

12. Kalıtım

Kalıtım (inheritance) bir nesnenin başka bir nesnenin özelliklerini birebir devralmasıdır. Eğer tanımlamak istediğimiz bir nesne önceden tanımlanmış bir nesnenin de özelliklerine sahip ise kalıtım yolu ile bu nesnenin özelliklerini alıp ek özellikleri ilave ederek nesneyi oluşturabiliriz. Böylece nesne tanımlama işlemini kolaylaştırabilir ve kısaltabiliriz.

Kalıtım nesneye dayalı programlamanın temel özelliklerinden biridir. Bu sayede bir sınıfın içine başka bir sınıfı dahil etmiş oluruz. Özellikleri devralınan sınıf **Temel sınıf(base class)** olarak isimlendirilir. Temel sınıfın dahil edildiği sınıf **türemiş sınıf(Derived class)** olarak isimlendirilir. **Temel sınıf yaygın, genel, ortak özelliklerden oluşur.** Türemiş sınıflar temel sınıflara ilave olarak ek özellikler içerir.

Temel sınıfa Ebeveyn sınıf(Parent class)

Türemiş sınıfa çocuk sınıf(child class) gibi isimlerde verilmektedir.

Bir türemiş sınıf şu şekilde tanımlanır.

class Türemiş Sınıf : Erişim türü Temel Sınıf{

};

Cokgen, Dortgen ve Ucgen şeklinde üç sınıf tanımlamak isteyelim. Dortgen ve Ucgen sınıfları aslında birer cokgendir. Cokgen sınıfını temel sınıf olarak tanımlayıp diğer sınıfları bu sınıftan türetebiliriz.

class Dortgen : public Cokgen { ... }

class Ucgen : public Cokgen { ... }

Erişim türü **public**, **private** veya **protected** olabilir. Sıklıkla kullanılan **erişim** türü **public** dir.

Aşağıda verilen programda kalıtım yolu ile Cokgen sınıfından türetilmiş Dortgen ve Ucgen sınıfları görülmektedir.

```
#include <iostream>
```

```
#include <conio.h>
```

```
using namespace std;
```

```
class Cokgen {
```

```
protected:
```

```
int genislik, yukseklik;
```

```
public:
```

```
void DegerVer(int a,int b) {
```

```
    genislik=a; yukseklik=b;
```

```
}
```

```
};
```

```
class Dortgen : public Cokgen{ //Dortgen sınıfı Cokgen sınıfından turetiliyor
```

```
public:
```

```
int Alan () {return genislik*yukseklik;}
```

```
};
```

```
class Ucgen : public Cokgen{ //Ucgen sınıfı Cokgen sınıfından turetiliyor
```

```

public:
    int Alan () {return (genislik*yukseklık)/2;}
};

int main ()
{
    Dortgen D1;
    Ucgen U1;
    //D1 Dortgen nesnesinin genişlik ve yüksekliğine değeri atanıyor.
    D1.DegerVer(3,4);
    //U1 Ucgen nesnesinin genişlik ve yüksekliğine değeri atanıyor.
    U1.DegerVer(7,2);

    cout << "Dortgen Alan: " << D1.Alan()<<endl;
    cout << "Ucgen Alan: " << U1.Alan()<<endl;

    getch();
    return 0;
}

```

```

Dortgen Alan: 12
Ucgen Alan: 7

```

Cokgen sınıfı içindeki **protected** erişim denetleyicisi **private** erişim denetleyicisine çok benzemektedir ve Çokgen sınıfı için private anlamındadır. Dışardan erişime private de olduğu gibi kapalıdır. Ancak Protected altında tanımlanan değişkenlere türemiş (Dortgen, Ucgen) sınıflar erişip kullanabilir. Türemiş sınıflar temel sınıfın private üyelerine erişemezler. Aşağıdaki tabloda erişim denetleyicilerin kullanımları özetlenmiştir.

| Erişim | Public | Protected | Private |
|-------------------------|--------|-----------|---------|
| Sınıfın kendi üyeleri | Evet | Evet | Evet |
| Türemiş sınıfın üyeleri | Evet | Evet | Hayır |
| Üye olmayanlar | Evet | Hayır | Hayır |

Başka bir örnek

```

#include <iostream>
#include <conio.h>
using namespace std;

class Cokgen {
public:
    int genislik, yukseklik;
public:
    void DegerVer(int a,int b) {
        genislik=a; yukseklik=b;
    }
}

```

```

};

class Dortgen : public Cokgen{ //Dortgen sınıfı Cokgen sınıfından turetiliyor
public:
    int Alan () {return genislik*yukseklik;}
};

class Ucgen : public Cokgen{ //Ucgen sınıfı Cokgen sınıfından turetiliyor
public:
    int Alan () {return (genislik*yukseklik)/2;}
};

int main ()
{
    Dortgen D1;
    Ucgen U1;
    //D1 Dortgen nesnesinin genişlik ve yüksekliğine değeri atanıyor.
    D1.DegerVer(3,4);
    //U1 Ucgen nesnesinin genişlik ve yüksekliğine değeri atanıyor.
    U1.DegerVer(7,2);

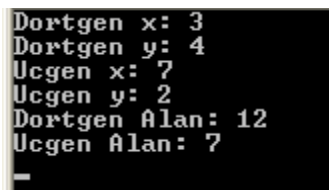
    cout << "Dortgen x: " << D1.genislik<<endl;
    cout << "Dortgen y: " << D1.yukseklik<<endl;

    cout << "Ucgen x: " << U1.genislik<<endl;
    cout << "Ucgen y: " << U1.yukseklik<<endl;

    cout << "Dortgen Alan: " << D1.Alan()<<endl;
    cout << "Ucgen Alan: " << U1.Alan()<<endl;

    getch();
    return 0;
}

```



```

Dortgen x: 3
Dortgen y: 4
Ucgen x: 7
Ucgen y: 2
Dortgen Alan: 12
Ucgen Alan: 7

```

Başka bir örnek:

```

#include <iostream>
#include <conio.h>
using namespace std;

```

```

class Cokgen {
protected:
    int genislik, yukseklik;
public:
    void Goster (void);
    void DegerVer(int a,int b) {
        genislik=a; yukseklik=b;
    }
};

class Dortgen : public Cokgen{ //Dortgen sınıfı Cokgen sınıfından turetiliyor
public:
    int Alan () {return genislik*yukseklik;}
};

class Ucgen : public Cokgen{ //Ucgen sınıfı Cokgen sınıfından turetiliyor
public:
    int Alan () {return (genislik*yukseklik)/2;}
};

void Cokgen::Goster (void) {
    cout << " genislik : " << genislik<<endl;
    cout << " yukseklik : " << yukseklik<<endl;
}

int main ()
{
    Dortgen D1;
    Ucgen U1;
    //D1 Dortgen nesnesinin genişlik ve yüksekliğine değeri atanıyor.
    D1.DegerVer(3,4);
    //U1 Ucgen nesnesinin genişlik ve yüksekliğine değeri atanıyor.
    U1.DegerVer(7,2);

    D1.Goster();
    U1.Goster();

    cout << "Dortgen Alan: " << D1.Alan()<<endl;
    cout << "Ucgen Alan: " << U1.Alan()<<endl;

    getch();
    return 0;
}

```

```

genislik : 3
yukseklik : 4
genislik : 7
yukseklik : 2
Dortgen Alan: 12
Ucgen Alan: 7

```

12.1 Public, protected ve private kalıtım

Bir temel sınıftan bir sınıf türetildiğinde temel sınıf türemiş sınıfa public, protected veya private olarak dahil edilebilir. **Prprotected** ve **private** kalıtımın kullanımı yaygın değildir ve kullanımı dikkat gerektirir.

Aşağıda verilen örnekte A sınıfı B sınıfından **public erişim türü** ile türetilmiştir. Buna public kalıtım denmektedir.

```
class A : public B { ... }
```

Bu tür kalıtımda

B **temel sınıfının public** üyeleri A **türemiş sınıfının public üyeleri** olurlar. Aynı şekilde

B **temel sınıfının protected** üyeleri A **türemiş sınıfının protected üyeleri** olurlar.

B **temel sınıfının private** üyeleri A **türemiş sınıfı için** gizlidir ve erişilemezdirler

Aşağıda verilen örnekte C sınıfı D sınıfından **protected erişim türü** ile türetilmiştir. Buna protected kalıtım denmektedir.

```
class C : protected D { ... }
```

Bu tür kalıtımda

D **temel sınıfının public** üyeleri C **türemiş sınıfının *protected üyeleri*** olurlar. Aynı şekilde

D **temel sınıfının protected** üyeleri C **türemiş sınıfının *protected üyeleri*** olurlar. Aynı şekilde

D **temel sınıfının private** üyeleri C **türemiş sınıfı için** gizlidir ve erişilemezdirler

Aşağıda verilen örnekte E sınıfı F sınıfından **private erişim türü** ile türetilmiştir. Buna private kalıtım denmektedir.

```
class E : private F { ... }
```

Bu tür kalıtımda

F **temel sınıfının public** üyeleri E **türemiş sınıfının *private üyeleri*** olurlar. Aynı şekilde

F **temel sınıfının protected** üyeleri E **türemiş sınıfının *private üyeleri*** olurlar. Aynı şekilde

F **temel sınıfının private** üyeleri E **türemiş sınıfı için** gizlidir ve erişilemezdirler

```
#include <iostream>
```

```
#include <conio.h>
```

```
using namespace std;
```

```
class A {  
    int x1;  
    protected:  
    int x,y;  
    public:  
        A(){x=5; y=10;}  
    void Goster1(void) {cout << "Deger = " << x<<" "<< y <<endl;}  
    void DegerVer(int a,int b) {  
        x=a; y=b;  
    }  
};
```

```
class B : public A{  
    public:  
        void Goster2(void) {cout << "Deger = " << x<<" "<< y <<endl;}  
};
```

```

class C : protected A{
public:
    void Goster3(void) {cout << "Deger = " << x<<" "<< y<<endl;}
};

class D : private A{
public:
    void Goster4(void) {cout << "Deger = " << x<<" "<< y <<endl;}
};

int main ()
{
    A nesne1;
    B nesne2;
    C nesne3;
    D nesne4;

    nesne1.DegerVer(1,2);
    nesne2.DegerVer(3,4);
    //Alttaki satırda Hata DegerVer() fonksiyonu C sınıfı için protected dir dışardan erişilemez
    //Bu satırı siliniz.
    nesne3.DegerVer(5,6);

    //Alttaki satırda Hata DegerVer() fonksiyonu D sınıfı için private dir dışardan erişilemez
    //Bu satırı siliniz.
    nesne4.DegerVer(7,8); //Hata DegerVer() fonksiyonu C sınıfı için dışardan erişilemez

    nesne1.Goster1();
    nesne2.Goster2();
    nesne3.Goster3();
    nesne4.Goster4();

    getch();
    return 0;
}

```