

OOP

Vererbung

abgeleitete Klassen erben Attribute und Methoden der Basisklasse

```
class CTier{
public:
    void sagwas() { cout << "Ich bin ein Tier." << endl; }
};

class CHund:public CTier{
};

int main(){
    CTier tier;
    tier.sagwas(); // Ich bin ein Tier.

    CHund hund;
    hund.sagwas(); // Ich bin ein Tier.

    return 0;
}
```

Polymorphie

Geerbte Methoden können überschrieben werden. Beim Aufruf einer Methode ist der **Typ der Instanz** entscheidend.

```
class CTier{
public:
    void sagwas() { cout << "Ich bin ein Tier." << endl; }
};

class CHund:public CTier{
public:
    void sagwas() { cout << "Ich bin ein Hund." << endl; }
};

int main(){
    CTier tier;
    tier.sagwas(); // Ich bin ein Tier.

    CHund hund;
    hund.sagwas(); // Ich bin ein Hund.

    return 0;
}
```

Zeiger auf Instanzen und virtuelle Funktionen

Beim Aufruf einer **nicht** virtuellen Methode ist der **Typ des Zeigers** entscheidend.

```
class CTier{
public:
    void sagwas(){
        cout << "Ich bin ein Tier." << endl;
    }
};

class CHund:public CTier{
public:
    void sagwas(){
        cout << "Ich bin ein Hund." << endl;
    }
};

int main(){
    CTier* pTier1 = new CTier;
    pTier1->sagwas(); // Ich bin ein Tier.

    CHund* pHund = new CHund;
    pHund->sagwas(); // Ich bin ein Hund.

    CTier* pTier2 = new CHund;
    pTier2->sagwas(); // Ich bin ein Tier.

    return 0;
}
```

Beim Aufruf einer **virtuellen** Methode ist der **Typ der Instanz** entscheidend.

```
class CTier{
public:
    virtual void sagwas(){
        cout << "Ich bin ein Tier." << endl;
    }
};

class CHund:public CTier{
public:
    void sagwas(){
        cout << "Ich bin ein Hund" << endl;
    }
};

int main(){
    CTier* pTier1 = new CTier;
    pTier1->sagwas(); // Ich bin ein Tier.

    CHund* pHund = new CHund;
    pHund->sagwas(); // Ich bin ein Hund.

    CTier* pTier2 = new CHund;
    pTier2->sagwas(); // Ich bin ein Hund.

    return 0;
}
```