시다.

>>> print if

*(결과값을 적어보세요)* ***SyntaxError: invalid syntax***

>>> if = "JinJin"

*(결과값을 적어보세요)* ***SyntaxError: invalid syntax***

>>>

따라서, 변수명은 새로운 이름으로 지어주어야 한다. 변수를 잘 사용하려면 파이썬의 키워드와 지정어의 종류를 알아두고 이를 피해서 이름을 지어주어야 한다.

퀴즈) 여러분의 이름을 적어보세요.

퀴즈) 여러분, “이름”이라고 적어보세요.

앞의 실습1)에서 Professor는 따옴표 없이 적었고, “JinJin”은 따옴표를 붙여 적었다. 문자열에 따옴표를 붙인 경우와 그렇지 않은 경우 파이썬은 문자열을 전혀 다르게 취급한다.

문자열을 따옴표로 감싸게 되면 파이썬은 그것을 데이터 자체로 받아들인다. 하지만, 문자열이 따옴표로 싸여있지 않으면 파이썬은 그 문자열이 키워드나 변수명이라고 보고, 키워드가 아닌 경우 변수가 지칭하는 데이터가 무엇인지 파악하려고 한다.

실습3) 다음의 명령어의 실행결과는 무엇일지 테스트해봅시다.

>>> SyntaxError: invalid syntax

*(결과값을 적어보세요)* ***SyntaxError: invalid syntax***

>>> print 12

*(결과값을 적어보세요)* ***12***

>>> print "12 + 34“

*(결과값을 적어보세요)* ***12+34***

>>> print 12 + 34

*(결과값을 적어보세요)* ***46***

파이썬에서 숫자를 따옴표로 싸게 되면(예: “12”) 문자열(string)로 인식되며, 따옴표로 싸지 않게 되면 (예: 12)숫자 데이터값 그 자체로 인식된다.

따옴표는 큰따옴표, 작은따옴표 모두 사용이 가능하다. 다만, 여는 따옴표와 닫는 따옴표는 동일한 형태여야 한다.

숫자로 구성된 산술식(arithmetic expression)을 따옴표로 싸게 되면(예: “12 + 34”) 문자열 데이터로 인식되며, 따옴표로 싸지 않게 되면(예: 12 + 34) 수학의 계산 수식으로 인식하여 46이라는 값을 만들어 내게 된다.

다행이도 파이썬의 키워드나 지정어가 숫자로 만들어진 것은 없다!

**2.4 숫자값의 변수 만들기**

문자열 데이터가 아닌 숫자 데이터도 변수를 만들어 처리할 수 있다.

실습4) 다음의 명령어를 실행하여 결과를 확인해 보세요

>>> First = 7

>>> Second = 8

>>> print First, Second, First + Second, First \* Second

*(결과값을 적어보세요)* ***7 8 15 56***

7과 8이라는 숫자값을 반복적으로 참조하기 위해서 First, Second와 같은 변수를 만들어 사용하였다.

실습5) 다음의 명령어를 실행하여 결과를 확인해 보세요

>>> Result = First - Second

>>> print Result

*(결과값을 적어보세요)* ***-1***

변수를 사용하여 산술 계산을 하고, 그 결과값을 Result라는 다른 변수에 저장하였다. 같은 값을 두 개 이상의 변수명으로 저장하는 것도 가능하다.

실습6) 다음의 명령어를 실행하여 결과를 확인해 보세요

>>> Another = Result

>>> print Another

*(결과값을 적어보세요)* ***-1***

>>> print Result

*(결과값을 적어보세요)* ***-1***

Another라는 새로운 변수가 만들어지고 현재 Result 변수에 할당된 값이 Another에도 할당된다. 메모리의 새로운 영역에 데이터값을 저장하고 여기에 Another라는 태그를 붙여두었다고 생각하면 된다.

실습7) 다음의 명령어를 실행하여 결과를 확인해 보세요

>>> Another = Another + 10

>>> print Another

*(결과값을 적어보세요)* ***9***

>>> print Result

*(결과값을 적어보세요)* ***-1***

파이썬 프로그램을 코딩할 때 사용할 수 있는 산술연산자는 다음과 같다.

|  |  |  |
| --- | --- | --- |
| 산술연산자 | 의미 | 설명 |
| = | 할당 | 변수이름에 오른쪽 값을 지정해준다 |
| + | 더하기 | 좌우의 숫자값을 더한다. 문자열의 경우 연결해준다 |
| - | 빼기 | 좌측값에서 우측값을 뺀다 |
| += | 증가 | 좌측값에 1을 더해준다 |
| -= | 감소 | 좌측값에서 1을 빼준다 |
| \* | 곱하기 | 좌측과 우측의 값을 곱해준다 |
| / | 나누기 | 좌측값을 우측값으로 나눈다. 두 값 모두 정수인 경우 결과값은 정수인 몫만 나온다 |
| % | 나머지 | 좌측값을 우측값으로 나눈 나머지를 구한다. 정수값들만 계산가능 |
| \*\* | 지수승 | 좌측값을 우측값 회수만큼 거듭제곱한다. 정수와 실수 모두 계산가능 |

**2.5 변수명의 명명규칙 알아보기**

변수에는 어떤 이름도 붙일 수 있다. 글자와 숫자, 기호 등을 섞어서 지을 수 있다.

하지만, 변수명을 지을 때 지켜야 하는 몇 가지 규칙이 있다.

- 앞에서 언급한 대로 파이썬의 키워드/지정어는 사용할 수 없다.

* 변수명에는 글자, 숫자, 밑줄 문자(\_)만이 사용될 수 있다. 즉, 변수명에 공백(space)이나 \_를 제외한 문장부호들은 포함될 수 없다.

- 변수명에 사용되는 알파벳의 경우 대소문자가 구별된다는 점에 유의해야 한다. 즉, Professor와 PROFESSOR는 서로 다른 변수가 된다.

- 변수명은 반드시 글자나 밑줄 문자(\_)로 시작해야 한다. \_이 변수명의 맨 앞이나 마지막에 나올 때는 특수한 의미를 지니는 경우가 있으므로 유의한다. 또한, 앞에서 언급한 대로 변수명은 숫자로 시작할 수 없고, 다만 변수명에 포함하여 사용될 수는 있다.

- 변수명은 용도에 맞게 지어주는 것이 프로그램의 이해도를 높이고 유지보수를 용이하게 해준다. 두 단어 이상으로 변수명을 지을 때 공백 대신 \_를 사용하여 단어를 연결시켜준다.

퀴즈) 명명규칙에 맞게 잘 지은 변수명을 찾아보세요.

(1) student\_name = “Michel Foucault”

~~(2) survey-answer = “Nop!”~~

(3) YourAnswer = “Yes”

(4) \_my\_cart = “full of stuff”

~~(5) 2017resolution = “Stop smoking”~~

~~(6) Title of Movie = “LA LA LAND”~~

~~(7) Total\_number! = 300~~

~~(8) Width&Height = 30 \* 10~~

~~(9) End... = “Happy ending”~~

~~(10) RightAnswer? = “Bingo”~~

**2.6 자료유형(data type) 구분하기 : 숫자와 문자열**

프로그램에서 다루게 되는 자료유형, 즉 데이터타입에는 여러 종류가 있다. 컴퓨터에 명령을 내리는 과정에서 다양한 유형의 입력과 출력을 다루게 된다. 값의 유형에 따라 처리하는 방식이 달라지므로 자료유형을 잘 알고 사용할 수 있어야 한다.

이제까지는 숫자 유형과 문자열 유형을 다루어 왔다.

정리하자면, 문자열(string) 타입의 데이터는 글자, 숫자, 구두점 등의 값이 나열된 값들의 종류이다. 따옴표로 값들을 싸주어야 파이썬이 문자열 타입의 데이터로 인식할 수 있다.

숫자 타입의 데이터는 따옴표없이 숫자로만 구성된 값들의 종류이다.

실습8) 다음 명령어들을 실행하여 결과를 확인하세요.

>>> First = 10

>>> Second = 20

>>> First + Second

*(결과값을 적어보세요)* ***30***

>>> First = '10'

>>> Second = '20'

>>> First + Second

*(결과값을 적어보세요)* ***1020***

위의 결과에서 보는 바와 같이 따옴표로 싸인 값과 안 싸인 값은 유형이 다른 데이터로 취급된다. 파이썬은 + 라고 하는 동일한 연산자(operator)에 대해서 숫자값들끼리는 산술적 덧셈(summation)을 해주고, 문자열값들끼리는 문자열들을 이어붙이는(concatenation) 식으로 다르게 실행한다.

퀴즈) 그런데, 코딩을 하다가 여러 줄에 걸쳐진 긴 문자열을 다루어야 할 때는 어떻게 처리해야 할까요?

실습9) 여러 문장으로 구성된 문자열을 변수에 지정해 봅시다.

>>> Longstring = "I am a student. I have been enjoying my campus life. *(Enter)*

*(결과값을 적어보세요)* ***SyntaxError: EOL while scanning string literal***

긴 문자열을 입력하기 위해 엔터를 쳐보았다.

하지만, IDLE 인터랙션 모드에서는 하나의 명령어 입력이 끝났다고 인식하고 바로 실행을 하게 되고 오류가 나게 된다.

오류메시지의 내용은 문법이 잘못되었다는 것, 문자열이 어디까지인지 알기 위해 따옴표가 나올 때까지 스캐닝하던 중 따옴표 대신 줄의 끝표시(EOL, End of Line. 엔터키값) 문자를 만나게 되었다는 것이다.

여러 문장으로 문자열을 입력할 때는 삼중 따옴표를 사용해야 한다.

함께 보여주고 싶은 문자열이 여러 줄에 걸쳐 있을 때 유용하다.

따옴표를 3개 연달아 적으면서 문자열을 시작하게 되면, 엔터키를 쳤을 때 계속해서 문자열을 입력하는 모드로 인식하기로 파이썬과 약속이 되어 있는 것이다.

실습10) 다음 명령어들을 실행하여 결과를 확인하세요.

>>> Longstring = """I am a student.

I have been enjoying my campus life.

I'm majoring in English Literature.

I'm learning programming language called Python."""

>>> print Longstring

*(결과값을 적어보세요)*

**I am a student.**

**I have been enjoying my campus life.**

**I'm majoring in English Literature.**

**I'm learning programming language called Python.**

**2.7 변수의 자료형 확인하고 변환하기(data type conversion)**

파이썬에는 type()이라는 함수가(function) 있다.

이 함수는 변수의 자료형이 무엇인지를 알려준다.

함수라는 것은 특정한 일을 수행하도록 작성한 코드의 덩어리이다. 함수의 이름 뒤에는 괄호 () 가 붙어 있어 쉽게 구분이 된다. 자주 수행할게 될 일은 미리 함수로 만들어 놓고 필요할 때마다 호출(call)하여 사용한다.

함수의 종류에는 사용자가 직접 정의해서 만든 함수와 파이썬 내부에 미리 만들어져 있는 내장함수(built-in function)가 있다. type() 함수는 파이썬의 내장함수이다.

실습11) 다음 명령어들을 실행하여 결과를 확인하세요.

>>> a = '10'

>>> b = 10

>>> c = 10.0

>>> type(a)

*(결과값을 적어보세요)* ***type ‘str’***

>>> type(b)

*(결과값을 적어보세요)* ***type ‘int’***

>>> type(c)

*(결과값을 적어보세요)* ***type ‘float’***

‘str’은 문자열(string) 유형을 의미하고, ‘int’는 숫자유형 중 정수(integer)를, ‘float’는 실수(real number)를 의미한다.

파이썬은 str(), int(), float() 라는 함수를 내장하고 있다.

이 함수들은 괄호 안에 들어오는 입력값을 각각 문자열, 정수, 실수로 유형을 변환해준다.

실습12) 다음 명령어들을 실행하여 결과를 확인하세요.

>>> a = 10

>>> b = float(a)

>>> a

*(결과값을 적어보세요)* ***10***

>>> b

*(결과값을 적어보세요)* ***10.0***

>>> c = 12.1

>>> d = int(c)

>>> c

*(결과값을 적어보세요)* ***12.1***

>>> d

*(결과값을 적어보세요)* ***12***

>>> e = 12.9

>>> f = int(e)

>>> e

*(결과값을 적어보세요)* ***12.9***

>>> f

*(결과값을 적어보세요)* ***12***

실수로 변환하게 되면 소수점 이하의 값을 갖게 된다.

정수로 변환하게 되면 소수점 이하의 값은 모두 버림이 된다.

참고로, 위 실습에서 보듯이 IDLE 인터랙티브 모드에서만 print 명령어 생략이 가능하다.

괄호의 입력값이 변환 가능한 유형일 때만 정상적으로 작동한다.

실습13) 다음 명령어들을 실행하여 결과를 확인하세요.

>>> print int('23')

*(결과값을 적어보세요)* ***23***

>>> print int('012')

*(결과값을 적어보세요)* ***12***

>>> print int('JinJin')

*(결과값을 적어보세요)*

***Traceback (most recent call last):***

***File "<pyshell#110>", line 1, in <module>***

***print int('JinJin')***

***ValueError: invalid literal for int() with base 10: 'JinJin'***

>>> print float('JinJin')

*(결과값을 적어보세요)*

**Traceback (most recent call last):**

**File "<pyshell#110>", line 1, in <module>**

**print int('JinJin')**

**ValueError: invalid literal for int() with base 10: 'JinJin'**

>>> print str(23)

*(결과값을 적어보세요)* ***23***

>>> print str('23')

*(결과값을 적어보세요)* ***23***

>>> print str('23.01')

*(결과값을 적어보세요)* ***23.01***

>>> print type(str(int('007')))

*(결과값을 적어보세요)* ***<type ‘str’>***