**====ABSTRACT====**

(melek gibi beden yok) public abstract class Honda{

motor(); deri(){“koltuklar deri”}

public class Civic extends Honda{ public class Accord extens Honda{ public class Crv extends Honda

motor(){2.5} deri(){“gerçek deri”}

motor(){2.0} deri(){“suni”}

motor(){1.6}

public abstract class Lastik extends Honda{

Bazen child class lar parent class taki metodların body lerini hep değiştirerek kullanırlar. Bu tip durumlarda parent class taki metoda body (süslü parantez ve içini ) koymak manasız olur, çünkü parent class taki metod body hiç kullanılmaz. Bu manasızlığı gidermek için Java bize body siz metod üretme imkanı verir. Body siz metodlara abstract metod denir.

Parent class taki body siz metodları (abstract metod) child class lar override etmek (metod signature ına dokunmadan süslünün içini değiştirmek) zorundadırlar. Biz bir aplication oluştururken child class ları parent class taki bazı metodları override etmek zorunda bırakmak istersek o metodları abstract metod yaparız.

abstract metod oluşturmak için; a)”{ }” body kısmını yazmayın

b)Access modifier dan sonra abstract keyword ünü kullanın 🡺public abstract void motor();

abstract keyword ü ile {} metod body bir arada kullanılmaz.

abstract metodlar sadece abstract class lar içinde oluşturulabilirler.

abstract class oluşturmak için Access modifier dan sonra abstract keyword ünü kullanın.

abstract class içinde hem abstract hem de abstract olmayan (concrete) metodlar kullanılabilirler.[motor(); ve deri() {“koltuklar deri”}]

Bir concrete class (mesela child ı gibi ) abstract class a extend ettiğinde, abstract class içindeki tüm abstract metodları override etmek zorundadırlar. Aksi takdirde Java CTE verir. Fakat concrete metodları override etmek zorunda değildirler. Concrete metodların override edilmesi isteğe bağlıdır.(üstteki şekilde bu anlatılmış, Lastik class ı hariç).

Bir abstract class başka bir abstract class a extend ettiğinde herhangi bir metodu override etmek zorunda değildir (şekildeki Lastik abstract class ı gibi). İsterse override eder isterse etmez.

Concrete class larda abstract metod üretilemez. (Civic class ında public abstract void yakit(); şeklinde metod olamaz.)

“final class” ların child class ı olamaz (adı üstünde son class, altında bir class olamaz ki child ı olsun). abstract class lar child class lara talimat vermek için oluşturulurlar. Eğer bir class ı “final” yaparsanız o class ın child class ı olamayacağından abstract olması mantıklı değildir. Bu yüzden Java abstract class ların “final” olmasına müsaade etmez.

abstract metod üretmemizin sebebi child class ların kullanabilmesidir. Eğer metodu “final” yaparsak bu, o metod override edilemez demektir yani, child class lar kullanamaz demektir. Bu çelişkiden dolayı Java abstract metodların “final” olmasına müsaade etmez.

abstract metod lar private olamazlar, çünkü abstract metod lar başka class lar tarafından kullanılmak için üretilir. Halbuki private metodlar başka class lar tarafından kullanılamazlar.

abstract metod lar static olamazlar. Çünkü static metodlar override edilemezler. Halbuki biz abstract metodları override etmek için üretiriz.

**ABSTRACT CLASS LARDAN OBJECT ÜRETİLEMEZ!!!!** **Soyut Sınıf (Abstract Class)**

**Nesne tabanlı programlamada sınıf hiyerarşisi oluşturulurken, bazen en tepede bulunan sınıf türünden nesneler programcılar için anlamlı olmayabilir.**

**Hiyerarşinin en tepesinde bulunan sınıfın kendisinden türetilecek olan alt sınıflar için ortak bir arayüz (interface) görevi yapması istenebilir.**

**Bunun için çözüm olarak oluşturulan metotlara ve sınıflara soyut metot (abstract method) ya da soyut sınıf (abstract class) denir.**

**Soyut sınıflar büyük projelerde kullanılırlar ve kalıtım özelliğini kullanarak kod tekrarını azaltırlar.**

**Soyut sınıflar diğer sınıflara taban olmak için kullanılırlar. Nesne türetemezler. Önlerine “abstract” sözcüğü yazılarak soyutlaştırılırlar.**

**Önlerine "virtual" yazılmaz çünkü “abstract” sözcüğü uygulanan tüm sınıf ve metotlar zaten sanaldır.**

**Kullanırken dikkat edilmesi gerekenler:**

**• Soyut sınıflar “abstract” türünden nesneler tanımlamazlar.**

**• Soyut sınıflar içerilerinde soyut olmayan metotlar da barındırabilir ancak soyut metotlar sadece soyut sınıflar içerisinde bildirilebilir.**

**• Soyut metotlar türeyen sınıfta mutlaka bildirilmelidir.**

**• Soyut metotlar override edilmek zorundadırlar, aksi takdirde derleyici hatası alınır.**

**• “Static” metotlar soyut olarak tanımlanamazlar.**

**• Soyut sınıflar “private” olarak tanımlanamazlar ama “public” ve “protected” olarak tanımlanabilirler.**

**• Soyut sınıflar “sealed” anahtar sözcüğü ile ifade edilemezler.**

**public abstract class SoyutSınıf {**

**abstract public void Soyut\_Metot();//gövdesi olmayan metot**

**public void Soyut\_Olmayan\_Metot() //gövdesi olan metot**

**{**

**//metot gövdesi**

**}**

**}**

**Soyut sınıf ve arayüz arasındaki farklar ve benzerlikler:**

**• Arayüz içerisindeki tüm öğelerin “public” olması gerekirken soyut sınıflarda tüm öğelerin “public” olması zorunlu değildir.**

**• Bir sınıf sadece bir soyut sınıftan miras olarak tanımlanabilir ancak birçok arayüz tarafından bildirilebilir.**

**• Soyut sınıf ve arayüz için yazılacak olan soyut metotların gövdeleri bulunmaz.**

**• Soyut sınıflar çoklu ortamı desteklemezler ancak arayüz destekler.**

**• Soyut sınıflar kullanım hızı açısından arayüzden daha avantajlıdır.**

**• Soyut bir sınıfın tüm metotları soyut yapılırsa arayüz olarak kullanılabilir.**

**Uygulama**

**“Soyut\_Sınıf” adında bir soyut sınıf tanımlanır ve içine soyut olan ve olmayan metotlar eklenir.**

**Gövdesi olmayan soyut metot ve soyut olmayan metot tanımlanmıştır.**

**“Soyut\_Sınıf”tan kalıtılan “Program” adında sınıf tanımlanır. “Soyut\_Metot” override edilerek gövde kazandırılır.**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**namespace Abstract**

**{**

**abstract class Soyut\_Sınıf**

**{**

**public void Soyut\_Olmayan\_Metot()**

**{ Console.WriteLine("Soyut Olmayan Metot"); }**

**public abstract void Soyut\_Metot(); //Soyut Metotlar gövdesi olmayan metotlardır.**

**}**

**class Program : Soyut\_Sınıf**

**{**

**public override void Soyut\_Metot()**

**{**

**Console.WriteLine("Soyut Metot");**

**1- Bazı durumlarda child classlar parent class daki methodların genellikle badylerine değiştirerek ovarride ederler**

**Bu gibi durumlarda parent classdaki methoda body tanımlamak ram ve memory açısından gereksizdir.**

**çözüm olarak java parent class a body'siz --> abstract (soyut) methodlar tanımlanmıştır.**

**2- Parent class'daki abstract methodlar app güvenliği açısından child class'da override edilmek zorundadır.**

**eğer parent class daki child classın mutlaka kullanması gereken method varsa abstract tanımlanmalı**

**motor () her aracta olmalı**

**3- abstract method tanımlamak için;**

**a){} body olmaz**

**b) access modifier sonrası abstract keyword kullanılmalı**

**abstract class tanımlamak için class ifadesinden önce abstract keyword kullanılmalı**

**c) Abstract method içinde abstract keyword ve {} bir arada kullanılamaz. aksi halde CTE verir.**

**4- Abstract methodlar sadece abstract Classta tanımlanabilir , kesinlikle concrete classda abs. methot tanımlanamaz**

**ama abs class'da concerete method tanımlanabilir.**

**5-Abstract class içinde hem abstract method hem de concrete method ve variable tanımlanabilir.**

**6- bir concrete child class parent abstract classa extend ettiğinde abstract methodları implement etmek zorunda**

**ama concrete methodlar isteğe göre override edilir.**

**7- Parent child ilşkisi olan abstarct class'lar abs method'ları implement etmek zorunda değil**

**8- Parent olan abs class birden çok concerete veya abs child class exyends edibilir.**

**ancak child class birden cok abs parent class'dan extends e-de-mezz...**

**9- Abs Class'da final ve abs method tanımlanamaz CTE -> final method override edilemyecegi için abs olamaz**

**10- Abs Class'da private ve abs method tanımlanamaz CTE -> private method override edilemyecegi için abs olamaz**

**11- Abs Class'da static ve abs method tanımlanamaz CTE-> static method override edilemyecegi için abs olamaz**

**\*\*\*\* ABSTRACT CLASS ASLA KESİNLİKLE OBJE URETEMEZ !!!!**

**TRİCK-> Abstract Class'lar referansı child Class'ın obj tutabilir(Polimorphism)**

**Soyut Sınıf (Abstract Class)**

**Nesne tabanlı programlamada sınıf hiyerarşisi oluşturulurken, bazen en tepede bulunan sınıf türünden nesneler programcılar için anlamlı olmayabilir.**

**Hiyerarşinin en tepesinde bulunan sınıfın kendisinden türetilecek olan alt sınıflar için ortak bir arayüz (interface) görevi yapması istenebilir.**

**Bunun için çözüm olarak oluşturulan metotlara ve sınıflara soyut metot (abstract method) ya da soyut sınıf (abstract class) denir.**

**Soyut sınıflar büyük projelerde kullanılırlar ve kalıtım özelliğini kullanarak kod tekrarını azaltırlar.**

**Soyut sınıflar diğer sınıflara taban olmak için kullanılırlar. Nesne türetemezler. Önlerine “abstract” sözcüğü yazılarak soyutlaştırılırlar.**

**Önlerine "virtual" yazılmaz çünkü “abstract” sözcüğü uygulanan tüm sınıf ve metotlar zaten sanaldır.**

**Kullanırken dikkat edilmesi gerekenler:**

**• Soyut sınıflar “abstract” türünden nesneler tanımlamazlar.**

**• Soyut sınıflar içerilerinde soyut olmayan metotlar da barındırabilir ancak soyut metotlar sadece soyut sınıflar içerisinde bildirilebilir.**

**• Soyut metotlar türeyen sınıfta mutlaka bildirilmelidir.**

**• Soyut metotlar override edilmek zorundadırlar, aksi takdirde derleyici hatası alınır.**

**• “Static” metotlar soyut olarak tanımlanamazlar.**

**• Soyut sınıflar “private” olarak tanımlanamazlar ama “public” ve “protected” olarak tanımlanabilirler.**

**• Soyut sınıflar “sealed” anahtar sözcüğü ile ifade edilemezler.**

**public abstract class SoyutSınıf**

**{**

**abstract public void Soyut\_Metot();//gövdesi olmayan metot**

**public void Soyut\_Olmayan\_Metot() //gövdesi olan metot**

**{**

**//metot gövdesi**

**}**

**}**

**Soyut sınıf ve arayüz arasındaki farklar ve benzerlikler:**

**• Arayüz içerisindeki tüm öğelerin “public” olması gerekirken soyut sınıflarda tüm öğelerin “public” olması zorunlu değildir.**

**• Bir sınıf sadece bir soyut sınıftan miras olarak tanımlanabilir ancak birçok arayüz tarafından bildirilebilir.**

**• Soyut sınıf ve arayüz için yazılacak olan soyut metotların gövdeleri bulunmaz.**

**• Soyut sınıflar çoklu ortamı desteklemezler ancak arayüz destekler.**

**• Soyut sınıflar kullanım hızı açısından arayüzden daha avantajlıdır.**

**• Soyut bir sınıfın tüm metotları soyut yapılırsa arayüz olarak kullanılabilir.**

**Uygulama**

**“Soyut\_Sınıf” adında bir soyut sınıf tanımlanır ve içine soyut olan ve olmayan metotlar eklenir.**

**Gövdesi olmayan soyut metot ve soyut olmayan metot tanımlanmıştır.**

**“Soyut\_Sınıf”tan kalıtılan “Program” adında sınıf tanımlanır. “Soyut\_Metot” override edilerek gövde kazandırılır.**

**using System;**

**using System.Collections.Generic;**

**using System.Linq;**

**using System.Text;**

**using System.Threading.Tasks;**

**namespace Abstract**

**{**

**abstract class Soyut\_Sınıf**

**{**

**public void Soyut\_Olmayan\_Metot()**

**{ Console.WriteLine("Soyut Olmayan Metot"); }**

**public abstract void Soyut\_Metot(); //Soyut Metotlar gövdesi olmayan metotlardır.**

**}**

**class Program : Soyut\_Sınıf**

**{**

**public override void Soyut\_Metot()**

**{**

**Console.WriteLine("Soyut Metot");**