PYTHON

İÇERİK

- ✓ **Sınıf(Class)**
- ✓ **Nesne (Obejct)**

KAZANIMLAR

- ✓ Programlama dilinde sınıf nedir ve kullanım alanları nelerdir öğrenir.
- ✓ Sınıflar ile ilgili uygulamalar yapabilir ve kendi sınıflarını oluşturabilir.
- ✓ Programlamada nesne nedir ve kullanım alanları nelerdir öğrenir.
- ✓ Nesneler ve sınıfları bir arada kullanmayı öğrenir ve uygulamalar yapar.

Nesne Tabanlı Programlama

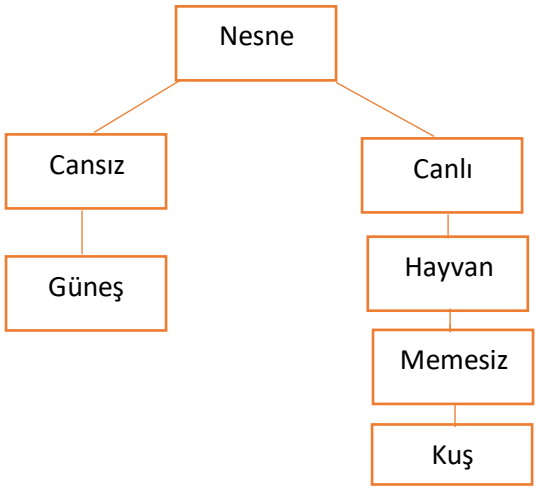
Bu bölümde, programlama faaliyetlerimizin önemli bir kısmını oluşturacak olan nesne tabanlı programlama yaklaşımına bir giriş yaparak, bu yaklaşımın temel kavramlarından biri olan sınıflara değineceğiz. Bu bölümde amacımız, sınıflar üzerinden hem nesne tabanlı programlamayı tanımak, hem bu yaklaşıma ilişkin temel bilgileri edinmek, hem de etrafımızda gördüğümüz nesne tabanlı yapıların büyük çoğunluğunu anlayabilecek seviyeye gelmek olacaktır. Bu bölümü tamamladıktan sonra, nesne tabanlı programlamayı orta düzeyde bilecek duruma geleceğiz.

Şimdiye kadar Python programlama dili ile ilgili olarak gördüğümüz konulardan öğrendiğimiz çok önemli bir bilgi var: Aslına bakarsak, bu programlama dilinin bütün felsefesi, 'bir kez yazılan kodların en verimli şekilde tekrar tekrar kullanılabilmesi,' fikrine dayanıyor.

Geçen bölümde incelediğimiz modüller için de geçerli olduğunu bariz bir şekilde görebilirsiniz. Gömülü fonksiyonlar, kendi tanımladığımız fonksiyonlar, hazır modüller, üçüncü şahıs modülleri hep belli bir karmaşık süreci basitleştirme, bir kez tanımlanan bir prosedürün tekrar tekrar kullanılabilmesini sağlama amacı güdüyor.

Bu fikir nesne tabanlı programlama ve dolayısıyla 'sınıf' (*class*) adı verilen özel bir veri tipi için de geçerlidir. Bu bölümde, bunun neden ve nasıl böyle olduğunu bütün ayrıntılarıyla ele almaya çalışacağız.

Nesneleri Sınıflara Ayırmak: Etrafımızda birbirine benzeyen ve benzemeyen oldukça fazla nesne bulunmaktadır. Öyle ki bu nesnelerin çoğu yaşamamızın vazgeçilmez unsurları arasında yer almaktadır. Örneğin güneş ve kuş her ikisi de etrafımızda bulunan nesneler. İki nesne arasında farklar bunların şöyle ifade edebiliriz:



Python'da nesneler, sınıflar tarafından tanımlanır ve bunu şöyle düşünebiliriz: nesneleri gruplara ayırmanın bir yolu da bu örnekteki gibidir.Burada ana sınıf nesnedir. Nesneler sınıfı iki ayrı kategoriye ayrılır canlılar ve cansızlar. Cansız nesne güneş ve canlı nesne olan kuş ise öncelikle özelliğine göre hayvan, memeli ve kuş olarak ayrılmaktadır.

Sınıflar

Buraya gelene kadar Python'da pek çok veri tipi olduğunu öğrendik. Mesela önceki derslerimizde incelediğimiz listeler, demetler, karakter dizileri, sözlükler ve hatta fonksiyonlar hep birer veri tipidir. Bu tiplerin, verileri çeşitli şekillerde evirip çevirmemizi sağlayan birtakım araçlar olduğunu biliyoruz. İşte sınıflar da, tıpkı yukarıda saydığımız öteki veri tipleri gibi, verileri manipüle etmemizi sağlayan bir veri tipidir.

Peki bu bölümde ele alacağımız 'sınıf' (*class*) veri tipi ne işe yarar?

Sınıf, bir nesne oluşturucu veya nesneler oluşturmak için bir "taslak" gibidir. Bir sınıf oluşturmak için şu anahtar kelimeyi kullanılır class:

sıcaklık adında bir özelliğe sahip Güneş adında bir sınıf oluşturalım:

```
class Güneş:

    sıcaklık="Güneş'in yüzey sıcaklığı 5500 derecedir"
```

Güneş isminde bir sınıf oluşturduk ve sıcaklık adında özelliğini belirttik. Şimdi nesneler oluşturmak için bu Güneş adlı sınıfı kullanabiliriz.

Değer isminde bir nesne oluşturalım ve sıcaklık değerini yazdıralım.

```
değer=Güneş()

print(değer.sıcaklık)
```

Alacağımız çıktı aşağıdaki gibi olacaktır.

```
========================================== RESTART: C:/Us
Güneş'in yüzey sıcaklığı 5500 derecedir
>>>
```

Bir önceki örnekte bahsettiğimiz canlı ve cansız nesneler tablosu üzerinden sınıflar ve nesneler oluşturalım.

Öncelikle varlıklar isminde içi pass komutu ile tanımlamalar yapılmayacak bir sınıf oluşturulur.

```
class Varlıklar:
    pass
```

varlıklar sınıfı içerisinde bulunan iki canlı ve cansızlar sınıfını oluşturalım. Bu sınıfların varlıklar sınıfı içerisinde olduğunu belirtelim:

```
class cansız(Varlıklar):
    pass
class canlı(Varlıklar):
    print("canlı")
```

cansızlar sınıfının içerisine herhangi bir tanımlama yapmadan pass komutu ile geçtik. Canlılar sınıfına özellik olarak print komutu ile canlı yazdırılmasını istedik.

Şimdi canlılar sınıfının özelliklerini belirleyeceğimiz bir sınıf Hayvan isminde bir sınıf oluşturalım ve bu hayvanın sınıfının özelliklerini fonksiyonlar ile belirtelim:

```
class Hayvan(canlı):
    def breathe(self):
        print("nefes alır")

    def hareket(self):
        print("hareket eder")
    def ürer(self):
        print("Kendi benzer canlılar üretirler")
```

Hayvanlar sınıfının ardından bir alt sınıfı olan Memesiz sınıfını oluşturalım. Bu sınıfta tabi ki kalıtımını Hayvanlar adında oluşturduğumuz sınıftan alacak

```
class Memesiz(Hayvan):
    def memesizler(self):
        print("Memeli hayvanlar içerisine dahil olmayan hayvanlar")
```

Memesiz sınıfı içerisinde memesizler isimli bir fonksiyon oluşturduk ve bu fonksiyon oluşturduk ve özellikler yazdık.

Şimdi Kuş isminde bir sınıf oluşturalım:

```
class Kuş(Memesiz):
    def yetenek(self):
        print("Uçar")
```

Şimdi bütün sınıflarımızı oluşturduktan sonra hepsini kapsayan en alt sınıf olan ve bütün sınıflardan kalıtım alan Kuş sınıfını bir nesne oluşturarak atayalım.

```
nesne = Kuş()
```

artık bütün özellikler nesnen isimli nesnemizin içerisinde olduğu için bütün fonksiyonları bu nesne sayesinde çağırabiliriz.

```
nesne = Kuş()
nesne.nefes()
nesne.hareket()
nesne.ürer()
nesne.memesizler()
nesne.yetenek()
```

__İnit__ ()işlevi

Tüm sınıfların __init __ () adında bir işlevi vardır ve bu, sınıf başlatılırken her zaman çalıştırılır.

Nesne özelliklerine değer atamak için __init __ () işlevini veya nesne oluşturulurken yapılması gereken diğer işlemleri kullanın:

Kullanıcının girdiği bir kelimedeki sesli harfleri sayan bir kod yazalım ve sınıfları kullanalım.

```
class HarfSayacı:
    def __init__(self):
        self.sesli_harfler = 'aeıioöuü'
        self.sayaç = 0

    def kelime_sor(self):
        return input('Bir kelime girin: ')

    def seslidir(self, harf):
        return harf in self.sesli_harfler

    def artır(self):
        for harf in self.kelime:
            if self.seslidir(harf):
                self.sayaç += 1
        return self.sayaç

    def ekrana_bas(self):
        mesaj = "{} kelimesinde {} sesli harf var."
        sesli_harf_sayısı = self.artır()
        print(mesaj.format(self.kelime, sesli_harf_sayısı))

    def çalıştır(self):
```


```
        self.kelime = self.kelime_sor()
        self.ekrana_bas()


if __name__ == '__main__':
    sayaç = HarfSayacı()
    sayaç.çalıştır()
```