Sınıflar ve Nesneler

Nesne yönelimli programlama

- Python, OOP
- Şimdiye kadar **yordamsal**
- Yordam = yöntem = fonksiyon odaklı programlama
- OOP ise **nesne** yönelimlidir
- Nesne = veri + işlevsellik

Kullanıcı tanımlı bileşik tipler

- Yeni veri türleri **sınıf** ile tanımlanır
- Yerleşik olanları kullandık
- Kendi türümüzü tanımlayacağız
- Nokta örneğini göz önüne alalım
- Nokta: (x, y)

Sınıf: Nokta

- Nokta sınıfı

```
class Nokta:
    pass
```

- Birleşik cümledir
- Yeni bir tür oluşturduk

Sınıf:örnek

- Yeni türün üyeleri = örnek = nesne
- Yeni bir örnek oluşturma süreci = örnekleme (instantiation)

```
>>> type(Nokta)
<type 'classobj'>
>>> p = Nokta()
>>> type(p)
<type 'instance'>
```

Nesne: Özellikler (veri)

- Yeni veri öğeleri ekleyebiliriz

```
>>> p = Nokta()
>>> p.x = 3
>>> p.y = 4
```

- Bunları ekrana yazdırırken

```
>>> print p.y
4
>>> x = p.x
>>> print x
3
```

Nesne: İlkleme

- Her noktanın x koordinatı olmalı fakat henüz söylememişiz!

```
>>> p2 = Nokta()
>>> p2.x
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
AttributeError: Nokta instance has no attribute 'x'
>>>
```

- Nasıl yapacağız?

Nesne: ilkleme

- İlkleme yöntemi kullanılır

```
class Nokta:
    def __init__(self, x=0, y=0):
        self.x = x
        self.y = y
```

- Artık p2.x geçerlidir.

Parametre olarak kullanmak

- Sınıf örneğini işleve parametre olarak göndermek

```
def noktayi_yaz(p):
    print '(%s, %s)' % (str(p.x), str(p.y))
```

Shallow X deep Copy -1

- İki nesnenin **aynı** olması ne demektir?
- Aynı değer X aynı nesne?

```
>>> p1 = Nokta()
>>> p1.x, p1.y = 3, 4
>>> p2 = Nokta()
>>> p2.x, p2.y = 3, 4
>>> p1 == p2
False
```

Shallow X deep Copy -2

- Bir de aşağıdaki kodlara bakın

```
>>> p2 = p1
>>> p1 == p2
True
```

- Bunlardan birisi shallow diğeri deep benzerliğe sahip

Shallow X deep Copy -3

```
def ayni_nokta(p1, p2):
    return (p1.x == p2.x) and (p1.y == p2.y)
```

- Test için (deep X shallow benzerlik?)

```
>>> p1 = Nokta()
>>> p1.x = 3
>>> p1.y = 4
>>> p2 = Nokta()
>>> p2.x = 3
>>> p2.y = 4
>>> ayni_nokta(p1, p2)
True
```

Nesne: Dikdörtgen

- Dikdörtgeni nasıl temsil ederiz?
- Sol-üst köşe ve sağ-alt köşe
- VEYA
- Sol-üst köşe ve genişlik-yükseklik

```
class Dikdortgen:
    pass
```

- tamamlayalım

Dönüş değeri olarak örnek

- Dönüş değeri olarak örnek döndüren işlev örneği

```
def merkezi_bul(box):
    p = Nokta()
    p.x = box.kose.x + box.genislik/2.0
    p.y = box.kose.y - box.yukseklik/2.0
    return p
```

- Test için

```
>>> merkez = merkezi_bul(box)
>>> noktayi_yaz(merkez)
(50.0, 100.0)
```

Nesne: değer (veri) güncelleme

- Dikdörtgeni genişletip-büyütelim

```
box.genislik = box.genislik + 50
box.yukseklik = box.yukseklik + 100
```

Shallow X deep Copy -4

- copy modülü yardımıyla deep copy

```
>>> import copy
>>> p1 = Nokta()
>>> p1.x = 3
>>> p1.y = 4
>>> p2 = copy.copy(p1)
>>> p1 == p2
False
>>> ayni_nokta(p1, p2)
True
```

- p3 = p (shallow copy)