Bölüm 2

Temel Kavramlar: Nesneler ve Sınıflar

2.1 Giriş

Nesne tabanlı programlamanın (OOP) temel yapı taşları nesneler ve sınıflardır. Nesneler, gerçek dünyadaki varlıkları temsil ederken, sınıflar bu nesnelerin şablonlarını oluşturur. Bu bölümde, nesne ve sınıf kavramları, bunların özellikleri ve metodları, nesne yaratma süreci ve nesne odaklı düşünmenin önemi ele alınacaktır. Temel amacımız OOP'nin temel mekanizmalarını anlamak ve kendi sistemlerini nesne odaklı bir şekilde tasarlayabilmektir.

2.2 Nesne Nedir?

Bir nesne, veri (özellikler -attributes) ve bu veriler üzerinde işlem yapan metodların (fonksiyonlar -methods) birleşimidir. Gerçek dünyada bir nesneyi, örneğin bir arabayı ele alalım: Bir araba nesnesi "renk, hız, model" gibi özelliklere ve "hızlanma, fren yapma" gibi metodlara sahiptir. OOP'de nesneler, yazılım sistemlerini modüler ve organize bir şekilde tasarlamak için kullanılır.

Örneğin, bir *Öğrenci* nesnesi şu şekilde tanımlanabilir:

— **Özellikler**: Ad, soyad, öğrenci numarası, not ortalaması.

— **Metodlar**: Not ekle, ortalama hesapla, bilgileri göster.

2.3 Sınıf Nedir?

Sınıf, nesnelerin şablonudur ve nesnelerin nasıl oluşturulacağını tanımlar. Bir sınıf, özellikler ve metodlar içerir. Örneğin, bir *Araba* sınıfı, tüm araba nesnelerinin ortak özelliklerini (renk, hız) ve davranışlarını (hızlan, fren yap) tanımlar. Sınıflar, kodun yeniden kullanılabilirliğini ve modülerliğini artırır.

Aşağıda, bir *Araba* sınıfının sözde-kodu (pseudo-code) temsili verilmiştir:

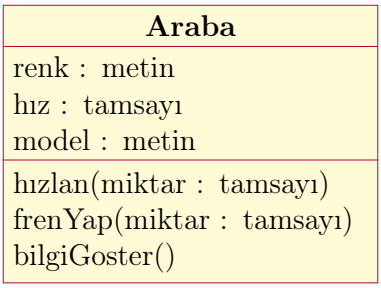
Sözde-kod 2.1 – Araba Sınıfı Tanımlama

```
sınıf Araba {
 özellik renk: metin
 özellik hız: tamsayı
 özellik model: metin

 metod hızlan(miktar: tamsayı) {
  hız = hız + miktar
 }

 metod frenYap(miktar: tamsayı) {
  hız = hız - miktar
  eğer hız < 0 ise
       hız = 0
 }

 metod bilgiGoster() {
  yaz "Model:", model, ", Renk:", renk, ", Hız:", hız
 }
}
```



Şekil 2.1 – Araba Sınıfının UML Diyagramı

2.4 Nesne Oluşturma ve Yaşam Döngüsü

Nesneler, sınıflardan örneklenerek (instantiation) oluşturulur. Bu süreç, genellikle bir *yapıcı* (constructor) metod ile başlar. Yapıcı, nesnenin başlangıç durumunu (örneğin, varsayılan değerler) ayarlar. Bazı dillerde, nesne yok edildiğinde çalışan bir *yıkıcı* (destructor) da bulunur, bu kaynakları serbest bırakır (örneğin, C++'ta).

Nesne yaşam döngüsü şu adımları içerir:

1. **Oluşturma**: Yapıcı ile nesne başlatılır.
2. **Kullanım**: Nesnenin metodları çağrılır, özellikleri güncellenir.
3. **Yok Etme**: Nesne artık gerekmediğinde bellekten kaldırılır (otomatik veya manuel, dile bağlı).

Şekil 2.2 – Nesne Yaşam Döngüsü

Örnek sözde-kod ile nesne oluşturma aşağıda verilmiştir:

Sözde-kod 2.2 – Nesne Oluşturma

```
sınıf Araba {
 özellik renk: metin
 özellik hız: tamsayı

 yapıcı(renk: metin, başlangıçHızı: tamsayı) {
  bu.renk = renk
  bu.hız = başlangıçHızı
 }

 metod hızlan(miktar: tamsayı) {
  hız = hız + miktar
 }

 metod frenYap(miktar: tamsayı) {
  hız = hız - miktar
  eğer hız < 0 ise hız = 0
 }

 metod bilgiGoster() {
  yaz ", Renk:", renk, ", Hız:", hız
 }
}

ana_program {
 araba1 = yeni Araba("Kırmızı", 0)  // Nesne yaratma
 araba1.hızlan(50)
 araba1.bilgiGoster()  // Çıktı: Renk: Kırmızı, Hız: 50
}
```

2.5 Nesne Odaklı Düşünme

Nesne odaklı düşünme (object thinking), yazılım geliştirirken gerçek dünya varlıklarını nesneler olarak modellemeyi içerir. Örneğin, bir kütüphane sistemi tasarlarken, *Kitap*, *Üye* gibi nesneler ve *Ödünç Alma* gibi metodlar tanımlanabilir. Bu yaklaşım, modülerliği ve bakım kolaylığını artırır. Modern yazılım mühendisliğinde, veri merkezli tasarım (data-driven design) bu fikri destekler; nesneler, verileri ve davranışları birleştirerek sistemin karmaşıklığını azaltır.

2.6 Alıştırmalar

1. Gerçek dünyadan bir nesne seçin (örneğin, bir telefon) ve bu nesneyi temsil eden bir sınıfın özelliklerini ve metodlarını listeleyin.
2. Yukarıdaki *Araba* sınıfını temel alarak, bir *Kamyon* sınıfı için sözde-kod yazın. Ek olarak, yük kapasitesi gibi bir özellik ekleyin.
3. Bir *Öğrenci* sınıfı tasarlayın ve bu sınıftan iki nesne yaratarak farklı metodları çağıran bir sözde-kod yazın.

4. Nesne yaşam döngüsünün adımlarını açıklayın ve bir nesnenin yaratılmasından yok edilmesine kadar geçen süreci bir örnekle tarif edin.
5. Şekil 2.1'deki UML diyagramını inceleyin ve başka bir sınıf (örneğin, *Bisiklet*) için benzer bir UML diyagramı çizin.