(13주차 실습) – 평가과제#1

12장(객체지향프로그래밍) 실습

2019. 11. 26.

오늘의 실습 (12장 실습)

- 다음을 Python으로 코딩하고, 소스코드와 실행결과의 capture 화면을 제출하라.
  - 12장
    1. 383 쪽 SELF STUDY 12-1
    2. 385 쪽 code-12-03.py 기본생성자 실습
    3. 386 쪽 code-12-04.py 매개변수가 있는 생성자 실습
    4. 387 쪽 code-12-05.py 클래스,인스턴스, 필드, 메서드, 생성자 실습 프로그램
    5. 390~91 쪽 code-12-06.py 클래스변수와 인스턴스변수 실습
    6. 394~95 쪽 code-12-07.py 상속에서의 메서드 오버라이딩 실습
    7. 395 쪽 SELF STUDY 12-2
    8. 396~97 쪽 code-12-08.py 상속 실습 프로그램

- 제출 : 12. 2.(월) 자정까지

- 제출물 : 1개의 아래아한글 또는 pdf 파일로된 보고서로 제출
  - 소스코드는 실행해 볼 수 있도록 표 안에 붙여넣기로 제출
  - 실행결과는 소스코드와 실행결과를 볼 수 있게 화면을 capture하여 제출

1. SELF STUDY 12-1 (383쪽)

- Code12-02.py의 upSpeed() 함수를 수정해서 속도가 150이 넘으면 최대 150으로 조절해 보자. 예를 들어 upSpeed(200)을 사용해도 출력인 150km가 되어야 한다.

Code12-02.py

```
## 클래스 선언 부분 ##
class  Car :
    color = ""
    speed = 0

    def  upSpeed(self, value) :
        self.speed += value

    def  downSpeed(self, value) :
        self.speed -= value

## 메인 코드 부분 ##
myCar1 = Car()
myCar1.color = "빨강"
myCar1.speed = 0

myCar2 = Car()
myCar2.color = "파랑"
myCar2.speed = 0


myCar3 = Car()
myCar3.color = "노랑"
myCar3.speed = 0

myCar1.upSpeed(30)
print("자동차1의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar1.color,
myCar1.speed))

myCar2.upSpeed(60)
print("자동차2의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar2.color,
myCar2.speed))

myCar3.upSpeed(0)
print("자동차3의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar3.color,
myCar3.speed))
```

2. 기본생성자실습 (385쪽)

- Code12-03.py의 기본생성자 예제를 실행시켜 보라.

```
## 클래스 선언 부분 ##
class  Car :
    color = ""
    speed = 0

    def __init__(self) :
        self.color = "빨강"
        self.speed = 0

    def  upSpeed(self, value) :
        self.speed += value

    def  downSpeed(self, value) :
        self.speed -= value

## 메인 코드 부분 ##
myCar1 = Car()
myCar2 = Car()

print("자동차1의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar1.color, myCar1.speed))
print("자동차2의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar2.color, myCar2.speed))
```

3. 매개변수가 있는 생성자 실습(386쪽)

- code-12-04.py를 실행시켜보라.

```
## 클래스 정의 부분 ##
class  Car :
    color = ""
    speed = 0

    def __init__(self, value1, value2) :
        self.color = value1
        self.speed = value2

    def  upSpeed(self, value) :
        self.speed += value

    def  downSpeed(self, value) :
        self.speed -= value

## 메인 코드 부분 ##
myCar1 = Car("빨강", 30)
myCar2 = Car("파랑", 60)

print("자동차1의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar1.color, myCar1.speed))

print("자동차2의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar2.color, myCar2.speed))
```

4. 클래스,인스턴스, 필드, 메서드, 생성자 실습 프로그램 (386쪽)

- code-12-05.py를 실행시켜보라.

```
## 클래스 정의 부분 ##
class  Car :
    name = ""
    speed = 0

    def __init__(self, name, speed):
        self.name = name
        self.speed = speed

    def  getName(self) :
        return self.name

    def getSpeed(self) :
        return self.speed

## 전역  변수 선언 부분 ##
car1, car2 = None, None

## 메인 코드 부분 ##
car1 = Car("아우디", 0)
car2 = Car("벤츠", 30)

print("%s의 현재 속도는  %d입니다." % (car1.getName(), car1.getSpeed()))
print("%s의 현재 속도는  %d입니다." % (car2.getName(), car2.getSpeed()))
```

5. 클래스변수와 인스턴스변수 실습(390쪽)

- code-12-06.py 를 실행시켜보라.

```
## 클래스 선언 부분 ##
class  Car :
    color = ""    # 인스턴스 변수
    speed = 0    # 인스턴스 변수
    count = 0    # 클래스 변수

    def printMessage() :
      print("시험 출력입니다.")

    def __init__(self):
        self.speed = 0
        Car.count += 1

# 변수 선언
myCar1, myCar2 = None, None

# 메인 코드 부분
myCar1 = Car()
myCar1.speed = 30
print("자동차1의 현재 속도는 %dkm, 생산된 자동차는 총 %d대입니다." %(myCar1.speed, Car.count))

myCar2 = Car()
myCar2.speed = 60
print("자동차2의 현재 속도는 %dkm, 생산된 자동차는 총 %d대입니다." %(myCar2.speed, myCar2.count))
```

5. 클래스변수와 인스턴스변수 실습(390쪽)

- code-12-06.py 를 실행시켜보라.

```
## 클래스 선언 부분 ##
class  Car :
    color = ""    # 인스턴스 변수
    speed = 0    # 인스턴스 변수
    count = 0    # 클래스 변수

    def printMessage() :
      print("시험 출력입니다.")

    def __init__(self):
        self.speed = 0
        Car.count += 1

# 변수 선언
myCar1, myCar2 = None, None

# 메인 코드 부분
myCar1 = Car()
myCar1.speed = 30
print("자동차1의 현재 속도는 %dkm, 생산된 자동차는 총 %d대입니다." %(myCar1.speed, Car.count))

myCar2 = Car()
myCar2.speed = 60
print("자동차2의 현재 속도는 %dkm, 생산된 자동차는 총 %d대입니다." %(myCar2.speed, myCar2.count))
```

6. 상속에서의 메서드 오버라이딩 실습(394쪽)

- code-12-07.py 를 실행시켜보라.

```
## 클래스 선언 부분 ##
class  Car :
    speed = 0
    def upSpeed(self, value):
        self.speed += value

        print("현재 속도(슈퍼 클래스) : %d" % self.speed)

class Sedan(Car) :
    def upSpeed(self, value):
        self.speed += value

        if self.speed > 150 :
            self.speed = 150

        print("현재 속도(서브 클래스) : %d" % self.speed)
```

```
class Truck(Car) :
    pass

## 변수 선언 부분 ##
sedan1, truck1 = None, None

## 메인 코드 부분 ##
truck1 = Truck()
sedan1 = Sedan()

print("트럭 --> ", end = "")
truck1.upSpeed(200)

print("승용차 --> ", end = "")
sedan1.upSpeed(200)
```

7. SELF STUDY 12-02 (395쪽)

- code-12-07.py에서 Sonata클래스를 추가해보자. 단, Sonata클래스는 Car→Sedan→Sonata 순서로 상속을 받도록 하자. Sonata클래스에서 특별히 추가하는 필드나 메서드가 없다. (붉은색 변수 선언 부분 이하의 코드는 수정되었으므로 그대로 이용할 것)

code-12-07.py

```
## 클래스 선언 부분 ##
class  Car :
    speed = 0
    def upSpeed(self, value):
        self.speed += value

        print("현재 속도(슈퍼 클래스) : %d" % self.speed)

class Sedan(Car) :
    def upSpeed(self, value):
        self.speed += value

        if self.speed > 150 :
            self.speed = 150

        print("현재 속도(서브 클래스) : %d" % self.speed)

class Truck(Car) :
    pass
```

```
## 변수 선언 부분 ##
sedan1, truck1, sonata1 = None, None, None

## 메인 코드 부분 ##
truck1 = Truck()
sedan1 = Sedan()
sonata1 = Sonata()

print("트럭 --> ", end = "")
truck1.upSpeed(200)

print("승용차 --> ", end = "")
sedan1.upSpeed(150)

print("소나타 --> ", end = "")
sonata1.upSpeed(150)
```

출력

```
트럭 --> 현재 속도(슈퍼 클래스) : 200
승용차 --> 현재 속도(서브 클래스) : 150
소나타 --> 현재 속도(서브 클래스) : 150
>>>
```

8. 상속 실습 프로그램(396~97쪽)

code-12-08.py를 실행시켜보라

```
import turtle
import random

## 클래스 선언 부분 ##
class Shape:  # 부모 클래스
    myTurtle = None
    cx, cy = 0, 0  # 사각형 및 원의 중심점.

    def __init__(self):
        self.myTurtle = turtle.Turtle('turtle') # 거북이 생성.

    def setPen(self) : # 펜 색상과 두께를 랜덤하게 뽑기
        r = random.random()
        g = random.random()
        b = random.random()
        self.myTurtle.pencolor((r, g, b))
        pSize = random.randrange(1,10)
        self.myTurtle.pensize(pSize)

    def drawShape(self):  # 하위 클래스에서 상속받아서 오버라이딩
        pass

class Rectangle(Shape): # 자식 클래스
    width, height = [0] * 2
    def __init__(self, x, y):
        Shape.__init__(self)
        self.cx = x
        self.cy = y
        self.width = random.randrange(20, 100)
        self.height = random.randrange(20, 100)

    def drawShape(self) :
        # 네모 그리기
        sx1, sy1, sx2, sy2 = [0] * 4  # 왼쪽 위 X, Y와 오른쪽 아래 X,Y

        sx1 = self.cx - self.width/2
        sy1 = self.cy - self.height /2
        sx2 = self.cx + self.width/2
        sy2 = self.cy + self.height /2

        self.setPen() # 부모 클래스 메서드
        self.myTurtle.penup()
        self.myTurtle.goto(sx1, sy1)
        self.myTurtle.pendown()
        self.myTurtle.goto(sx1, sy2)
        self.myTurtle.goto(sx2, sy2)
        self.myTurtle.goto(sx2, sy1)
        self.myTurtle.goto(sx1, sy1)

## 함수 선언 부분 ##
def  screenLeftClick(x,y):
    rect = Rectangle(x, y)
    rect.drawShape()

## 메인 코드 부분 ##
turtle.title('거북이로 객체지향 사각형  그리기')
turtle.onscreenclick(screenLeftClick,1)
turtle.done()
```