데이터처리프로그래밍

Object Oriented programming
(Inheritance)

강원대학교 교육혁신원 송혜정

<hjsong@kangwon.ac.kr>

Inheritance

✓ 학습목표
- Inheritance(상속)을 이해한다.

✓ 학습내용
- Inheritance (상속)
- Overriding (메소드 재정의)
- Polymorphism (다형성)

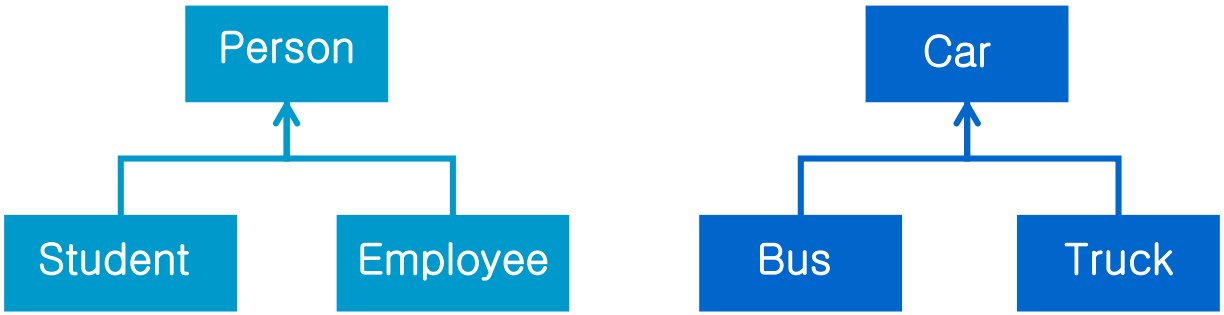
강의에 앞서서..

- 본 강의자료는 아래의 자료들을 참고하여 만들어 졌음을 알립니다

1. 데이터과학을 위한 파이썬 프로그래밍, 최성철, 한빛아카데미,2019
2. Python (https://docs.python.org)
3. 실용 파이썬 프로그래밍(Practical Python Programming) (https://wikidocs.net/84360)
4. 점프 투 파이썬 (https://wikidocs.net/book/1)

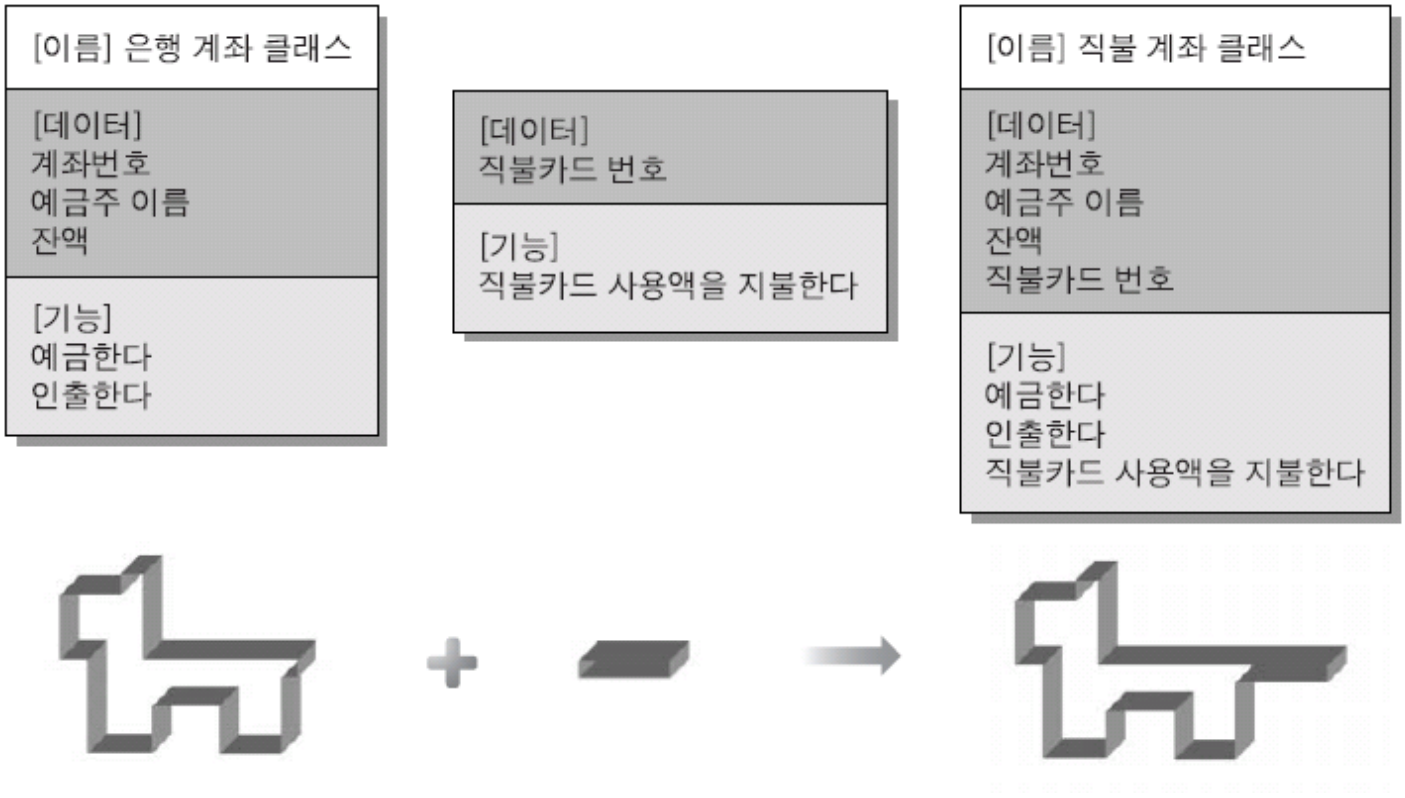
Inheritance

- Inheritance(상속)
  - 이미 정의된 클래스의 구성요소들을 계승 받아 새로운 클래스로 확장하는 것
  - 재사용성, 추상화 구현
  - 상속관계는 클래스들의 계층구조로 표현
    - 상위 클래스 (Super class)
      - 상속되는 클래스, 일반적인 특성
    - 하위 클래스 (Sub class, Derived class, Extended class)
      - 상속받는 클래스, 구체적인 특성
      - 상위 클래스의 구성요소 + 하위 클래스의 추가 기능
      - 상위클래스를 확장



Inheritance

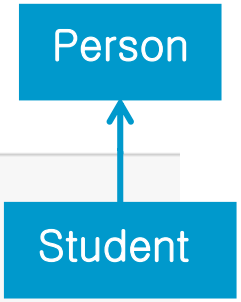
- 은행계좌 상속 예



기존 클래스 확장기능 추가 확장 클래스

Inheritance

- 상속 예
  - 상속관련 클래스 정의




```
class Person(object):                    # 부모 클래스 Person 선언
    def __init__(self, name, age):
        self.name = name                 #부모 클래스 속성 선언
        self.age = age

    def print1(self):                    #부모 클래스 메서드 선언
        print("이름:", self.name, " 나이:", str(self.age))
```

```
class Student(Person):  #부모 클래스 Person 으로부터 상속
    def __init__(self, name, age, dept):
        self.name = name                 #부모 클래스 속성 사용
        self.age = age                   #부모 클래스 속성 사용
        self.dept = dept                 #자식 클래스 속성 추가
    def print2(self):                    # 자식 클래스 메서드 추가
        self.print1()                    #부모 메서드 사용
        print("성명:", self.name, " 학과:", self.dept) #부모속성, 자식 속성 사용
```

Inheritance

- 상속 예
  - 상속 관련 객체 생성 및 멤버 접근

```
p = Person("홍길동", 25) # Person 객체 생성
p.print1()  #p1 객체의 print1() 함수(메서드) 호출
```

```
이름: 홍길동  나이: 25
```

```
s = Student("이순신", 22, "컴공")  # Student 객체 생성
print("s.print1() = ")
s.print1()   #Student 객체로 부모 객체 함수 호출
print("s.print2() = ")
s.print2()   #Student 객체함수 호출
```

```
s.print1() =
이름: 이순신  나이: 22
s.print2() =
이름: 이순신  나이: 22
성명: 이순신  학과: 컴공
```

Inheritance

- 메소드 재정의 (overriding)
  - 상속관계에서 상위 클래스 메소드와 동일한 이름으로 하위 클래스에서 재정의
  - 기존 클래스의 메서드 구현 부분을 수정하여 새로운 클래스로 확장
  - 다형성(polymorphism)을 구현

- super()사용

  (1)부모 객체 접근 시 사용

  (2)초기화 함수 호출 시 사용

  (3)메소드 재정의시 동일한 이름의 메소드 호출 시 사용

Inheritance

- 메소드 재정의 예

```
class Person(object):                          # 부모 클래스 Person 선언
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def about_me(self):                        # 메서드 선언
        print("이름:", self.name, " 나이:", str(self.age))
```

```
class Student(Person):  #부모 클래스 Person 으로부터 상속
    def __init__(self, name, age, dept):
        super().__init__(name, age)   #부모 객체(super) 초기화함수 호출
        self.dept = dept              #속성 추가
    def about_me(self):               # 부모클래스 메서드 재정의
        super().about_me()            #부모 객체(super) 메서드 호출
        print("학과:", self.dept) #Student(학생) 클래스 관련 내용 추가
```

```
p = Person("홍길동", 25) # Person 객체 생성
p.about_me() #p1 객체의 about_me()함수(메서드) 호출
```

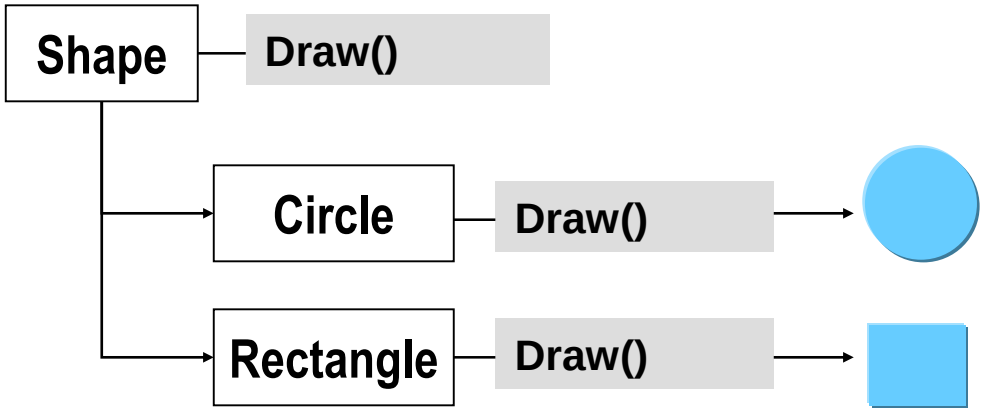
```
이름: 홍길동  나이: 25
```

```
s = Student("이순신", 22, "컴공")  # Student 객체 생성
s.about_me()   #Student 객체함수 호출
```

```
이름: 이순신  나이: 22
학과: 컴공
```

Inheritance

- 다형성(Polymorphism)
  - “one interface, multiple implementation”
  - 하나의 명령으로 다양한 결과를 얻을 수 있는 특징
  - 상속관계의 메소드 재정의로 구현



Inheritance

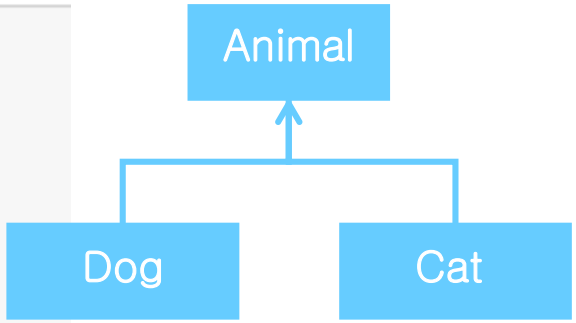
- 다형성 예1

```
class Animal:
    def __init__(self, name):
        self.name = name
    def talk(self):
        return '??'

class Cat(Animal):
    def talk(self):
        return '야옹!'

class Dog(Animal):
    def talk(self):
        return '멍! 멍!'

animals = [Cat('나비'), Cat('냥이'), Dog('바둑이')]
for animal in animals:
    print(animal.name + ': ' + animal.talk())
```




```
나비: 야옹!
냥이: 야옹!
바둑이: 멍! 멍!
```

Inheritance

- 다형성 예제
  - 이름과 좌표값을 가지는 도형 (Shape)클래스로부터 상속받은 Circle, Rectangle 클래스를 정의하여 각 도형의 면적을 출력하는 문제

```
class Shape:
    def __init__(self, x, y):
        self.name = "Shape"     #도형이름
        self.x = x              #x좌표
        self.y = y              #y좌표

    def draw(self):
        print(self.name, "(" + str(self.x) + "," + str(self.y) + ")")

class Circle(Shape):
    def __init__(self, x, y, r):
        super().__init__(x, y)
        self.name = "Circle"    #도형이름
        self.r = r

    def getArea(self):
        a = self.r * self.r * 3.14
        return  a

    def draw(self):
        super().draw()
        print("radius = ", str(self.r))
        a = self.getArea()
        print("area = {0:.2f}".format(a))
```

Inheritance

- 다형성 예제

```
class Rectangle(Shape):
    def __init__(self, x, y, w, h):
        super().__init__(x, y)
        self.name = "Rectangle"
        self.w = w
        self.h = h

    def getArea(self):
        a = self.w * self.h
        return a

    def draw(self):
        super().draw()
        print("width = ", str(self.w), "height = ", str(self.h))
        a = self.getArea()
        print("area = {0:.2f}".format(a))
```

```
shapes = [Shape(30,30), Circle(10,20,20.5), Rectangle(5,3,100,70), Circle(50,40,50.5)]
for s in shapes:
    s.draw()
```

```
Shape (30,30)
Circle (10,20)
radius =  20.5
area = 1319.59
Rectangle (5,3)
width =  100 height =  70
area = 7000.00
Circle (50,40)
radius =  50.5
area = 8007.79
```

QUIZ!

Report 10. 상속연습

1)Person클래스로부터 상속받은 Employee 클래스 선언

(1)Employee 클래스에 급여, 입사일 속성 추가

(2)about_me() 메소드 재정의

2)Student, Employee 객체를 리스트에 생성하여 about_me() 출력 (다형성 확인)

- 학습활동 결과는 e-루리에 제출바랍니다.
- 제목 : Report10. 상속연습
- 제출내용 : Report10_inhe_성명. ipynb 파일을 제출
- 제출기한 : 2021년 6월 1일 오후 11:58