S.O.L.I.D Prensibi:

1. Single Responsibility Principle (SRP)

“Her sınıfın veya metodun tek bir sorumluluğu olmalı”:

Bir sınıf (nesne) yalnızca bir amaç uğruna değiştirilebilir, o da o sınıfa yüklenen sorumluluktur, yani bir sınıfın(fonksiyona da indirgenebilir) yapması gereken yalnızca bir işi olması gerekir.

2. Open / Closed Principle (OCP)

“Sınıflar değişikliğe kapalı ancak gelişime açık olmalıdır.”:

Bir sınıf ya da fonksiyon halihazırda var olan özellikleri korumalı ve değişikliğe izin vermemelidir. Yani davranışını değiştirmiyor olmalı ve yeni özellikler kazanabiliyor olmalıdır.

3. Liskov’s Substitution Principle (LSP)

Kodlarımızda herhangi bir değişiklik yapmaya gerek duymadan alt sınıfları, türedikleri(üst) sınıfların yerine kullanabilmeliyiz.

4. İnterface Segregation Principle

“Arayüzlerin ayrılması prensibi”:

Sorumlulukların hepsini tek bir arayüze toplamak yerine daha özelleştirilmiş birden fazla arayüz oluşturmalıyız.

5. Dependency Inversion Principle

Sınıflar arası bağımlılıklar olabildiğince az olmalıdır özellikle üst seviye sınıflar alt seviye sınıflara bağımlı olmamalıdır.

OOP - Nesne Yönelimli Programlama:

Nesne yönelikli programlamaya örnek verecek olursak gerçek hayatta gördüğümüz araba, radyo, bina… gibi nesnelerin bilgisayar ortamına aktarılmasına denir.

1. Özellikleri

- Soyutlama (Abstraction): Bir sınıfta davranış ve özelliklerin tanımlanmasına soyutlama diyoruz.
- Kapsülleme (Encapsulation): Davranış ve özellikler sınıfta soyutlanarak kapsüllenir. Kapsülleme ile hangi özellik ve davranışın dışarıya sunulup sunulmayacağını belirleriz.
- Miras Alma (Inheritance): Sınıflar birbirinden türeyebilir. Alt sınıf üst sınıfın özelliklerini alabilir.
- Çok biçimlilik (Polymorphism): Alt sınıflar üst sınıfın gösterdiği davranışları göstermek zorunda değildir. Alt sınıfların farklı davranışları göstermesine Çok biçimlilik denilmektedir.

Stack ve Heap:

RAM’in mantıksal bölümleridir diyebiliriz. Stack’de değer tipleri, pointer ve adresler saklanırken, Heap’de ise referans değerleri saklanmaktadır.

Get metodunda alınan parametreler ve içlerindeki bilgiler adres satırında görünür.

Post metodunda alınan parametreler ve içlerindeki bilgiler adres satırında görünmez.

API’leri yazılımların birbirleriyle konuşmasına izin veren arayüzler olarak açıklayabiliriz.

Abstract sınıflar genelde is-a ilişkilerinde kullanılır.
Örnek vermek gerekirse;
+Ferrari is-a Araba
Ferrari bir arabadır ve arabanın sahip olduğu tüm özelliklere sahiptir.

Interface’ler genelde can-do ilişkisi vardır.
Örnek vermek gerekirse;
+Ferrari can-do drive itself

Ovveride, Kalıtım aldığı ana sınıftaki alınan bir özelliği üzerini yazarak
değiştirilmesi işlemidir.

Overloading, aynı isme sahip ama farklı sayıda ve farklı parametre türü alan fonksiyonun farklı amaca uygun kullanılmasıdır.

MVC, Model (Veri-DB), View (arayüz) ve Controller (orta katman) olarak basitce üçe ayılmaktadır. MVC olacaktır. Model(Veri-DB), View(arayüz) ve Controller(orta katman) olarak basitce üçe ayrılmaktadır.
