Nesne tabanlı programlama, gerçek dünyadaki nesneleri temsil eden sınıflar ve bu sınıflardan oluşturulan nesneler kullanarak programlama yapma şeklidir.

**Sınıflar**

Sınıflar, nesnelerin temel özelliklerini ve davranışlarını tanımlayan yapılardır. Sınıflar, değişkenler, metotlar ve alt sınıflar gibi üyeler içerebilir.

Örneğin, bir Kişi sınıfı tanımlayabiliriz:

```
class Kişi:
    isim = ""
    soyisim = ""
    yas = 0

    def info(self):
        print("İsim:", self.isim)
        print("Soyisim:", self.soyisim)
        print("Yaş:", self.yas)
```

Bu sınıf, isim, soyisim ve yas adlı üç değişken ve info() adlı bir metod içerir. info() metodu, nesnenin bilgilerini yazdırır.

**Nesneler**

Sınıflardan oluşturulan nesneler, sınıfın özelliklerine ve davranışlarına sahiptir.

Örneğin, aşağıdaki kodda Kişi sınıfından iki nesne oluşturuyoruz:

```
ali = Kişi()
veli = Kişi()

ali.isim = "Ali"
ali.soyisim = "Yılmaz"
ali.yas = 30

veli.isim = "Veli"
veli.soyisim = "Kalkan"
veli.yas = 25
```

ali ve veli nesneleri, Kişi sınıfının özelliklerine ve davranışlarına sahiptir. Örneğin, ali.info() ifadesini çalıştırırsak aşağıdaki çıktıyı alırız:

```
İsim: Ali
Soyisim: Yılmaz
Yaş: 30
```

**Nesne Yönelimli Programlamanın Faydaları**

Nesne tabanlı programlama, aşağıdaki faydaları sağlar:

- **Kodun okunabilirliğini ve anlaşılabilirliğini artırır.**
- **Kodun yeniden kullanılabilirliğini artırır.**
- **Kodun bakımını kolaylaştırır.**
- **Kodun daha esnek ve ölçeklenebilir olmasını sağlar.**

**Nesne Tabanlı Programlamanın Temel Kavramları**

Nesne tabanlı programlamanın temel kavramları şunlardır:

- **Sınıflar:** Nesnelerin temel özelliklerini ve davranışlarını tanımlayan yapılardır.
- **Nesneler:** Sınıflardan oluşturulan ve sınıfın özelliklerine ve davranışlarına sahip yapılardır.
- **Özellikler:** Nesnelerin sahip olduğu verilerdir.
- **Metotlar:** Nesnelerin gerçekleştirdiği işlemlerdir.
- **Soyutlama:** Gerçek dünyanın karmaşıklığını basitleştirerek kodlamayı kolaylaştırmak için kullanılan bir tekniktir.
- **Enkapsülasyon:** Nesnelerin özelliklerini ve davranışlarını dış dünyayla izole etmek için kullanılan bir tekniktir.
- **Kalıtım:** Bir sınıfın özelliklerini ve davranışlarını başka bir sınıfa miras alma işlemidir.

**Nesne Tabanlı Programlama Örnekleri**

Nesne tabanlı programlama, birçok farklı alanda kullanılan yaygın bir programlama yaklaşımıdır. Örneğin, aşağıdaki uygulamalarda nesne tabanlı programlama kullanılabilir:

- **Web uygulamaları:** Web uygulamaları, nesne tabanlı programlama kullanarak kullanıcı arayüzleri, veri modelleri ve iş mantığı gibi farklı bileşenleri oluşturabilir.
- **Mobil uygulamalar:** Mobil uygulamalar, nesne tabanlı programlama kullanarak kullanıcı arayüzleri, veri modelleri ve iş mantığı gibi farklı bileşenleri oluşturabilir.
- **Veritabanı uygulamaları:** Veritabanı uygulamaları, nesne tabanlı programlama kullanarak veritabanı tablolarını ve kayıtlarını temsil edebilir.
- **Grafik uygulamaları:** Grafik uygulamaları, nesne tabanlı programlama kullanarak kullanıcı arayüzleri ve grafik nesneleri oluşturabilir.
- **İş uygulamaları:** İş uygulamaları, nesne tabanlı programlama kullanarak iş süreçlerini ve iş verilerini temsil edebilir.

Nesne tabanlı programlama, modern programlamada yaygın olarak kullanılan güçlü bir araçtır. Bu yaklaşımı öğrenmek, programlama becerilerinizi geliştirmenize ve daha karmaşık ve esnek uygulamalar geliştirmenize yardımcı olabilir.

Python ile nesne tabanlı programlama için bir örnek ve açıklama yapalım.

**Örnek:**

Bir **Kişi** sınıfı tanımlayalım. Bu sınıf, bir kişinin **isim**, **soyisim** ve **yaş** bilgilerini tutacak.

Python

```
class Kişi:
    def __init__(self, isim, soyisim, yaş):
        self.isim = isim
        self.soyisim = soyisim
        self.yaş = yaş

    def info(self):
        print("İsim:", self.isim)
        print("Soyisim:", self.soyisim)
        print("Yaş:", self.yaş)
```

Bu sınıfı kullanarak bir **Kişi** nesnesi oluşturalım:

Python

```
ahmet = Kişi("Ahmet", "Mehmet", 30)
```

Bu nesneye **info()** metodu ile erişelim:

Python

```
ahmet.info()
```

Bu kod, aşağıdaki çıktıyı üretecektir:

```
İsim: Ahmet
Soyisim: Mehmet
Yaş: 30
```

**Açıklama:**

- **class** anahtar kelimesiyle bir sınıf tanımladık.
- **init()** metodu, bir nesnenin oluşturulduğunda çağrılan ilk metottur. Bu metot, nesnenin özelliklerini ayarlamak için kullanılır.
- **info()** metodu, nesnenin bilgilerini yazdırmak için kullanılır.

**Nesne Tabanlı Programlamanın Temel Kavramları:**

- **Sınıf:** Nesnelerin temelini oluşturan yapıdır. Sınıflar, nesnenin özelliklerini ve davranışlarını tanımlar.
- **Nesne:** Sınıftan oluşturulan bir örnektir. Nesneler, sınıfın özelliklerine ve davranışlarına sahiptir.
- **Özellik:** Nesnenin sahip olduğu veridir.
- **Metot:** Nesnenin gerçekleştirdiği eylemdir.

**Nesne Tabanlı Programlamanın Faydaları:**

- **Kod tekrarını azaltır.**
- **Kodun okunabilirliğini ve anlaşılabilirliğini artırır.**
- **Kodun bakımını kolaylaştırır.**
- **Esneklik ve genişletilebilirlik sağlar.**

**Nesne Tabanlı Programlama Örnekleri:**

- **Kişi** sınıfı örneğinde olduğu gibi, gerçek dünyadaki varlıkları temsil etmek için nesneler kullanılabilir.
- **Bir oyun geliştirirken, karakterler, nesneler ve seviyeler gibi nesneler kullanılabilir.**
- **Bir web uygulaması geliştirirken, kullanıcılar, sayfalar ve veriler gibi nesneler kullanılabilir.**

Nesne tabanlı programlama, modern programlamada yaygın olarak kullanılan bir yöntemdir. Bu yöntemi öğrenerek, daha esnek, okunabilir ve bakımı kolay kodlar yazabilirsiniz.

**Örnek 1: Kişi Sınıfı**

**Kod:**

```
class Kisi:
    def __init__(self, ad, soyad, yas):
        self.ad = ad
        self.soyad = soyad
        self.yas = yas

    def bilgileri_goster(self):
        print("Ad:", self.ad)
        print("Soyad:", self.soyad)
        print("Yaş:", self.yas)


kisi1 = Kisi("Mehmet", "Yılmaz", 25)
kisi1.bilgileri_goster()
```

**Açıklama:**

Bu örnekte, Kisi adında bir sınıf tanımladık. Bu sınıf, bir kişinin ad, soyad ve yaş bilgilerini temsil ediyor. Sınıfın __init__() metodu, bir kişi nesnesi oluşturulurken çağrılır ve nesnenin özelliklerini alır. bilgileri_goster() metodu ise nesnenin bilgilerini ekrana yazdırır.

**Örnek 2: Üçgen Sınıfı**

**Kod:**

```
class Ucgen:
    def __init__(self, kenar1, kenar2, kenar3):
        self.kenar1 = kenar1
        self.kenar2 = kenar2
        self.kenar3 = kenar3

    def alan_hesapla(self):
        s = (self.kenar1 + self.kenar2 + self.kenar3) / 2
        return math.sqrt(s * (s - self.kenar1) * (s - self.kenar2) *
(s - self.kenar3))

ucgen1 = Ucgen(3, 4, 5)
print(ucgen1.alan_hesapla())
```

**Açıklama:**

Bu örnekte, Ucgen adında bir sınıf tanımladık. Bu sınıf, bir üçgenin kenar uzunluklarını temsil ediyor. Sınıfın __init__() metodu, bir üçgen nesnesi oluşturulurken çağrılır ve nesnenin özelliklerini alır. alan_hesapla() metodu ise üçgenin alanını hesaplar.

**Nesne Tabanlı Programlamanın Faydaları**

Nesne tabanlı programlamanın birçok faydası vardır. Bunlardan bazıları şunlardır:

- **Kod tekrarını azaltır.** Bir sınıfı tanımlayarak, o sınıftan oluşturulan tüm nesneler için ortak olan özellikleri ve davranışları tanımlamış oluruz. Bu, aynı özelliklerin ve davranışların birden fazla kez tanımlanmasını önler.
- **Kodun okunabilirliğini ve anlaşılabilirliğini artırır.** Nesneler, gerçek dünyadaki nesneleri temsil eder. Bu, kodun daha kolay anlaşılmasını sağlar.
- **Programın esnekliğini ve sürdürülebilirliğini artırır.** Nesne tabanlı programlama, programın yeni özellikler eklenmesine veya mevcut özelliklerin değiştirilmesine olanak tanır.

**Sonuç olarak, nesne tabanlı programlama, karmaşık programlar geliştirmek için güçlü bir araçtır.**