

The slide features a red diagonal banner in the top-left corner containing the Peking University logo, which includes the university's name in English ('PEKING UNIVERSITY') and Chinese characters. Below the banner is a grayscale image of a traditional Chinese building facade with the characters '北京大学' (Peking University) inscribed on a plaque.

从客观事物抽象出类的例子

郭 炜 刘家瑛



例: 客观事物→类

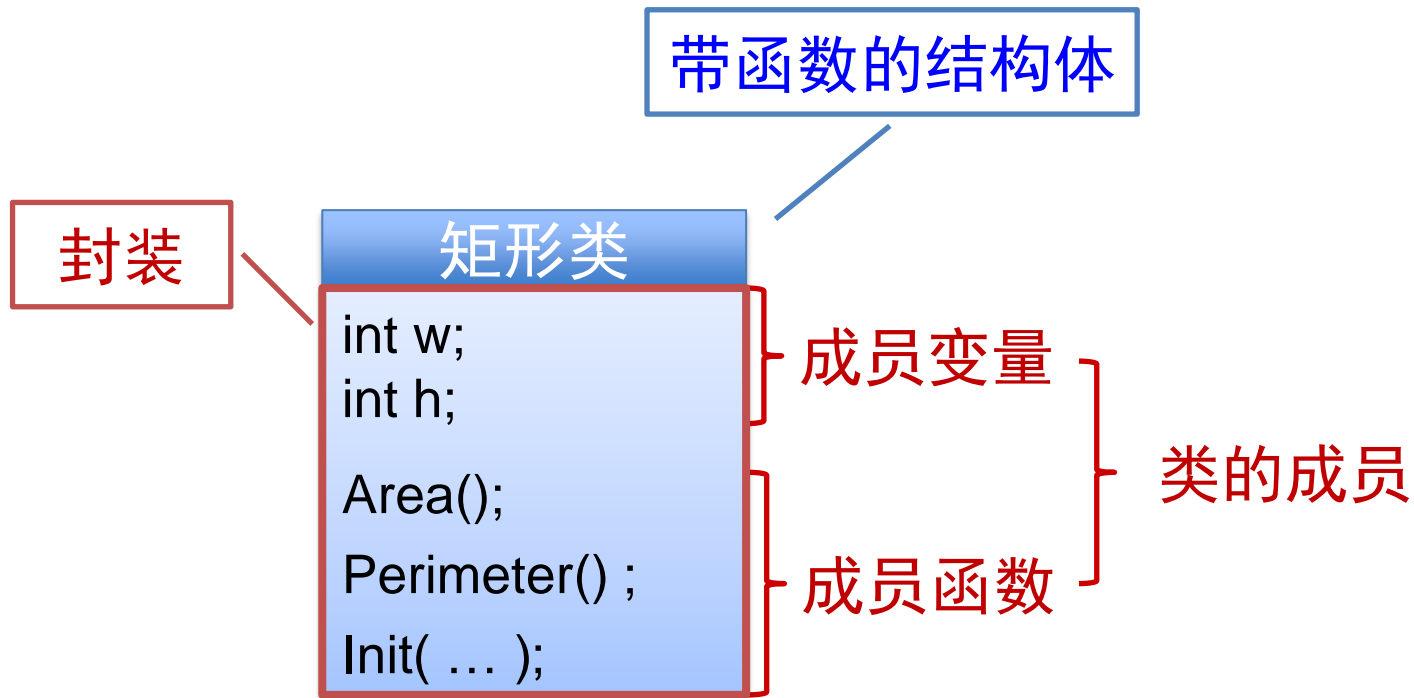
- 写一个程序, 输入矩形的宽和高, 输出面积和周长



- 矩形的属性 – 宽和高
 - 两个变量, 分别代表宽和高
- 对矩形的操作
 - 设置宽和高
 - 计算面积
 - 计算周长



例: 客观事物→类





```
class CRectangle {  
    public:  
        int w, h;  
  
        void Init( int w_, int h_ ) {  
            w = w_; h = h_;  
        }  
  
        int Area() {  
            return w * h;  
        }  
  
        int Perimeter() {  
            return 2 * ( w + h );  
        }  
}; //必须有分号
```



```
int main() {  
    int w, h;  
    CRectangle r; //r是一个对象  
    cin >> w >> h;  
    r.Init(w, h);  
    cout << r.Area() << endl << r. Perimeter();  
    return 0;  
}
```

- 类定义的变量 → 类的实例 → “对象”



对象的内存分配

对象的内存空间

- 对象的大小 = 所有成员变量的大小之和
- E.g. CRectangle类的对象, `sizeof(CRectangle) = 8`

每个对象各有自己的存储空间

- 一个对象的某个成员变量被改变, 不会影响到其他的对象



对象间的运算

- 对象之间可以用 '=' 进行赋值
- 不能用 '==', '!=', '>', '<', '>=', '<=' 进行比较
 - 除非这些运算符经过了“重载”



访问类的成员变量和成员函数

用法1: **对象名.成员名**

```
CRectangle r1, r2;
```

```
r1.w = 5;
```

```
r2.Init(3,4);
```




访问类的成员变量和成员函数

用法2: 指针->成员名

```
CRectangle r1, r2;
```

```
CRectangle * p1 = & r1;
```

```
CRectangle * p2 = & r2;
```

```
p1->w = 5;
```

```
p2->Init(3,4); //Init作用在p2指向的对象上
```



访问类的成员变量和成员函数

用法3: 引用名.成员名

```
CRectangle r2;
```

```
CRectangle & rr = r2;
```

```
rr.w = 5;
```

```
rr.Init(3,4);    //rr的值变了, r2的值也变
```



▲ 另一种输出结果的方式

```
void PrintRectangle(CRectangle & r) {  
    cout << r.Area() << ", "<< r.Perimeter();  
}  
  
CRectangle r3;  
r3.Init(3,4);  
PrintRectangle(r3);
```



类的成员函数的另一种写法

- 成员函数体和类的定义分开写

```
class CRectangle
```

```
{
```

```
    public:
```

```
        int w, h;
```

```
        int Area();    //成员函数仅在此处声明
```

```
        int Perimeter() ;
```

```
        void Init( int w_, int h_ );
```

```
};
```



类的成员函数的另一种写法

```
int CRectangle::Area() {  
    return w * h;  
}  
int CRectangle::Perimeter() {  
    return 2 * ( w + h );  
}  
void CRectangle::Init( int w_, int h_ ) {  
    w = w_; h = h_;  
}
```

调用通过: 对象 / 对象的指针 / 对象的引用