Introduction to Programming in Python

Jonghwa Park
suakii@gmail.com

GYEONGGI SCIENCE HIGH SCHOOL

Variables, Data Type

Object

- Every Variable is an Object

객체

- 프로그램은 실행 중 여러 데이터를 생성한다.
- 파이썬에서는 각각의 데이터는 객체라고 부른다.
- 모든 객체는 형태(Type)를 가지고 있다.
- 형태는 객체가 수행할 수 있는 일들을 결정한다.

객체

- Gshs Student
  - 신입생 생성
  - 신입생 정보가 입력
  - 신입생 반 평성
  - 신입생 교과목 결정
  - 동아리 결정
  - 알앤이 결정
  - 학생은 수업에 참여하고 글을 쓰기도 하고
  - 졸업을 하기도 합니다.

Make Object

Make Object : 숫자

숫자: 그대로 적으면 됩니다.

3

3.14

1+2j

type()을 통해 객체의 종류를 알 수 있습니다.

문자열

- 문자열을 따옴표(“’) 사이에 입력합니다.
- “gshs”
- ‘gshs’

논리값

- True
- False

복잡한 객체 생성

- 이미 로봇을 만들면서 복잡한 객체를 생성해보았습니다.
- from cs1robots import *
- Robot()

- list : [1,2,3,4]
- tuple: (“red”, “yellow”, “green”)
- dictionary: {'name':'pey', 'phone':'0119993323', 'birth': '1118'}
- set: set([1,2,3]) , set(“Hello”)

Type

- 모든 객체는 타입을 가지고 있다.
- 객체의 형태는 객체를 규정짓는다.
- 파이썬에서는
- type()을 이용해 객체가 가지는 형태를 알 수 있다.

Type

- **>>>** type(3)
- <class 'int'>
- **>>>** type(3.1415)
- <class 'float'>
- **>>>** type("gshs")
- <class 'str'>
- **>>>** type(3 + 7j)
- <class 'complex'>
- **>>>** type(True)
- <class 'bool'>

Type

```
from cs1robots import *
load_world('worlds/amazing2.wld')
gshs = Robot("green", beepers=100)
print(type(gshs))
```

```
<class 'cs1robots.Robot'>
```

Name

- 경곽의 모든 학생들은 이름이 있다.
- 객체에는 이름을 부여할 수 있습니다.
- 이름을 부여하는 문장을 대입문(assignment)라고 합니다.

```
from cs1robots import *
load_world('worlds/amazing2.wld')
gshs = Robot("green", beepers=100)

print(type(gshs))
message = "gshs"
n = 17
pi = 3.1415926535897931
finished = True
```

Naming Rules

- 영어 문자, 숫자, 그리고 밑줄 문자(_)로만
- 숫자는 이름의 첫 글자로 올 수 없다.
- 파이썬에서 등록된 키워드(예약어)와 동일한 이름은 불가.

  e.g.) def, if, else, while
- 이름은 대소문자를 구분합니다

Variable: 변수

- 이름이 가리키는 객체는 바뀔 수 있다 없다?
- 따라서 이름은 변수라고 부른다.
- gshs = “gshs”
- gshs = 17
- gshs = 17.0
- 변수에 대입된 객체는 변수가 가지는 값이다.
- 즉 바뀌는 것은 변수의 값이다.
- None은 비어 있는 것을 표시하는데 사용

Member Variable

- 객체는 일을 할 수 있다?
- 경곽 학생은 수업을 들을 수 있고, 말을 할 수 있고
- 잠을 잘수도 있고, 공부를 할 수 도 있고, 게임을 할 수 도 있다.
- 객체가 가지는 이러한 행동(함수)을 멤버함수라고 하며
  .(점) 연산자를 통해 해당 함수를 실행(호출)할 수 있다.

Member Variable

- gshs = Robot()
- gshs.move()
- gshs.turn_left()
- a = “gshs”
- print(a.upper())

여러 이름을 가지는 객체

- 하나의 객체는 여러 이름을 가질 수 있다.
- from cs1robots import *
- load_world('worlds/amazing2.wld')
- gshs = Robot("yellow", beepers=100)
- gs = gshs

- print(gshs)
- print(gs)

여러 이름을 가지는 객체

- /Users/suakii/opt/anaconda3/envs/suakii/bin/python /Users/suakii/PycharmProjects/HuboTest/main.py
- <cs1robots.Robot object at 0x7f84f01a3910>
- <cs1robots.Robot object at 0x7f84f01a3910>
- Close canvas windows to end program.

Operators

산 술연산자(Arithmetic Operators)

Arithmetic Operators

a = 5
b = 2

Operator	Meaning	Example	Result
+	Addition Operator. Adds two Values.	a + b	7
-	Subtraction Operator. Subtracts one value from another	a – b	3
*	Multiplication Operator. Multiples values on either side of the operator	a * b	10
/	Division Operator. Divides left operand by the right operand.	a / b	2.5
%	Modulus Operator. Gives reminder of division	a % b	1
**	Exponent Operator. Calculates exponential power value. a ** b gives the value of a to the power of b	a ** b	25
//	Integer division, also called floor division. Performs division and gives only integer quotient.	a // b	2

식(expression)

- 식은 객체, 변수, 연산자, 함수 호출의 조합으로 이루어짐.
- 3.0 * (2 ** 15 - 12 / 4) + 4 ** 3
- 연산자는 수학에서 사용하는 것처럼 우선순위가 존재
- https://wikidocs.net/20708
- 연산자의 계산 순서가 헷갈릴 때는 괄호를 사용
  예시) $\frac{a}{2\pi}$ 는 a/2*pi 가 아닙니다.
  a/(2*pi)나, a/2/pi 와 같이 써야 합니다.

  모든 연산자는 복소수에서도 사용할 수 있습니다.

문자열 연산

- +
- *
- “gshs” + “ best”
- “gshs”*8

논리식

- 논리식은 계산의 결과가 논리값인 식.
- 이 값은 if, while에서 사용한다.

관계 연산자(Relational Operators)

- > : 크다
- >= : 크거나 같다
- < : 작다
- <= : 작거나 같다
- == : 같다
- != : 같지 않다

논리연산자(Logical Operators)

- and : 논리곱
- or : 논리합
- not : 논리부정

a	b	a and b	a or b	not a
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

Tuple

- tuple 자료형은 데이터의
- 콤마(,)로 데이터를 나열하면 tuple.
- 괄호()안에 ,로 나열해도 tuple입니다.
- 길이 1개짜리 tuple을 만들려면 반드시 뒤에 콤마(,)를
- tuple은 다른 자료형들로 만들 수 있습니다.
- tuple은 tuple을 가질 수 있습니다.

Tuple Test

- position = (1, 2, 3)
- print(type(position))
- print(position.count(1))
- x, y, z = position
- print(x,y)
- x, y = y, x
- print(x, y)

List

- 튜플과 리스트는 앞으로도 많이 사용하게 됩니다.
- 자세한 것은 나중에 다루고 오늘은
- 간단하게 작성하고 사용법만 보고
- 과제를 해결하는데 사용할 것입니다.

List

- 리스트를 사용하면 값들의 모음을 다음과 같이 표현할 수 있다.

- even = [2,4,6,8]
- 리스트는 list 타입의 객체
- 리스트의 각 원소는 위치 값을 사용해서 접근 가능
- 첫 번째 원소는 0번째, 두 번째는 1
- even[0], even[1]
- even.append(10)

list

- >>> a = []
- >>> b = [1, 2, 3]
- >>> c = ['Life', 'is', 'too', 'short']
- >>> d = [1, 2, 'Life', 'is']
- >>> e = [1, 2, ['Life', 'is']]

Function with return

Function

- 함수는 매개변수와 리턴값을 가질 수 있습니다.
- 지금은 리턴값만 가지는 함수를 알아보겠습니다.

Function with Return Value

```
def f():
    n = 0
    for i in range(100):
        n = n + i

    return n


print(f())
```

Python range() Function

Syntax

```
range(start, stop, step)
```

Parameter Values

Parameter	Description
start	Optional. An integer number specifying at which position to start. Default is 0
stop	Required. An integer number specifying at which position to stop (not included).
step	Optional. An integer number specifying the incrementation. Default is 1

https://www.w3schools.com/python/ref_func_range.asp

실습

현재 위치에 비퍼가 몇 개 있는지 출력하는 함수를 작성하고 호출해서 그 결과를 출력하세요.

1. 비퍼 추가(8queens.wld)

```
avenues = 8
streets = 8
walls = []
beepers = {}

beepers = {
    (1, 1): 10,

}
```

실습

```
from cs1robots import *
load_world('worlds/8queens.wld')
gshs = Robot("yellow", beepers=100)

def how_many_beepers():

print(how_many_beepers())
```