

NESNE TABANLI PROGRAMLAMA

- Klasik programlamada veriler ve fonksiyonlar vardı. Her şey veri ve metotlarla yapılıyordu.
- Bu ders ile Algoritmik düşünmeyi geliştirmenizin yanı sıra nesne merkezli düşünme ile algoritmaların ve problemlerin modellenmesi gerçekleştirilecektir.
- Ders için Java'da program derlemek ve çalıştırmak için Java Development Kit (JDK) ve geliştirme ortamı olan Eclipse IDE (Integrated Development Environment) kullanılacaktır.

Alternatif olarak;

- Program yazabilmek için <https://www.oracle.com/java/technologies/downloads/#jdk17-windows> adresinden işletim sisteminize uygun Oracle JDK'yi indirip kurmalısınız.
- <https://www.eclipse.org/downloads/packages/> adresinden kullandığınız platforma uygun **Eclipse IDE for Java Developers**'i indirin.

- İnsanlar açısından nesne, bir varlığın temsil biçimidir. Örneğin insan bir nesnedir. İnsanı bilgisayar yazılımı içinde temsil etmek istersek, insanın ayırt edici özelliklerini belirlemeliyiz;
 - Boy
 - Göz rengi
 - Kilo
 - IQ
 - Konuşma şekli

- Nesnenin karakteristikleri, özellikler ve gerçekleştirebileceği eylemler de metot adını alır. Böylece soyutlaştırma sonucunda nesne özellikleri ve metotlar ile temsil edilir.

- NTP de soyutlamayı sınıf yapısı ile gerçekleştiririz. Sınıf yapısı içinde o sınıfa ait özellikler ve metotlar tanımlanır.
- Sınıftan üretilen her örnek, aynı özellik ve metotlara sahip olacak ancak özelliklerin değerleri her örnek için birbirinden farklı olacaktır.

- NTP'nin 3 temel kavramı;
 - Kapsülleme (encapsulation)
 - Kalıtım (Inheritance)
 - Çok biçimlilik (Polimorphism)

Kapsülleme

- İşlemlerin ve verilerin gizlenmesine veya bir grup içerisinde toplanmasına kapsülleme denir. Sınıf içerisinde kullanılan alanların dışarıdan doğrudan değiştirilmesi engellenir. Böylece bazı değerler sadece programcının belirlediği çizgi dahilinde değişir.

Kalıtım

- Bir sınıftan yeni sınıflar türetmektir. Türetilen sınıf, türediği sınıfın özelliklerini taşır.

Çok biçimlilik

- Bir metodun farklı nesnelerde veya farklı ortamlarda farklı sonuçlar üretmesidir.

Java

- Sun tarafından 23 Mayıs 1995'te genel amaçlı bir programlama dili olarak piyasaya sunuldu.
- Netscape'in web yenilikleriyle birlikte gelişti
- 2010 yılında Oracle tarafından satın alındı.
- Syntax açısından C++'a çok benzeyen fakat daha soyut, daha küçük (yaklaşık 50 civarı keyword) ve yetenekli bir dil.

- Java hem derlenen hem yorumlanan bir dildir. Kaynak kodun derlendiği zamana derleme zamanı (compile time) çalıştırıldığı zamana çalışma zamanı (run time) denir. Derleme zamanı ile çalışma zamanı arasında kod bytecode'a (.class dosyası) (Platformdan bağımsız, JVM bağımlı)dönüştürülür.
- Yorumlamada kod, başka forma çevrilmeden yorumlayıcı tarafından doğrudan makine kodundaki bloklar çağrılarak çalıştırılır. Direkt çalıştırıldığı için hatalar çalışma zamanında fark edilir.
- Derlenen dillerdeki gibi .exe veya .out gibi bir çıktı kodu dosyası yoktur.

- Java'nın yaratıcısı James Gosling;
 - «Java; basit, nesne-merkezli, ağda yetenekli, yorumlanan, sağlam, güvenli, mimari olarak tarafsız (donanım ve işletim sistemi), taşınabilir (derlenmiş java kodu), yüksek performanslı, çok kanallı, dinamik bir dil»
- Java ile dünyada “*bir kere yaz, her yerde kullan*” devri başladı.

JVM

- Derlenmiş java kodlarını (bytecode) çalıştıran sistemdir.
- Just in Time Compiler barındırır.
- Java'nın çalışması gereken her platformda kurulu olmalıdır.
- Geliştiriciyi platformdan soyutlayan bir ara katmandır.

- Java nesne tabanlı bir dildir ve her şey bir nesnedir. Nesnenin yapısını da sınıflar oluşturur.
- Nesne özellik ve davranışlara sahip fiziksel yada fiziksel olmayan olgudur.

Türler

- Türler ikiye ayrılır.
 - Değer türleri (Stack) (Sbyte, Byte, Char, Float, Decimal, Double.....)
 - Referans türleri (Heap) (Class, String, Array.....)

Sınıf üyeleri

- Veri üyeleri ve fonksiyon üyeleri olarak ikiye ayrılır.

Veri üyeleri

- Sınıfın veri üyeleri değer türünde olabileceği gibi referans türünde de olabilir. Her nesnenin datası kendine aittir, hiçbir şekilde karışmaz. Veri üyeleri şu 3 öğeden oluşur;
 - Alanlar; nesneye ait bilgilerin tutulduğu üye değişkenler.
 - Sabitler; nesnenin değiştirilemeyen alanları.
 - Olaylar; belirli bir tetiklemenin meydana gelmesiyle çağrılacak olan fonksiyonların adres bilgilerini tutar.

Fonksiyon üyeleri

- Şunlardan oluşur;
 - Yöntemler (metotlar); nesnenin davranış şekilleridir.
 - Özellikler; nesnenin üye verilerine değer atamasının ve değer çekilmesinin kontrollü olarak yapılmasını sağlar.
 - Yapıcılar (Constructors); nesne ilk oluşturulduğunda çağrılır.
 - Yıkıcılar (Destructor); nesnenin ömrü tamamlandığında çağrılır.
 - Operatörler; nesnenin daha esnek kullanılmasına izin verirler.
 - Dizinleyiciler; nesnenin bir dizi gibi kullanılmasına izin verirler.

Metotlar ve dönüş değeri tipleri

- Tüm metotlar bir sınıfın üyesi olmak zorundadır.

[Erişim] <dönüş değeri> metot_ismi (parametre listesi)

{

metot gövde metni

}

Sınıflar

- Her nesne, mutlaka sınıflardan türetilmektedir.

```
Class sinif_ismi
```

```
{
```

```
    veri_tipi değişken;
```

```
    donus_değeri metot_ismi(parametreler)
```

```
{
```

```
}
```

```
}
```

Örnek

- Öğrenci isminde bir sınıfımız olsun. Sınıfın isim adında bir özelliği ve isim_ne, ekrana_yaz isimli iki metodu olsun. Main kısmında bu sınıftan bir öğrenci türetilsin ve öğrenciye isim atansın.

- Nesne tanımlayarak sınıfın özelliklerini nesneye yüklemiş oluruz. Sonra bu nesne ile sınıf içindeki metot ve değişkenlere ulaşabiliriz.

- Bir metot kullanılırken bulunduğu sınıf türünden bir nesne tanımlanır ve “.” operatörü ile metoda ulaşılır.
- Geri dönüş değeri olmayan metotlar “void” olarak tanımlanır. “void” tanımlı bir metot için “return” kullanılamaz.

- “static” olarak tanımlanan metotları çağırmak için nesne tanımlanmasına gerek yoktur. Eğer bulunduğu sınıfta çağırılacaksa ismini yazmak yeterlidir, başka sınıftan çağırılacaksa “sinifadi.metot();” şeklinde çağırılır.

Erişim belirleyiciler

- Erişim belirleyiciler ile nesne sahip olduğu üyelere kimin ve hangi şartla erişebileceğini belirler.

Erişim türleri

- Public; Bütün sınıflar erişebilir
- Private; Alt sınıf dahil başka hiçbir sınıf erişemez
- Protected; Alt sınıflar ve aynı pakettekiler erişebilir
- Private Protected; Yalnızca alt sınıflar erişebilir
- <default>; Aynı pakettekiler erişebilir

Kaynak

- <https://teknoloji.org/james-gosling-java-programlama-dilinin-oncusu>