继承

继承语法

```
// 父类
class Baseclass {
    ...
};
// 子类 -> 公有继承自父类
class Subclass : public Baseclass {
    ...
};
```

- 父类所拥有的成员函数子类也同样会拥有
- 父类是个抽象的概念，一般不会创建对象。子类是具象化的，可以为其创建对象
- 关系（**子类-派生类**与**父类-基类**）
    1. 子类 -> 父类
    2. 派生类 -> 基类

继承权限

public继承

父类	子类作用域内	子类作用域外
public	√	√
protected	√	×
private	×	×

protected继承

父类	子类作用域内	子类作用域外
public	√	×
protected	√	×
private	×	×

private继承

父类	子类作用域内	子类作用域外
public	√	×
protected	√	×
private	×	×

继承内存分布

- 先存放父类的成员，后存放子类的成员
- 父类的私有成员存在于子类的内存中，但是对于子类是不可见的
- 权限是编译期控制，在运行期可以借助指针进行修改

父类和子类的指针转换

- 子类指针转父类是安全的
- 父类指针转子类是不安全的——会发生访问越界
- 只有一种情况父类指针转子类是安全的——即父类指针本身所指向的就是子类对象

```
class Baseclass {
    ...
};
class Subclass : public Baseclass {
    ...
};

Subclass sc;
Baseclass *ptr_base = &sc;
Subclass *ptr_sub = (Subclass *)ptr_base;
```

- 子类可以赋值给父类引用

隐藏数据

- 子类有成员与父类成员名字相同，则父类的成员会被隐藏
- 若指定访问父类的成员，则需要加父类的作用域

```
class Baseclass {
public:
    void foo()
    {
        ...
    }
};
class Subclass : public Baseclass {
    public:
    void foo()
    {
        ...
```

```
    }
};

Subclass sc;
sc.foo();                // 调用子类的foo成员函数
sc.Baseclass::foo();     // 调用父类的foo成员函数
```

构造和析构的顺序

构造的顺序

1. 父类对象
2. 成员对象
3. 子类对象

- 初始化列表中的顺序不影响构造顺序
- 若子类没有构造，父类有构造，编译器会为子类生成默认构造

析构的顺序

1. 子类对象
2. 成员对象
3. 父类对象