

# Programlama Dillerinin Prensipleri

Hafta 9 - Nesne Yönelimli Programlama - 2

Dr. Öğr. Üyesi M. Fatih ADAK

# İçerik

- Arayüzler
- Çoklu Sınıf ve Arayüz Kalıtları
- namespace ve Paketler
- Soyut Sınıflar
- Kalıtım Hiyerarşisi
- Object Veri Türü
- Generic - Şablon Yapılar
- Friend Erişim Niteleyicisi

# Arayüz Tanımı

- Arayüz kullanımı aslında bir sözleşme imzalamaktır.
- Bir arayüzden kalıtım alan bir sınıf o arayüzü gerçekleştireceğini vadediyor demektir.
- Arayüzler yardımıyla kullanılabilirlik dış dünyaya rahatlıkla açılabilir.
- Java ve C#'ta arayüzler alan içeremez sadece public metot tanımı içerebilirler.
- Java ve C# dilinde direk desteği bulunan arayüzler, C++ dilinde soyut sınıflar kullanılarak gerçekleştirilir.

```
public interface MuzikCalar {  
    public void Oynat(String dosyaTuru,String dosyaAdi);  
}
```



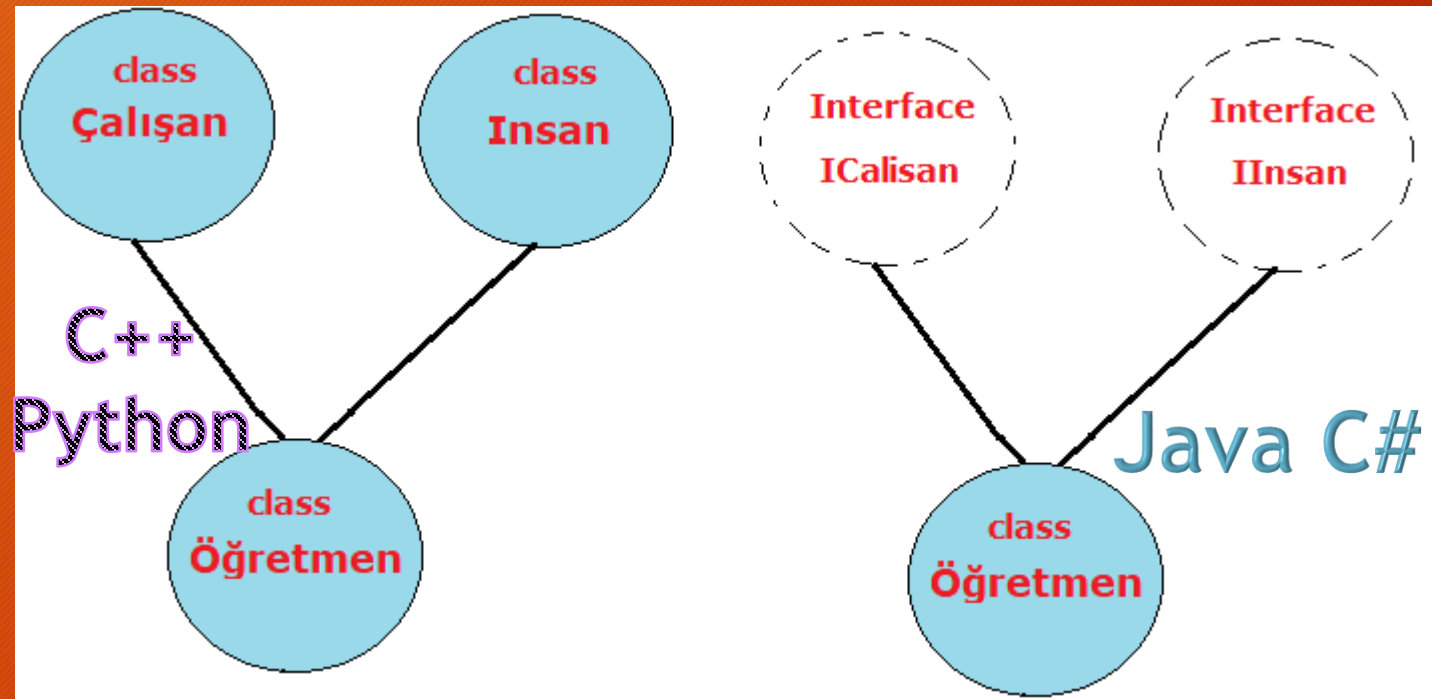
# Arayüzler

```
interface IEmail{
    public void setGonderen(String gonderen);
    public void setAlan(String alan);
    public void setMesaj(IIcerik mesaj);
}
interface IIcerik{
    public String StringOlarakIcerik();
}
class Email implements IEmail{
    public void setGonderen(String gonderen){ /*Göndereni ayarlar*/ }
    public void setAlan(String alan){ /*Alanı ayarlar*/ }
    public void setMesaj(IIcerik mesaj){ /*Mail mesajını ayarlar*/ }
}
```

```
class Html implements IIcerik{
    public String StringOlarakIcerik(){
        // HTML'i String olarak ifade et.
    }
}
```

# Arayüzler

- Çoklu sınıf kalıtımı izin verilmeyen dillerde çözüm olarak arayüz bulunmalıdır. Çoklu arayüz kalıtımı yapılmalıdır.



# Çoklu Sınıf Kalıtımı

```
class Sekil
{
    protected double genislik;
    protected double yukseklik;
    1 başvuru
    public Sekil(double genislik,double yukseklik)
    {
        this.genislik = genislik;
        this.yukseklik = yukseklik;
    }
}
```

C#



base

```
class Daire : Sekil
{
    0 başvuru
    public Daire(double g,double y) : base(g,y)
    {
    }
}
```

```
class İnsan{
    protected:
        double boy;
    public:
        İnsan(double boy) { this->boy = boy; }
};
class Calisan{
    protected:
        int Id;
    public:
        Calisan(int Id){ this->Id = Id; }
};
class Ogretmen : public İnsan, public Calisan{
    public:
        Ogretmen(double boy,int Id) : İnsan(boy),Calisan(Id){
        }
};
```

C++

```
class İnsan:
    ...
class Calisan:
    ...
class Ogretmen(Insan, Calisan):
    super()
```

Python

Yazılan sıraya göre çağırır  
Önce İnsan sonra Calisan sınıfın  
Kurucusu çağrılır.

# namespace (isim uzayları) ve Paketler

- Bir arayüzden farklı olarak isim uzayları kodu organize etmek, derli toplu bir halde bulunmasını sağlar. C++ ve C# dillerinde desteklenir.
- Java dilindeki karşılığı paketlerdir ve klasör mantığı ile saklanır.

```
namespace A{
    double Hesapla() { return 0; }
}
namespace B{
    double Hesapla() { return 1; }
}
int main() {
    cout<<A::Hesapla()<<endl; 0 yazar
    cout<<B::Hesapla()<<endl; 1 yazar
    return 0;
}
```

**C++**



# Soyut Sınıflar

- İçerisinde tamamlanmamış alanlar içeren sınıflara denir.
- Arayüzden farkı tanımlanmış alanlar da içerebilirler.
- Tanımlanmamış alanlar kalıtım yolu ile tanımlanır.
- Eksik tanım içerdikleri için soyut sınıflardan nesne türetilemez.



# Soyut Sınıflar

## C++

```
class Canli{
    private:
        int yas;
    public:
        virtual void YemekYe () = 0;
        int Yas ()const { return yas; }
};

class Kedi : public Canli{
    public:
        void YemekYe () {
        }
};
```

## C#

```
public abstract class Canli{
    public int yas { get; }
    public abstract void YemekYe();
}

public class Kedi : Canli
{
    public override void YemekYe()
    {
    }
}
```

# Kalıtım Hiyerarşisi

- Java ve C#



# Object Veri Türü

- Herhangi bir türü içinde barındırabilecek şekilde tasarlanmış veri türüdür.
- Boxing ve Unboxing kullanımlarında büyük önem arz eder.
- Java dilinde en üstteki sınıf Object sınıfıdır ve bu sınıftan diğer bütün sınıflar gizli olarak kalıtım alır.
- Java ve C# dillerinde Object sınıfı bulunurken, C++ dilinde böyle bir sınıf yoktur.



# Generic - Şablon Yapılar

- C# ve C++ dillerinde aktif olarak kullanılan şablon yapılar Java diline sonradan dahil olmuştur.
- Bir dilde Object sınıfı ile şablon yapıları taklit edilebilir.

```
public class Koleksiyon<Tur>
{
    public Tur[] Elemanlar;

    public Koleksiyon(Tur []Elemanlar)
    {
        this.Elemanlar = Elemanlar;
    }

    public String EnBuyukEnKucukBirlestir()
    {
        return Elemanlar.Max().ToString() + Elemanlar.Min().ToString();
    }
}
```

C#

```
int[] tamsayilar = { 15,95,20,2,18,32};
Koleksiyon<int> koleksiyonsayilar = new Koleksiyon<int>(tamsayilar);
koleksiyonsayilar.EnBuyukEnKucukBirlestir(); 952 döner

char[] karakterler = { 'a', 'w', 'r', 't', 'k', 'p' };
Koleksiyon<char> koleksiyonkarakterler = new Koleksiyon<char>(karakterler);
koleksiyonkarakterler.EnBuyukEnKucukBirlestir(); wa döner
```

# Generic - Şablon Yapılar


C++

```
template <typename Tur>
class Koleksiyon{
public:
    Tur *Elemanlar;
    int uzunluk;
    Koleksiyon(Tur *Elemanlar,int uzunluk){
        this->Elemanlar = Elemanlar;
        this->uzunluk = uzunluk;
    }
    string EnBuyukEnKucukBirlestir(){
        return toString(*max_element(Elemanlar,Elemanlar+uzunluk))+
            toString(*min_element(Elemanlar,Elemanlar+uzunluk));
    }
    string toString(Tur t)
    {
        ostringstream ss;
        ss << t;
        return ss.str();
    }
};
```

# friend Erişim Niteleyicisi

```
class Sayi{
    private:
        double deger;
        void setDeger(double dgr){
            deger = dgr;
        }
    friend class Top;
};

class Top{
    private:
        Sayi *s;
    public:
        Top() {
            s = new Sayi();
            s->setDeger(100);
        }
        double getDeger() const{
            return s->deger;
        }
};
```



- Java dilinde **friend** diye bir terim yoktur. Fakat bu işlem, Sınıfları aynı pakete yerleştirerek kısmen gerçekleştirilebilir.
- Aynı paketteki sınıflar birbirlerinin protected ve erişim niteleyicisi olmayan elemanlarına erişebilirler.



# Kaynaklar

- Yumusak N., Adak M.F. *Programlama Dillerinin Prensipleri*. 1. Baskı, Seçkin Yayıncılık, 2018
- Sebesta, Robert W. *Concepts of programming languages*. 11 ed. Pearson Education Limited, 2016.
- Sethi, Ravi. *Programming languages: concepts and constructs*. Addison Wesley Longman Publishing Co., Inc., 1996.
- Watt, David A. *Programming language design concepts*. John Wiley & Sons, 2004.
- Malik, D. S., and Robert Burton. *Java programming: guided learning with early objects*. Course Technology Press, 2008.
- Waite, Mitchell, Stephen Prata, and Donald Martin. *C primer plus*. Sams, 1987.
- Hennessey, Wade L. *Common Lisp*. McGraw-Hill, Inc., 1989.
- Liang, Y. Daniel. *Introduction to Java programming: brief version*. pearson prentice hall, 2009.
- Yumusak N., Adak M.F. *C/C++ ile Veri Yapıları ve Çözümlü Uygulamalar*. 2. Baskı, Seçkin Yayıncılık, 2016