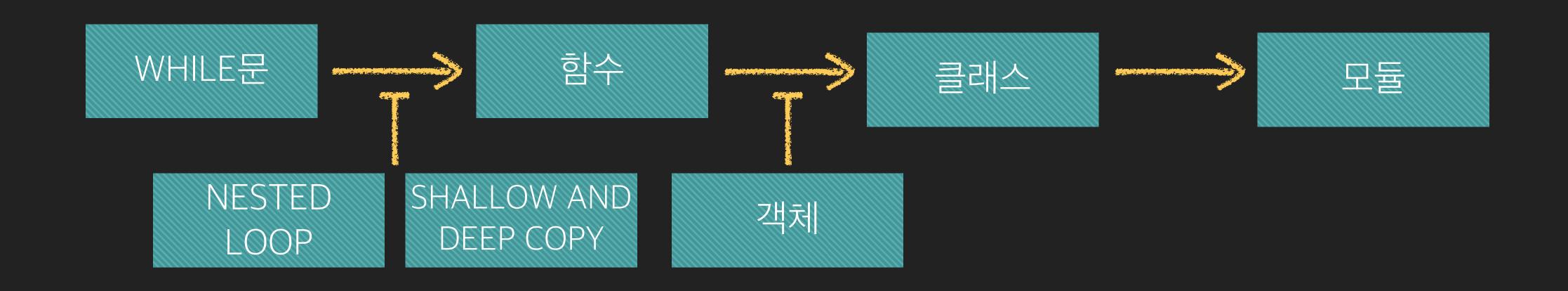
국민대학교 인공지능 연구실 김인경

파이썬기초3주차

3주차 목차



다음 주에는?



INTRO

- ▶ 1주차 colab 실습 링크
 - https://colab.research.google.com/drive/1wgHQKrv2Q6bAjhfKZFItB785nGuEVw-Q
- ▶ 2주차 colab 실습 링크
 - https://colab.research.google.com/drive/1dQ9VYzAuNrHVtF2XOshEztpToIZMF0hz
- ▶ 3주차 colab 실습 링크
 - https://colab.research.google.com/drive/1YdKdD3Y08Bz08_xG1Ext00ywJQCzp1fJ

WHILE 문

- ▶ Conditional loop 특정한 조건이 만족될 때 까지 반복
 - 지난 번 for문은 counting loop
 - ▶ 즉, 정해진 횟수만큼 반복하는 것
- ▶ 조건문이 참인 동안에 while 아래의 문장이 반복해서 수행

while <조건문>: <코드>

NESTED LOOP

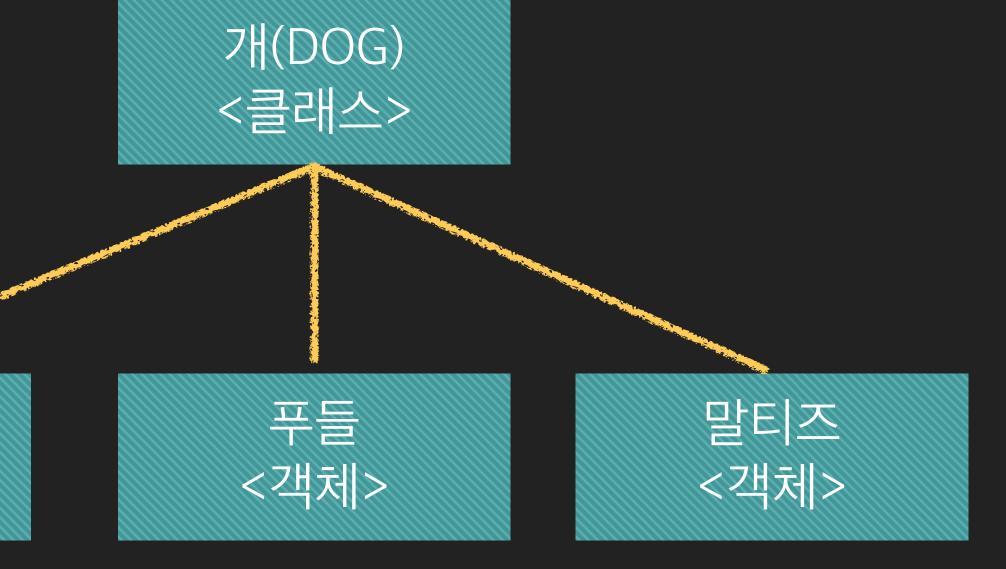
- ▶ for문을 여러개 사용할 수 있지만 2개 이상 사용하는 것을 권장하지 않음
 - ▶ 만약 2개가 넘어간다…? ─> 코드를 잘못 짰을 경우가 높다!
- ▶ 작동 방법
 - ▶ for문의 경우: <시퀀스1>의 첫번째 타겟에서 <시퀀스2>가 모두 반복되면 이후 <시퀀스1>의 두번째 타겟으로 넘어가고 <시퀀스2>를 모두 반복하면서 <시퀀스1>의 타겟을 모두 반복!
 - ▶ while문의 경우: <조건문2>가 만족될 때까지 그 안에 있던.<코드1>을 반복 시행한 후 <조건1>이 만족되면 종료
 - ▶ 실습 자료 참고
- ▶ 예시
 - ▶ 구구단: 1단부터 9단까지 표현할 수 있는 방법은?
 - ▶ 삼각형 출력하기

```
for <타겟1> in <시퀀스1>:
for <타겟2> in <시퀀스2>:
<코드1>
<코드2>
```

```
while <조건문1>:
while <조건문2>:
<코드1>
<코드2>
```

SHALLOW AND DEEP COPY

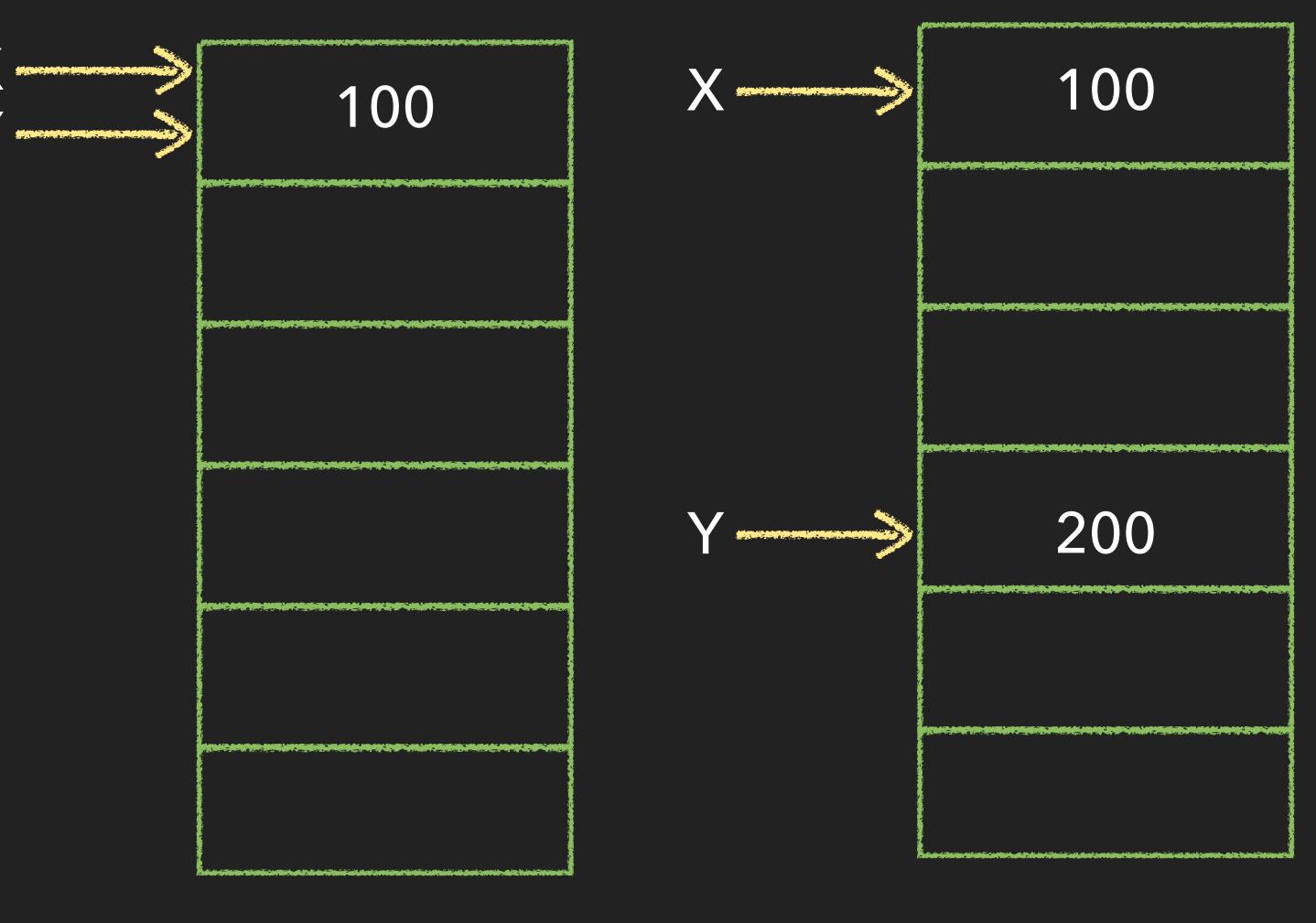
- ▶ 파이썬은 객체지향언어
 - ▶ 컴퓨터 프로그램을 명령어의 목록이 아닌, 여러개의 독립된 단위 즉, 객체들의 모임으로 파악하는 것
 - ▶ 각각의 객체는 메시지를 주고 받고 데이터를 처리할 수 있음
 - ▶ 객체는 주변에 보이는 사물, 생명이라고 생각할 수 있음
 - ▶ 객체의 중요한 속성을 공유하는 객체들의 모임이 클래스
 - ▶ 객체에 대해서는 뒤에서 조금 더 자세히!



리트리버 <객체>

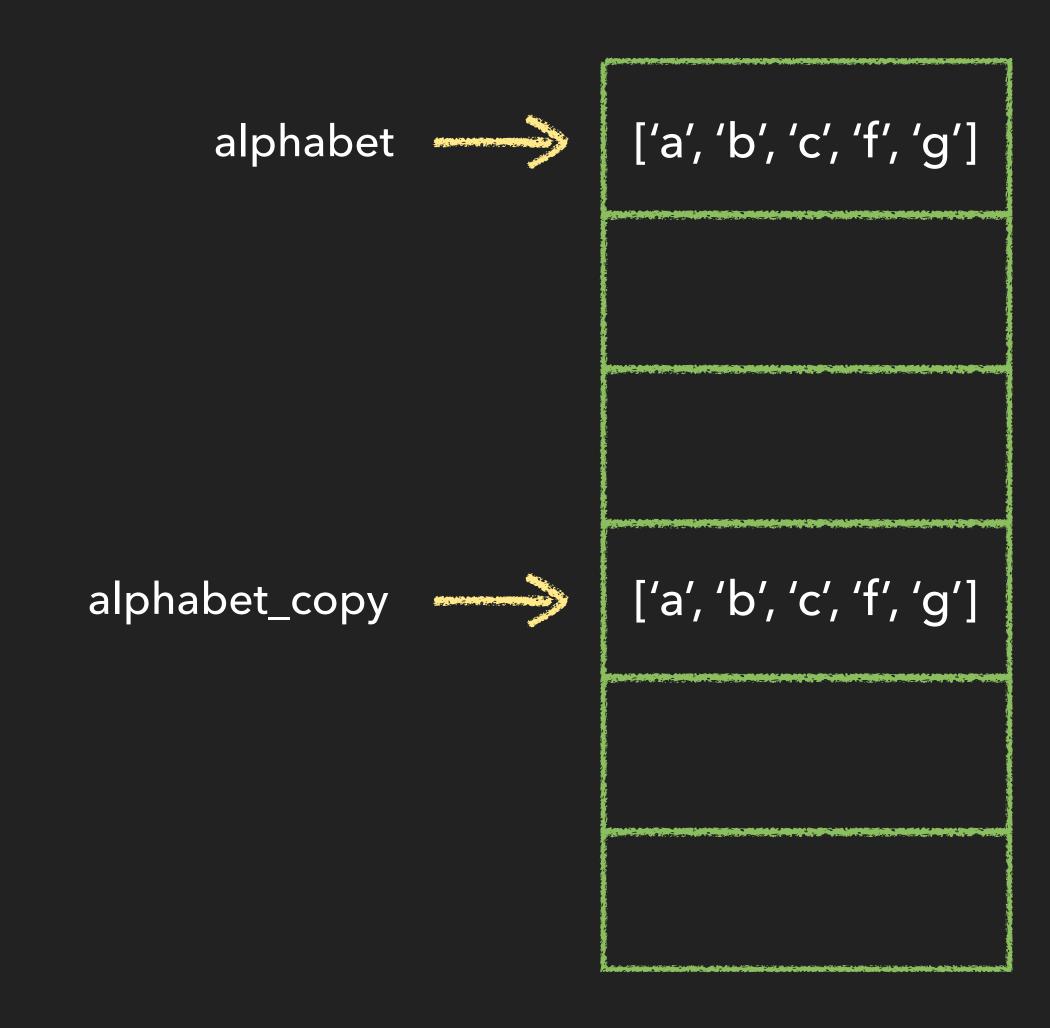
SHALLOW AND DEEP COPY

- ▶ 파이썬은 모든 것을 객체로 보고, 이 객체를 컴퓨터가 인식할 수 있는 identity가 존재함
 - ▶ id()함수를 사용해서 객체의 id를 확인할 수 있음
- ▶ 두 변수안에 같은 값이 있으면 데이 터 자체가 복사되는 것이 아니라 주 소가 복사 되는 것
- ▶ 데이터 자체를 복사해서 새로운 주소 에 넣고 싶다면 깊은 복사를 해야함



SHALLOW AND DEEP COPY

- ▶ 깊은 복사를 하기 위해서는 copy 모듈을 사용
 - import copy
 - ▶ 깊은 복사를 하게 되면 주소를 공유하지 않고 새로운 주소에 같은 데이터를 새롭게 저장함
 - ▶ 때문에 id()를 통해서 출력한 주소값이 다름
- ▶ 리스트 같은 경우는 [:]를 사용해서 전체 복사를 해서 사용할 수 있음



- ▶ 함수는?
 - ▶ 입력값을 가지고 어떤 일을 수행한 다음에 그 결과물을 내어 놓는 것!
 - ▶ ex. Y = 2X + 3 -> X라는 입력 값을 넣었을 때, 어떤 변화에 의해 Y값이 출력되는 것
- ▶ 함수는 왜 사용할까?
 - ▶ 똑같은 내용을 반복해서 작성하는 경우가 발생함
 - ▶ 반복적으로 사용하는 것을 "어떤 입력값이 주어졌을 때, 어떤 결과값을 돌려준다"라는 함수로 작성
- ▶ def키워드를 사용하여 생성하며 호출해서 사용
- ▶ 우리가 자주 사용하는 print, input, len, int, float모두 함수

- > <매개 변수>는 없을 수 있음
- ▶ 리턴 값은 없을 수 있음
- ▶ 가변 매개변수를 사용한 함수

```
def <함수이름> (<매개변수1>, <매개변수2>, ···):
<수행할 문장1>
<수행할 문장2>
```

- ▶ 파이썬에서는 매개변수를 원하는만큼 받을 수 있는 함수를 만들 수 있음
- ▶ 가변 매개변수는 하나만 사용할 수 있으며 가변 매개변수 뒤에는 일반 매개변수가 올 수 없음
 - ▶ 가변 매개변수 함수를 정의할 때는 *를 가변 매개변수 이름 앞에 붙임
- ▶ 파이썬에서는 데이터 타입을 지정해주지 않아도 되기 때문에 매개변수를 5개까지 사용할 것을 권 장 ─> 너무 많으면 가독성 저하

```
def <함수이름> (<매개변수1>, <매개변수2>, ···):
<수행할 문장1>
<수행할 문장2>
```

```
def <함수이름> (<매개변수1>, <매개변수2>, ···, *<가변 매개변수>):
<수행할 문장1>
<수행할 문장2>
```

- ▶ 기본 매개변수
 - ▶ 인자가 전달되지 않을 경우, 값을 정의하는 매개변수
 - ▶ 기본 매개변수 뒤에는 일반 매개변수가 올 수 없음
- ▶ 함수 호출(call)
 - ▶ 함수는 호출될 때, 독자적 메모리 공간이 할당됨
 - ▶ 함수가 종료될 때, 할당되었던 메모리 공간은 소멸됨 ─> 메모리 공간을 사용하고 있던 모든 변수 및, 인자도 동시에 소멸됨
 - ▶ 따라서 함수가 끝나고 나서도 해당 값을 유지하고 싶다면 리턴해주어야 함

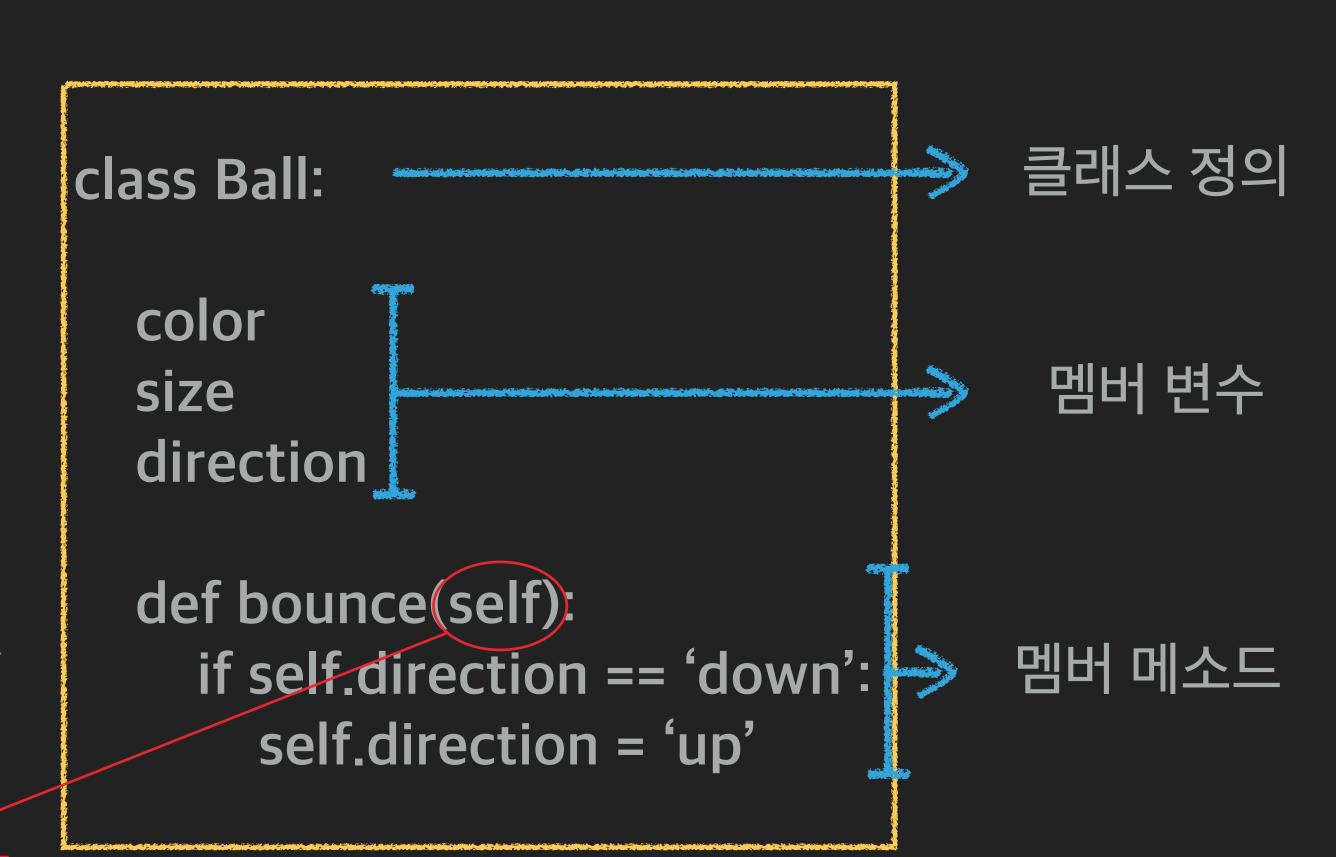
- ▶ 인자(argument)넘기기
 - 함수 내에서 변경된 인자는 함수가 끝나고 반영되지 않음
 - ▶ Mutable data인 list, dictionary, set는 예외
 - ▶ 구조화된 데이터는 용량이 클 것이라고 예상하기 때문에 주소값이 저장되어 있음
 - ▶ 바뀐 데이터들이 그대로 저장되어서 바뀐 것으로 나타남
 - ▶ 함수 내에서 변경한 인자를 함수가 끝나고 반영하고 싶으면 return을 사용
- ▶ 함수도 함수의 인자가 될 수 있음

- ▶ lambda함수
 - ▶ def와 동일한 역할을 수행하지만 보통 함수를 한줄로 간결하게 만들 때 사용함
 - ▶ 사용법: <lambda 매개변수1, 매개변수2, ··· : 매개변수를 이용한 표현식>
 - ex. get_sum = lambda x, y : x + y
- ▶ 파이썬 내장 함수
 - ▶ abs(x) : 절대값 반환
 - ▶ min(x), max(x): 최소값, 최대값 반환
 - ▶ int(x), float(x): 문자열 혹은 실수를 정수로 반환, 문자열 혹은 정수를 실수로 반환
 - ▶ enumerate(x): 시퀀스 객체를 입력받아, enumerate객체로 반환. enumerate객체는 첫번째 요소로 번호, 두번째 요소로 번 호에 해당하는 값을 가짐

- ▶ 객체란?
 - ▶ 어떤 대상에 대해 설명 가능한 속성과 그 대상을 가지고 할 수 있는 행동을 모아놓은 것
 - ▶ 객체 = 속성 + 메소드
 - ▶ 속성 = 대상에 대한 특징 혹은 대상에 대해 알고 있는 정보
 - ▶ 메소드 = 대상의 행동 또는 대상을 가지고 할 수 있는 일
- ▶ 클래스란?
 - ▶ 추상적이지만 어떤 요소들에 대한 공통적이고 일반적인 '정의'를 할 수 있는 개념
 - ▶ 객체의 속성을 저장하기 위한 변수들과 객체의 행동을 수행하기 위한 메소드들이 같은 이름 공간에 묶인 집합 체

- ▶ 클래스를 비유하자면
 - ▶ 과자 틀 ─> 클래스
 - ▶ 과자 틀에 의해 만들어진 과자 ─> 객체
- ▶ 클래스는 크게 클래스 정의, 멤버 변수, 멤 버 메소드 3가지 부분으로 구성됨
 - 에 비 변수와 메소드는 반드시 필요한 것은 아님

self는 파이썬만의 독특한 변수로 클래스 내에서 정의되는 함수는 무조건 첫 번째 인자로 self를 사용해야 함



- ▶ 인스턴스(Instance)
 - ▶ 클래스를 사용하여 만든 실제 객체
 - ▶ 클래스는 인스턴스를 만드는 하나의 틀
 - ▶ 인스턴스는 생성이 완료된 직후 원본 클래스와 동일한 데이터와 함수를 가짐
- ▶ 인스턴스를 생성한 후 인스턴스의 데이터를 변경하고, 메소드를 호출하고, 멤버 변수를 추가하거나 삭제할 수 있음
- ▶ 인스턴스를 통해서 자신을 생성한 클래스를 참조할 수 있음
 - ▶ 내장 속성 '_class_'를 사용

- self
 - ▶ 현재의 인스턴스 객체를 가르키는 기능을 하는 지시어
 - ▶ 즉, 메소드가 어떤 인스턴스에서 호출되었는지 알려주는 인스턴스 참조자
- ▶ 생성자 메소드
 - ▶ 인스턴스가 생성될 때 자동으로 호출
 - ▶ 인스턴스가 생성되었을 때 초기화 작업을 하기 위함
 - ▶ 특수 메소드 '_init_()'으로 정의
 - ▶ 객체를 생성하면서 속성값을 설정하기 위한 방법
 - ▶ 생성자를 통해 인스턴스 생성 시 초기화할 멤버 변수 값을 인자로 전달

- ▶ 인스턴스 초기화
 - ▶ 객체를 생성하면서 속성값을 설정하기 위한 방법
 - ▶ 생성자를 통해 인스턴스 생성 시, 초기화 할 멤버 변수 값을 인자로 전달
- ▶ 소멸자 메소드
 - ▶ 인스턴스의 레퍼런스 카운트가 0이 될때 자동으로 호출
 - ▶ 특수 메소드 '__del__()'로 정의
 - ▶ 파이썬의 경우 자동으로 실행되기 때문에 필요할 때만 사용
- ▶ 객체 출력 메소드
 - ▶ 객체를 출력할 때 무엇을 보여줄지 알려줌
 - ▶ 특수 메소드 '_str_()'로 정의

- ▶ 다형성(Polymorphism)
 - ▶ 이름이 같은 메소드를 서로 다른 클래스마다 가질 수 있는 것
 - > 동일한 메소드에 대하여 구체적인 인스턴스마다 다른 동작을 수행하는 특징
- > 상속(Inheritance)
 - ▶ 클래스를 정의할 때, 다른 클래스에 있는 공통된 속성 및 메소드를 매번 처음부터 만들지 않고 사용하는 것
 - ▶ 기존 클래스가 라이브러리 형태로 제공되거나 수정이 허용되지 않는 경우도 있기 때문에 상속을 사용함

모듈1

- ▶ 모듈이란?
 - ▶ 파이썬 함수와 객체들의 묶음
 - ▶ 즉, 다른 파이썬 프로그램에서 불러와 사용할 수 있게끔 만든 파이썬 파일
 - ▶ 파이썬 언어는 프로그래밍에서 공통적으로 많이 쓰일 수 있는 모듈을 빌트인 모듈 형태로 가지고 있음
- ▶ 모듈 임포트
 - ▶ import문을 사용하여 특정 모듈을 프로그램에 포함
 - ▶ 사용 방법 예제
 - import math
 - ▶ from math import sin 모듈의 특정 함수/ 변수 가져오기
 - ▶ dir(<모듈이름>)을 사용하면 모듈에 포함된 함수/ 변수를 확인할 수 있음

참고자료

- https://wikidocs.net/28
- http://pythonstudy.xyz/python/article/16-%ED%95%A8%EC%88%98