



지능로봇공학과

INTELLIGENT ROBOT

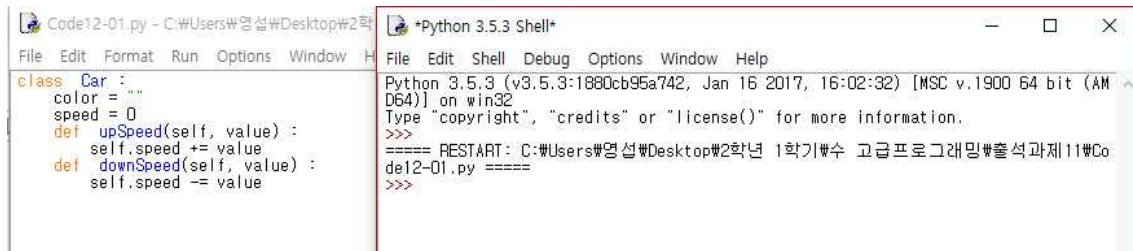
담당 교수	임종관 교수님
학과	지능로봇학과
학번	1558021
이름	이영섭

목 차

code12-01	3P
code12-02	4P
code12-03	5P
code12-04	6P
code12-05	7P
code12-06	8P
code12-07	9P~10P
code12-08	11P
code12-09	12P
code12-10	13P
code12-11	14P
code12-12	15P
code12-13	16P
응용예제1	17P~18P
응용예제2	19P~21P
참고문헌	22P

※code12-01

```
class Car : #car클래스
    color = "" #문자형변수선언
    speed = 0 #정수형변수선언
    def upSpeed(self, value) : #upspeed선언
        self.speed += value #speed를 불러와 value값을 더한다
    def downSpeed(self, value) : #downspeed 선언
        self.speed -= value #speed를 불러와 value값을 뺀다
```



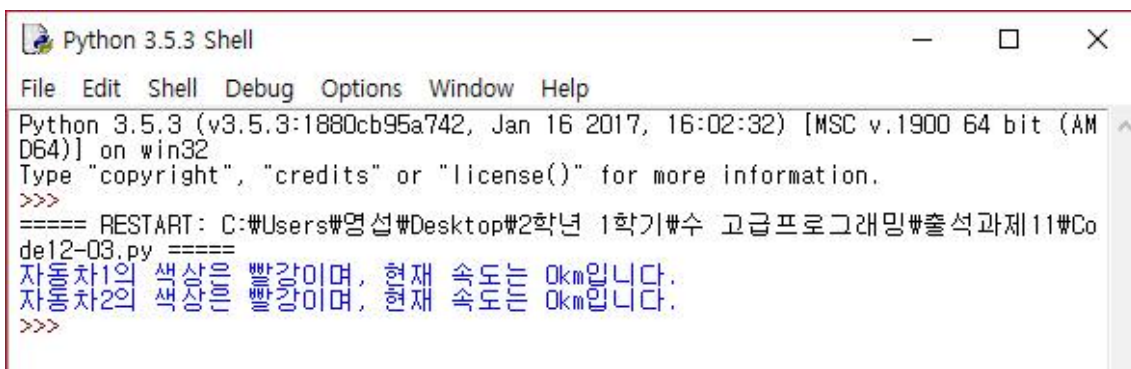
※code12-02

```
class Car :#car클래스
    color = "" #문자형 변수 선언
    speed = 0 #정수형 변수 선언
    def upSpeed(self, value) : #upspeed 선언
        self.speed += value #speed를 불러와 value값을 더한다
    def downSpeed(self, value) : #downspeed선언
        self.speed -= value #speed를 불러와 value값을 뺀다
myCar1 = Car() #Car클래스 실행
myCar1.color = "빨강" # myCar1의 color지정
myCar1.speed = 0 # myCar1의 speed지정
myCar2 = Car() #Car클래스 실행
myCar2.color = "파랑" # myCar2의 color지정
myCar2.speed = 0# myCar2의 speed지정
myCar3 = Car() #Car클래스 실행
myCar3.color = "노랑" # myCar3의 color지정
myCar3.speed = 0# myCar3의 speed지정
myCar1.upSpeed(30) #myCar1의 클래스에서 upSpeed실행
print("자동차1의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar1.color,
myCar1.speed)) #출력
myCar2.upSpeed(60) #myCar2의 클래스에서 upSpeed실행
print("자동차2의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar2.color,
myCar2.speed)) #출력
myCar3.upSpeed(0) #myCar3의 클래스에서 upSpeed실행
print("자동차3의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar3.color,
myCar3.speed))#출력
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\명섭\Desktop\2학년 1학기\수업 고급프로그래밍\출석과제11\Code12-02.py =====
자동차1의 색상은 빨강이며, 현재 속도는 30km입니다.
자동차2의 색상은 파랑이며, 현재 속도는 60km입니다.
자동차3의 색상은 노랑이며, 현재 속도는 0km입니다.
>>> |
```

※code12-03

```
class Car :#Car클래스
    color = "" #문자형 변수선언
    speed = 0 #정수형 변수 선언
    def __init__(self) : #__init__함수 선언 항상실행
        self.color = "빨강" #color에 문자 저장
        self.speed = 0 #speed에 정수 저장
    def upSpeed(self, value) : #upSpeed선언
        self.speed += value #Speed에 value값을 더한다
    def downSpeed(self, value) :#downSpeed 선언
        self.speed -= value#Speed에 value값을 뺀다
myCar1 = Car() #Car클래스실행
myCar2 = Car() #Car클래스 실행
print("자동차1의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar1.color,
myCar1.speed)) #출력
print("자동차2의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar2.color,
myCar2.speed))#출력
```



```
Python 3.5.3 Shell
File Edit Shell Debug Options Window Help
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\영섭\Desktop\2학년 1학기\수업 고급프로그래밍\출석과제11\Code12-03.py =====
자동차1의 색상은 빨강이며, 현재 속도는 0km입니다.
자동차2의 색상은 빨강이며, 현재 속도는 0km입니다.
>>>
```

※code12-04

```
class Car :#Car 클래스
    color = "" #문자형 변수 선언
    speed = 0 #정수형 변수 선언
    def __init__(self, value1, value2) : #__init__선언 항상실행
        self.color = value1 #color값을 value1값으로 지정
        self.speed = value2#speed값을 value2값으로 지정
    def upSpeed(self, value) : #upSpeed선언
        self.speed += value #speed에 value값을 더한다
    def downSpeed(self, value) :#downSpeed선언
        self.speed -= value#speed에 value값을 뺀다
myCar1 = Car("빨강", 30) #Car클래스 실행
myCar2 = Car("파랑", 60)#Car클래스 실행
print("자동차1의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar1.color,
myCar1.speed)) #출력
print("자동차2의 색상은 %s이며, 현재 속도는 %dkm입니다." % (myCar2.color,
myCar2.speed))#출력
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\명섭\Desktop\2학년 1학기\수 고급프로그래밍\출석과제11\Code12-04.py =====
자동차1의 색상은 빨강이며, 현재 속도는 30km입니다.
자동차2의 색상은 파랑이며, 현재 속도는 60km입니다.
>>>
```

※code12-05

```
class Car : #Car 클래스
    name = "" #문자형 변수 선언
    speed = 0 #정수형 변수 선언
    def __init__(self, name, speed): #__init__선언 항상실행
        self.name = name #name에 name저장
        self.speed = speed #speed에 speed저장
    def getName(self) : #getName선언
        return self.name #name값 return
    def getSpeed(self) : #getSpeed선언
        return self.speed #speed값 return
car1, car2 = None, None #변수 선언
car1 = Car("아우디", 0) #car클래스 실행
car2 = Car("벤츠", 30)#car클래스 실행
print("%s의 현재 속도는 %d입니다." % (car1.getName(), car1.getSpeed())) #출력
print("%s의 현재 속도는 %d입니다." % (car2.getName(), car2.getSpeed())) #출력
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\영섭\Desktop\2학년 1학기\수업 고급프로그래밍\출석과제11\Code12-05.py =====
아우디의 현재 속도는 0입니다.
벤츠의 현재 속도는 30입니다.
>>> |
```

※code12-06

```
class Car : #Car클래스
    color = "" #문자형 변수 선언
    speed = 0 #정수형 변수 선언
    count = 0 #정수형 변수 선언
    def printMessage() : #printMessage선언
        print("시험 출력입니다.") #출력
    def __init__(self): #__init__선언 항상 실행
        self.speed = 0 #speed값 지정
        Car.count += 1 #Car클래스의 count값 +1
myCar1, myCar2 = None, None #변수선언
myCar1 = Car() #Car클래스 실행
myCar1.speed = 30 #myCar1의 speed에 30값 저장
print("자동차1의 현재 속도는 %dkm, 생산된 자동차는 총 %d대입니다."
      %(myCar1.speed, Car.count))#출력
myCar2 = Car() #Car클래스 실행
myCar2.speed = 60 #myCar2의 speed에 60값 저장
print("자동차2의 현재 속도는 %dkm, 생산된 자동차는 총 %d대입니다."
      %(myCar2.speed, myCar2.count))#출력
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\영섭\Desktop\2학년 1학기\수 고급프로그래밍\출석과제11\Code12-06.py =====
자동차1의 현재 속도는 30km, 생산된 자동차는 총 1대입니다.
자동차2의 현재 속도는 60km, 생산된 자동차는 총 2대입니다.
>>>
```


※code12-07

```
class Car : #Car클래스
    speed = 0 #정수형 변수 선언
    def upSpeed(self, value): #upSpeed 선언
        self.speed += value #speed에 value값을 더함
        print("현재 속도(슈퍼 클래스) : %d" % self.speed)#출력
class Sedan(Car) : #Sedan클래스
    def upSpeed(self, value): #upSpeed 선언
        self.speed += value#speed에 value값을 더함
        if self.speed > 150 :# if조건문 speed가 150초과 일 경우
            self.speed = 150#speed에 150저장
        print("현재 속도(서브 클래스) : %d" % self.speed) #출력
class Truck(Car) :#Truck클래스
    pass #pass함수 넘어감
sedan1, truck1 = None, None #변수 선언
truck1 = Truck() #Truck실행
sedan1 = Sedan()#Sedan실행
print("트럭 --> ", end = "")#출력
truck1.upSpeed(200) #truck1의 upspeed실행 (Car에서 실행)
print("승용차 --> ", end = "")#출력
sedan1.upSpeed(200) #sedan1의 upSpeed실행(Sedan에서 실행)
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\명섭\Desktop\2학년 1학기\수업 고급프로그래밍\출석과제11\Code12-07.py =====
트럭 --> 현재 속도(슈퍼 클래스) : 200
승용차 --> 현재 속도(서브 클래스) : 150
>>> |
```

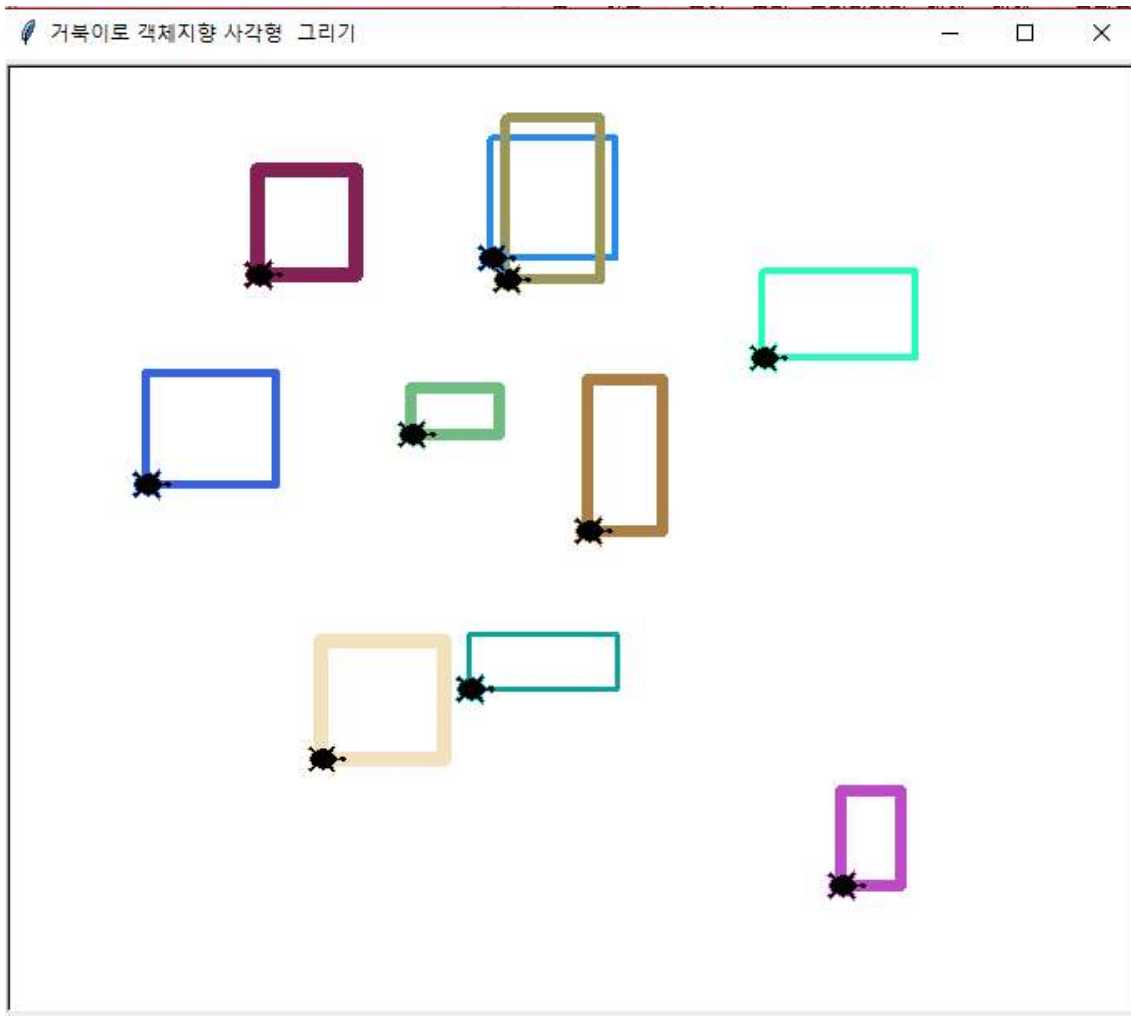
※code12-08

```
import turtle #turtle임포트
import random #random임포트
class Shape: #Shape클래스
    myTurtle = None #변수선언
    cx, cy = 0, 0 #정수형 변수 선언
    def __init__(self): #__init__선언
        self.myTurtle = turtle.Turtle('turtle')#myTurtle에 turtle지정
    def setPen(self) : #setPen선언
        r = random.random() #r값을 랜덤으로 불러옴
        g = random.random()#g값을 랜덤으로 불러옴
        b = random.random()#b값을 랜덤으로 불러옴
        self.myTurtle.pencolor((r, g, b)) #pencolor의 값을 지정함
        pSize = random.randrange(1,10) #pSize의 값을 1부터 10중하나로 불러옴
        self.myTurtle.pensize(pSize) #pensize값 지정
    def drawShape(self): #drawShape선언
        pass #pass
class Rectangle(Shape): #Rectangle 선언
    width, height = [0] * 2 #리스트 변수 선언
    def __init__(self, x, y): #__init__선언 항상실행
        Shape.__init__(self) # __init__모양 설정
        self.cx = x #cx값 지정
        self.cy = y #cy값 지정
        self.width = random.randrange(20, 100) #width의 값을 20과 100사이의 하나로
지정
        self.height = random.randrange(20, 100) #height의 값을 20과 100사이의 하나로
지정
    def drawShape(self) : #drawShape선언
        sx1, sy1, sx2, sy2 = [0] * 4 #리스트변수 선언
        sx1 = self.cx - self.width/2 #sx1에 self.cx - self.width/2값저장
        sy1 = self.cy - self.height /2 #sy1에 self.cy - self.height/2값저장
        sx2 = self.cx + self.width/2#sx2에 self.cx + self.width/2값저장
        sy2 = self.cy + self.height /2#sy1에 self.cy + self.height/2값저장
        self.setPen()# setPen()실행
        self.myTurtle.penup()#펜을 쓰지않음
        self.myTurtle.goto(sx1, sy1) #sx1,sy1로 이동
        self.myTurtle.pendown() #펜을 적음
        self.myTurtle.goto(sx1, sy2)#sx1,sy2로 이동
```

```

self.myTurtle.goto(sx2, sy2)#sx2,sy2로 이동
self.myTurtle.goto(sx2, sy1)#sx2,sy1로 이동
self.myTurtle.goto(sx1, sy1)#sx1,sy1로 이동
def screenLeftClick(x,y): #screenLeftClick선언
    rect = Rectangle(x, y) #rect에 Rectangle(x, y)실행한값 저장
    rect.drawShape() #rect에 저장되어있는값으로 drawShape실행
turtle.title('거북이로 객체지향 사각형 그리기') #turtle창의 제목설정
turtle.onscreenclick(screenLeftClick,1) #screenLeftClick실행
turtle.done() #마침

```



※code12-09

```
class Line: #Line 클래스
    length = 0 #정수형 변수선언
    def __init__(self, length) : #__init__선언
        self.length = length #length에 length저장
        print(self.length, '길이의 선이 생성되었습니다.')
```

#출력

```
    def __del__(self) : #__del__선언
        print(self.length, '길이의 선이 제거되었습니다.')
```

#출력

```
    def __repr__(self) : #__repr__선언
        return '선의 길이 : ' + str(self.length)# return  선의 길이 : ' + str(self.length)

    def __add__(self, other):# __add__선언
        return self.length + other.length# return self.length + other.length

    def __lt__(self, other) :#__lt__선언
        return self.length < other.length# return  self.length < other.length

    def __eq__(self, other) :#__eq__선언
        return self.length == other.length# return elf.length == other.length

myLine1 = Line(100) #myLine1에 Line(100)값저장
myLine2 = Line(200)#myLine2에 Line(200)값저장
print(myLine1) #출력
print('두 선의 길이 합 : ', myLine1 + myLine2)#출력
if myLine1 < myLine2 : #if조건문
    print('선분 2가 더 기네요.')
```

#출력

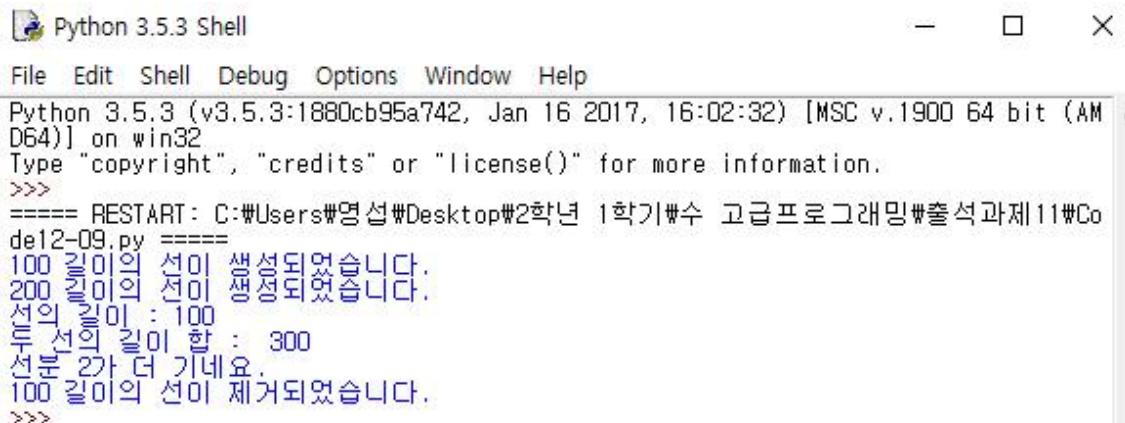
```
elif myLine1 == myLine2 : #elif조건문
    print('두 선분이 같네요.')
```

#출력

```
else : #else조건문
    print('모르겠네요.')
```

#출력

```
del(myLine1) #myLine를 지움
```



```
Python 3.5.3 Shell
File Edit Shell Debug Options Window Help
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\#영섭\Desktop\#2학년 1학기#수 고급프로그래밍#출석과제11#Code12-09.py =====
100 길이의 선이 생성되었습니다.
200 길이의 선이 생성되었습니다.
선의 길이 : 100
두 선의 길이 합 : 300
선분 2가 더 기네요.
100 길이의 선이 제거되었습니다.
>>>
```

※code12-10

```
class SuperClass :#SuperClass클래스
    def method(self) : #method선언
        pass#pass
class SubClass1 (SuperClass) :#SubClass1클래스
    def method(self) : #method선언
        print('SubClass1에서 method()를 오버라이딩 함')#출력
class SubClass2 (SuperClass) : #SubClass2클래스
    pass #pass
sub1 = SubClass1()#sub1에 SubClass1을 실행시킨값 지정
sub2 = SubClass2()#sub2에 SubClass2을 실행시킨값 지정
sub1.method() #sub1에서 method 실행
sub2.method()#sub2에서 method 실행
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\명섭\Desktop\2학년 1학기\수 고급프로그래밍\출석과제11\Code12-10.py =====
SubClass1에서 method()를 오버라이딩 함
>>> |
```

※code12-11

```
import time#time임포트
class RacingCar : #RacingCar클래스
    carName = '' #문자 변수 선언
    def __init__(self, name) : #__init__ 선언
        self.carName = name #carName에 name저장
    def runCar(self) : #runCar선언
        for _ in range(0, 3) : #for문
            carStr = self.carName + '~~ 달립니다.Wn' #문자열 저장
            print(carStr, end = '') #출력
            time.sleep(0.1) #0.1초 멈춤
car1 = RacingCar('@자동차1') #car1에 RacingCar클래스 실행
car2 = RacingCar('#자동차2') #car2에 RacingCar클래스 실행
car3 = RacingCar('$자동차3') #car3에 RacingCar클래스 실행
car1.runCar() #car1이 runCar실행
car2.runCar()#car1이 runCar실행
car3.runCar()#car1이 runCar실행
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\명섭\Desktop\2학년 1학기\수업\고급프로그래밍\출석과제11\Code12-11.py =====
@자동차1~~ 달립니다.
@자동차1~~ 달립니다.
@자동차1~~ 달립니다.
#자동차2~~ 달립니다.
#자동차2~~ 달립니다.
#자동차2~~ 달립니다.
$자동차3~~ 달립니다.
$자동차3~~ 달립니다.
$자동차3~~ 달립니다.
>>>
```

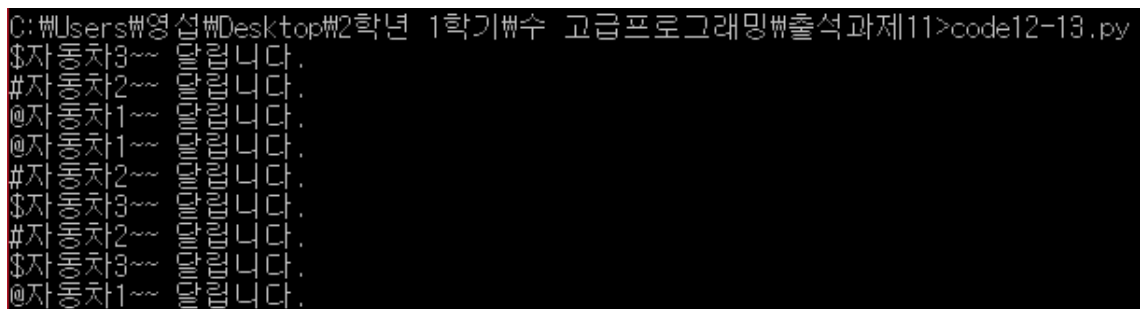
※code12-12

```
import threading #threading임포트
import time#time임포트
class RacingCar : #RacingCar클래스
    carName = '' #문자변수 선언
    def __init__(self, name) : #__init__선언
        self.carName = name #carName에 name 저장
    def runCar(self) : #runCar선언
        for _ in range(0, 3) : #for문
            carStr = self.carName + '~~ 달립니다.Wn'#carStr에 문자열 저장
            print(carStr, end = '') #출력
            time.sleep(0.1) #0.1초 느림
car1 = RacingCar('@자동차1') #car1에 RacingCar클래스 실행
car2 = RacingCar('#자동차2') #car2에 RacingCar클래스 실행
car3 = RacingCar('$자동차3') #car3에 RacingCar클래스 실행
th1 = threading.Thread(target = car1.runCar)#?
th2 = threading.Thread(target = car2.runCar)#?
th3 = threading.Thread(target = car3.runCar)#?
th1.start()#th1.시작
th2.start()#th2시작
th3.start()#th3 시작
```

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 16:02:32) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\#영섭\Desktop#2학년 1학기#수 고급프로그래밍#출석과제11#Code12-12.py =====
>>> @자동차1~~ 달립니다.
#자동차2~~ 달립니다.
$자동차3~~ 달립니다.
#자동차2~~ 달립니다.
$자동차3~~ 달립니다.
@자동차1~~ 달립니다.
#자동차2~~ 달립니다.
@자동차1~~ 달립니다.
$자동차3~~ 달립니다.
```

※code12-13

```
import threading #threading임포트
import time#time임포트
class RacingCar : #RacingCar클래스
    carName = '' #문자변수 선언
    def __init__(self, name) : #__init__선언
        self.carName = name #carName에 name 저장
    def runCar(self) : #runCar선언
        for _ in range(0, 3) : #for문
            carStr = self.carName + '~~ 달립니다.Wn'#carStr에 문자열 저장
            print(carStr, end = '') #출력
            time.sleep(0.1) #0.1초 느림
if __name__ == "__main__" :#main
    car1 = RacingCar('@자동차1') #car1에 RacingCar클래스 실행
    car2 = RacingCar('#자동차2') #car2에 RacingCar클래스 실행
    car3 = RacingCar('$자동차3') #car3에 RacingCar클래스 실행
    mp1 = multiprocessing.Process(target=car1.runCar)#?
    mp2 = multiprocessing.Process(target=car2.runCar)#?
    mp3 = multiprocessing.Process(target=car3.runCar)#?
    mp1.start()#mp1시작
    mp2.start()#mp2시작
    mp3.start()#mp3시작
    mp1.join()#mp1.만날때
    mp2.join()#mp1.만날때
    mp3.join()#mp1.만날때
```



```
C:\Users\영섭\Desktop\2학년 1학기\수업 고급프로그래밍\출석과제11>code12-13.py
$자동차3~~ 달립니다.
#자동차2~~ 달립니다.
@자동차1~~ 달립니다.
@자동차1~~ 달립니다.
#자동차2~~ 달립니다.
$자동차3~~ 달립니다.
#자동차2~~ 달립니다.
$자동차3~~ 달립니다.
@자동차1~~ 달립니다.
```


※응용예제1

```
from tkinter import * #tkinter모듈 임포트
from tkinter.ttk import * #tkinter.ttk모듈임포트
import random #random모듈 임포트
import time #time모듈 임포트
import threading #threading임포트
class ThreadProgressBar(): #ThreadProgressBar임포트
    thread = None #변수선언
    progress = None #변수선언
    def __init__(self, parent): #__init__선언 항상실행
        self.progress = Progressbar(parent, orient = HORIZONTAL, length =
500)#progress에 Progressbar저장
        self.progress.pack(side = TOP, fill = X, ipadx = 10, ipady = 10, padx =
10, pady = 10)#화면에 출력
        self.thread = threading.Thread(target=self.runProgress, args =
(self.progress))# thread에threading.Thread저장
        self.thread.start() #thread시작
    def runProgress(self, progress) : #runProgress선언
        hop = 0 #정수 선언
        while True : #while무한반복문
            hop = random.randrange(0, 10)#hop에 0,10사이의 랜덤값 저장
            if progress['value'] >= 100 : #if 조건문
                break #멈춤
            progress['value'] += hop#progress['value']에 hop더함
            time.sleep(0.5) #0.5초 느림
    def runThreadProgress() : #runThreadProgress선언
        thBar1 = ThreadProgressBar(window)#ThreadProgressBar를thBar1에 저장
        thBar2 = ThreadProgressBar(window)#ThreadProgressBar를thBar2에 저장
        thBar3 = ThreadProgressBar(window)#ThreadProgressBar를thBar3에 저장
if __name__ == "__main__" : #main
    window = Tk() #윈도우창에 화면출력
    window.geometry("300x250") #화면 크기 지정
    window.title('멀티 스레드') #화면 이름 지정
    threadtButton = Button(window, text = '멀티 스레드 시작', command =
runThreadProgress) #버튼생성
    threadtButton.pack(side = TOP, fill = X, ipadx = 10, ipady = 10, padx = 10,
pady = 10) #버튼을 화면에 출력
    window.mainloop() #mainloop
```


※응용예제2

```
from tkinter import * #tkinter모듈 임포트
import math #math임포트
import random#random임포트
class Shape:#shape 클래스
    color, width = '', 0 #변수선언
    shX1, shY1, shX2, shY2 = [0] * 4 #리스트 변수 선언
    def drawShape(self):#drawShape선언
        raise NotImplementedError() #?
class Rectangle(Shape):#Rectangle 선언
    objects = None#변수선언
    def __init__(self, x1, y1, x2, y2, c, w):#__init__선언
        self.shX1 = x1 #shX1에 x1값 저장
        self.shY1 = y1#shY1에 y1값 저장
        self.shX2 = x2#shX2에 x2값 저장
        self.shY2 = y2#shY2에 y2값 저장
        self.color = c #color에 c값 저장
        self.width = w #width에 w값저장
        self.drawShape() #drawShape()실행
    def __del__(self) : #__del__선언
        for obj in self.objects : #for문
            canvas.delete(obj) #canvas에 obj지움
    def drawShape(self) : #drawShape선언
        sx1 = self.shX1; sy1 = self.shY1; sx2 = self.shX2; sy2 =self.shY2 #변수값 저장
        squireList = [] #리스트 변수 선언
        squireList.append(canvas.create_line(sx1, sy1, sx1, sy2, fill = self.color, width
= self.width))#squireList에 추가
        squireList.append(canvas.create_line(sx1, sy2, sx2, sy2, fill = self.color, width
= self.width))#squireList에 추가
        squireList.append(canvas.create_line(sx2, sy2, sx2, sy1, fill = self.color, width
= self.width))#squireList에 추가
        squireList.append(canvas.create_line(sx2, sy1, sx1, sy1, fill = self.color, width
= self.width))#squireList에 추가
        self.objects=squireList #objects에squireList값 저장
class Circle(Shape) : #Circle클래스
    objects = None #변수선언
    def __init__(self, x1, y1, x2, y2, c, w):#__init__선언
        self.shX1 = x1 #shX1에 x1값 저장
        self.shY1 = y1#shY1에 y1값 저장
```

```

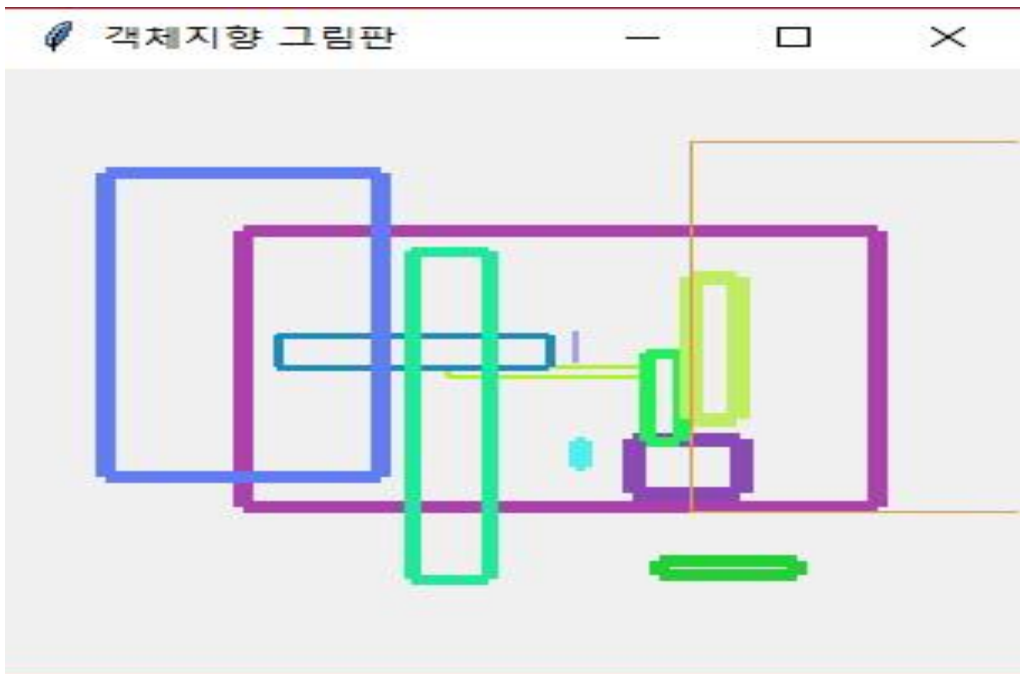
        self.shX2 = x2#shX2에 x2값 저장
        self.shY2 = y2#shY2에 y2값 저장
        self.color = c #color에 c값 저장
        self.width = w #width에 w값저장
        self.drawShape() #drawShape()실행
def __del__(self) : #__del__선언
    canvas.delete(self.objects)#canvas에 objects지움
def drawShape(self) : #drawShape선언
    sx1= self.shX1;    sy1= self.shY1;    sx2 = self.shX2;    sy2 = self.shY2#변수값
저장
    self.objects = canvas.create_oval(sx1, sy1, sx2, sy2, outline = self.color, width
= self.width) #objects에 값 저장
def getColor() : #getColor선언
    r = random.randrange(16, 256)#r값 16~256사이의 값중 랜덤으로 지정
    g = random.randrange(16, 256)#g값 16~256사이의 값중 랜덤으로 지정
    b = random.randrange(16, 256)#b값 16~256사이의 값중 랜덤으로 지정
    return "#" + hex(r)[2:] + hex(g)[2:] + hex(b)[2:] #값 리턴
def getWidth() :#getWidth선언
    return random.randrange(1, 9)#1,9사이의 값을 랜덤으로 리턴
def startDrawRect(event):#startDrawRect선언
    global x1, y1, x2, y2, shapes #전역변수 불러오기
    x1 = event.x #x1에 x값 저장
    y1 = event.y#y1에 y값 저장
def endDrawRect(event): #endDrawRect선언
    global x1, y1, x2, y2, shapes #전역변수 불러오기
    x2 = event.x#x2에 x값 저장
    y2 = event.y #y2에 y값 저장
    rect = Rectangle(x1, y1, x2, y2, getColor(), getWidth())#rect에 Rectangle저장
    shapes.append(rect) #shapes리스트에 추가
def startDrawCircle(event):#startDrawCircle선언
    global x1, y1, x2, y2, shapes #전역변수 불러오기
    x1 = event.x #x1에 x값 저장
    y1 = event.y#y1에 y값 저장
def endDrawCircle(event):#endDrawCircle선언
    global x1, y1, x2, y2, shapes #전역변수 불러오기
    x2 = event.x#x2에 x값 저장
    y2 = event.y #y2에 y값 저장
    cir = Circle(x1, y1, x2, y2, getColor(), getWidth())#cir에 circle값 저장
    shapes.append(cir) #shapes리스트에 cir 추가
def deleteShape(event):#deleteShape선언
    global shapes #전역변수 불러오기

```

```

if len(shapes) != 0 : #if조건문
    shp = shapes.pop() #shp에 shapes의 마지막에 있는값 저장
    del(shp) #shp제거
shapes = [] #리스트변수선언
window = None #변수선언
canvas = None #변수선언
x1, y1, x2, y2 = None, None, None, None#변수선언
if __name__ == "__main__" : #main
    window=Tk()#윈도우에 화면출력
    window.title("객체지향 그림판") #화면 제목설정
    canvas = Canvas(window, height = 300, width = 300)캔버스의 크기설정
    canvas.bind("<Button-1>", startDrawRect) #이벤트 함수
    canvas.bind("<ButtonRelease-1>", endDrawRect)#이벤트 함수
    canvas.bind("<Button-2>", deleteShape)#이벤트 함수
    canvas.bind("<Button-3>", startDrawCircle)#이벤트 함수
    canvas.bind("<ButtonRelease-3>", endDrawCircle)#이벤트 함수
    canvas.pack()#canvas를 화면에 출력
    window.mainloop()#mainloop

```



우재남, 파이썬 of beginner, 한빛아카데미,
2017