

2. Araştırma Konusu(10 Puan)

Java'da Çoklu katılımın mümkün olmadığını belirttik. Bunun nedenini araştırın, olması durumunda nasıl bir problem oluşacağını kod yazarak anlatın.

Karmaşıklığı azaltmak ve dili basitleştirmek için, java'da birden fazla kalıtım desteklenmemektedir.

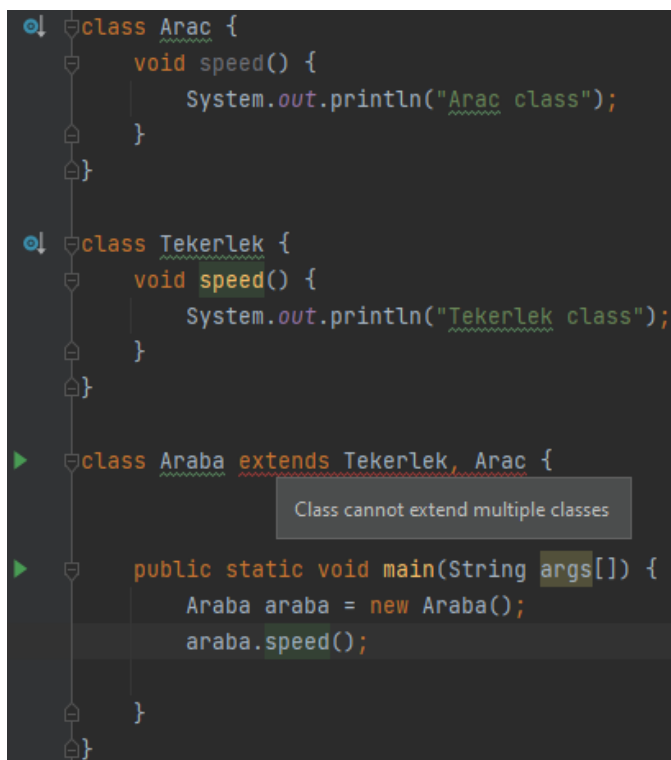
```
package com.hepsiemlak.hw1.inheritance2;

class Arac {
    void speed() {
        System.out.println("Arac class");
    }
}

class Tekerlek {
    void speed() {
        System.out.println("Tekerlek class");
    }
}

class Araba extends Tekerlek, Arac {

    public static void main(String args[]) {
        Araba araba = new Araba();
        araba.speed();
    }
}
```



```
class Arac {
    void speed() {
        System.out.println("Arac class");
    }
}

class Tekerlek {
    void speed() {
        System.out.println("Tekerlek class");
    }
}

class Araba extends Tekerlek, Arac {
    public static void main(String args[]) {
        Araba araba = new Araba();
        araba.speed();
    }
}
```

Class cannot extend multiple classes

Olması durumunda ise Diamon Problem ortaya çıkmaktadır.

Çoklu kalıtımı bu şekilde halledebiliriz.

```
package com.hepsiemlak.hw1.inheritance2;

interface Arac {

    void brand();

    default void speed() {
        System.out.println("Arac");
    }

}

interface Tekerlek {
    default void speed() {
        System.out.println("Tekerlek");
    }

}

class Araba implements Tekerlek, Arac {

    @Override
    public void brand() {

    }

    @Override
    public void speed() {
        Tekerlek.super.speed(); //tekerleğin speed i gibi hareket edecektir.
    }

    public static void main(String args[]) {
        Arac arac = new Araba();
        arac.speed();
    }

}
```

Hangi diller bu duruma izin veriyor ve bu durumu nasıl sağlıyor?

C++, Python, Scala

```
class arac {
    public:
        arac();
};
arac::arac()
{
    cout << "Bu bir arac" << endl;
}
class tekerlek
{
    public:
        tekerlek();
};
tekerlek::tekerlek()
{
    cout << "Bu arac 4 tekerlekli" << endl;
}
class araba :public arac, public tekerlek
{
};

int main()
{
    araba obj;
    system("Pause");
    return 0;
}
```

Araba sınıfı hem arac hemde tekerlek sınıfından miras almış oldu. ikisindeki değerleride kullanabildi.

Bu durumu Java'da OOP prensipleriyle sağlayabilir miyiz? Evet ise nasıl?

(Anlatımımız içinde örnek kodların bulunması sade ve net anlaşılır olması alacağımız puanı etkileyecektir)

Evet sağlayabiliriz ki OOP'un temelinde aslında bu karmaşıklığı gidermektir. Interface'ler ile bu durumu halledebiliyoruz.

Çoklu Kalıtım gerektiren çoğu yerde interface kullanılarak yapılan Multiple Kalıtım gerekli çözümleri sağlıyor. Sağlamadığı durumlarda da encapsulation, delegation gibi kavramlardan yararlanabiliyoruz.