CLASSES:

INHERITANCE & POLYMORPHISM

Overview:

- define new classes with inheritance

Inheritance

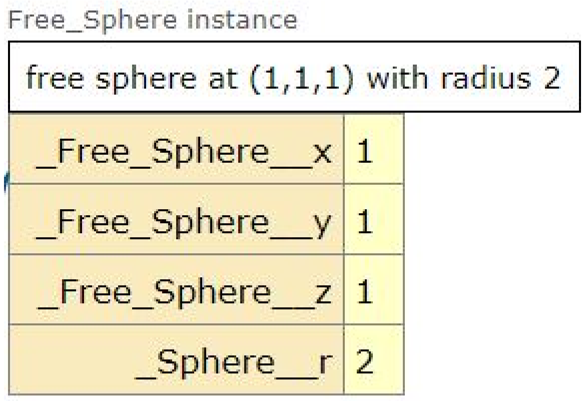
- can define new classes
  1. inherit parents methods
  2. can override parent methods
  3. can define new methods
- example: class *Free_Sphere*
  1. derived from *Sphere*
  2. has center at (x, y, z)
  3. inherits *volume*() method
  4. overrides *__str__*() method
  5. defines method *move*()

     $(x, y, z) \mapsto (x+dx, y+dy, z+dz)$

Free_Sphere Class

```
import Sphere   # all previous code for Sphere

class Free_Sphere(Sphere):
    def __init__(self,x=0,y=0,z=0,radius=1):
        Sphere.__init__(self, radius)
        self.__x = x
        self.__y = y
        self.__z = z

    def __str__(self):
        return 'free sphere at ({},{},{}) \
                radius {}'.format(self.__x,
                self.__y, self.__z,
                Sphere.get_radius(self))

    def move(self, dx=0, dy=0, dz=0):
        self.__x = self.__x + dx
        self.__y = self.__y + dy
        self.__z = self.__z + dz
```

Free_Sphere Instance

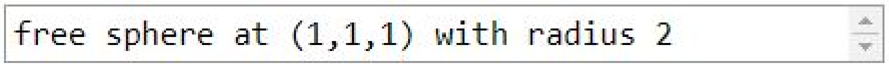

Free_Sphere instance

free sphere at (1,1,1) with radius 2

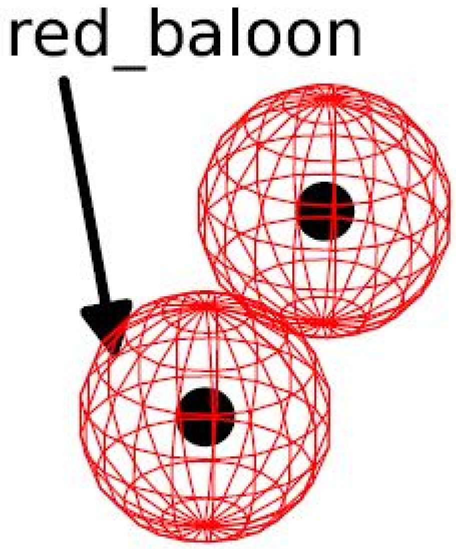
_Free_Sphere__x	1
_Free_Sphere__y	1
_Free_Sphere__z	1
_Sphere__r	2

- contains a *Sphere* instance

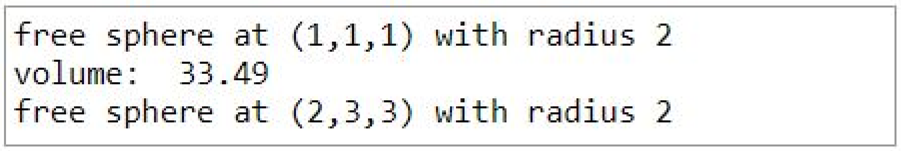
```
red_balloon = Free_Sphere(1, 1, 1, 2)
print(red_balloon)
```


```
free sphere at (1,1,1) with radius 2
```

Free_Sphere Class

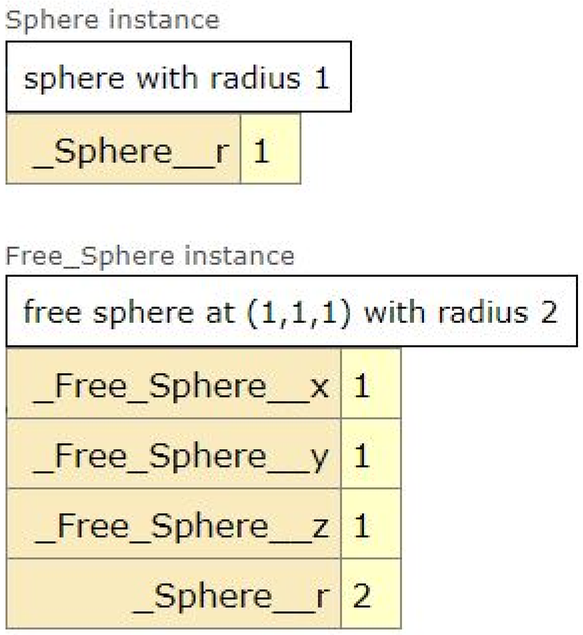


```
red_balloon = Free_Sphere(1, 1, 1, 2)
print(red_balloon)
volume = red_balloon.volume()
red_balloon.move(1,2,2)
print(red_balloon)
```



```
free sphere at (1,1,1) with radius 2
volume:  33.49
free sphere at (2,3,3) with radius 2
```

Polymorphism

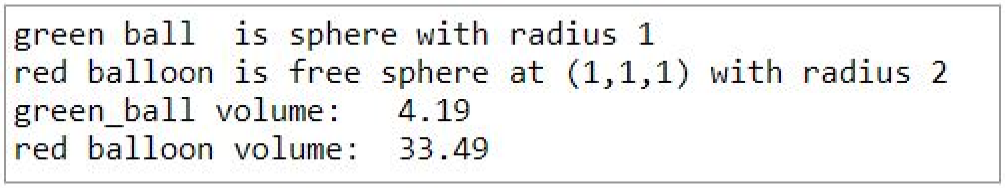


- both classes have *__str__*() method

```
green_ball = Sphere(1)
red_balloon = Free_Sphere(1, 1, 1, 2)

print('green ball  is', green_ball)
print('red balloon is', red_balloon)
```

Polymorphism (cont'd)

```
green_ball = Sphere(1)
red_balloon = Free_Sphere(1, 1, 1, 2)

print('green ball  is', green_ball)
print('red balloon is', red_balloon)

print('green_ball volume:   ',
                green_ball.volume())
print('red balloon volume: ',
               red_balloon.volume())
```



```
green ball  is sphere with radius 1
red balloon is free sphere at (1,1,1) with radius 2
green_ball volume:   4.19
red balloon volume:  33.49
```

- same function name in in different classes

Exercise(s):

- define a derived class *Shifted_Circle*
  1. takes radius and (x, y, z) coordinates for the center
  2. defines new method *distance*() to compute its distance from $(0, 0, 0)$
  3. overrides its *__str__*() method