



# C++

## 第五讲 类和对象（一）- 补充

基础课教研室C++ 课程组

1

## 类的作用域

概念

名字查找

# 类的作用域：概念

❖ **概念**：每个类都定义了自己的新作用域和唯一的类型，类的声明和定义都属于这个类的作用域！

❖ **[注意]**

➤ 不同的两个类可以具有相同的成员！

```
class A
{
public:
    void display()
    {... ..}
private:
    int ival;
};
```

```
class B
{
public:
    void display()
    {... ..}
private:
    int ival;
};
```

# 类的作用域：概念

- 成员函数的形参表和函数体处于类作用域中。

```
class Goods
{
public:
    typedef double Money;
    void SetPrice(Money value) {
        price = value;
    }
    Money GetPrice() {
        return price;
    }

private:
    Money price;
};
```

不必写成  
Goods::Money  
Goods::price

# 类的作用域：概念

- 成员函数的返回类型不一定处于类作用域中。

```
class Goods {  
public:  
    typedef double Money;  
    void SetPrice(Money value);  
    Money GetPrice();  
private:  
    Money price;  
};  
void Goods::SetPrice(Money value) {  
    price = value;  
}  
Goods::Money Goods::GetPrice() {  
    return price;  
}
```

# 类的作用域：概念

- 类的作用域之外访问类成员，只能通过类的对象，类的指针，或者类的引用访问，而不能直接访问。

```
class A
{
public:
    void display() {... ..}
private:
    int val;
};
void main(void)
{
    A obj;
    obj.display(); // Right
    display();     // Error
}
```

1

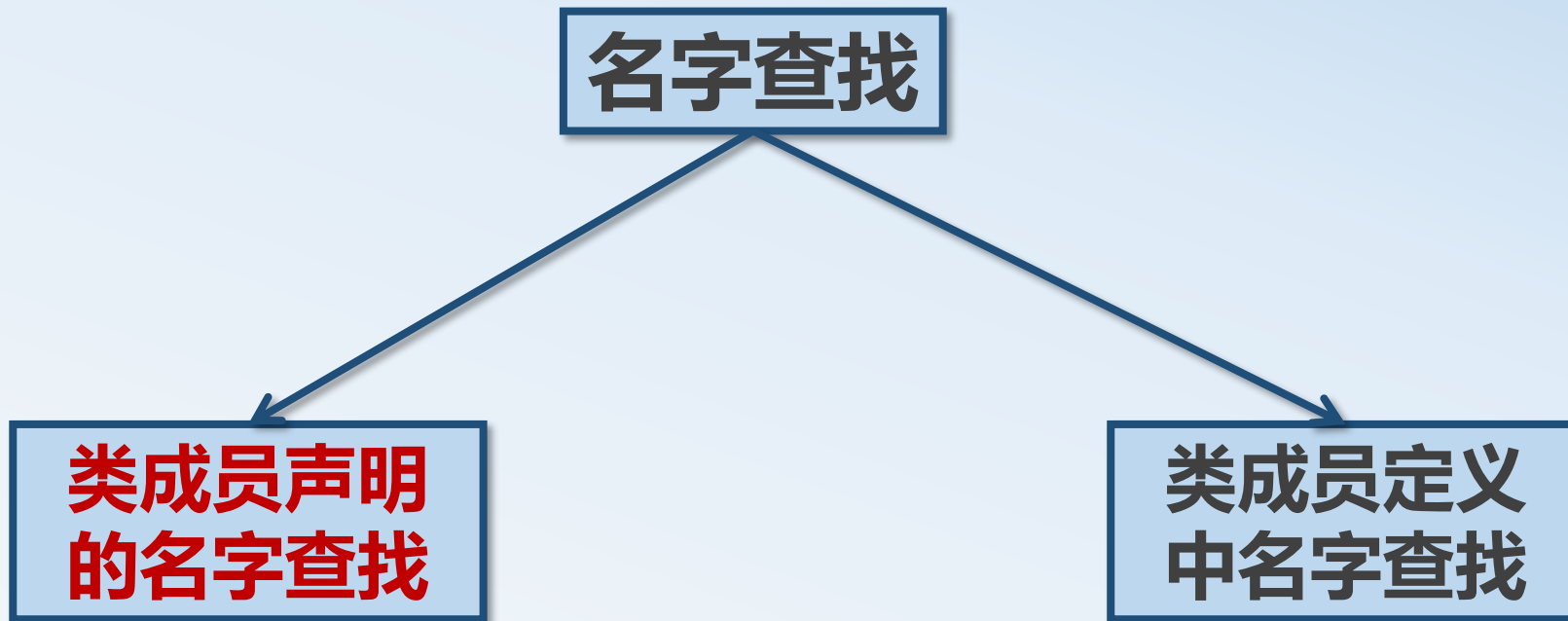
## 类的作用域

概念

名字查找

# 类的作用域：名字查找

❖ **概念**：在作用域中，寻找给定名字的过程。





# 类成员声明时的名字查找

- ① 在名字出现之前的类的作用域中检查
- ② 检查全局作用域

```
typedef double Money;  
... ..  
class Account  
{  
public:  ↕  
    Money balance();  
private:  
    Money bal;  
    ... ..  
};  
Money balance()  
{ return bal;}
```

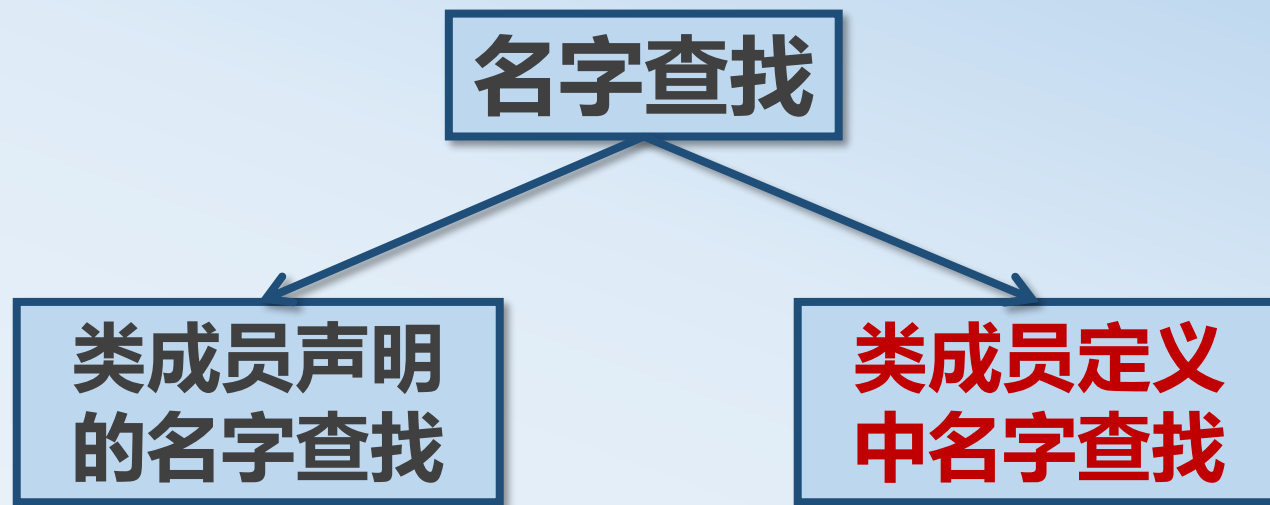
# 类成员声明时的名字查找

## [注意]

- 必须先定义类型名字，才能将他们用作成员类型
- 编译器按照成员声明在类中出现的次序处理名字，一旦一个名字被用作类型名，该名字就不能被重复定义

```
typedef double Money;  
class Account  
{  
public:  
    Money balance(){ return bal; }  
private:  
    typedef long double Money;  
    Money bal;  
    ...  
};
```

# 类成员定义中的名字查找




- ① 从成员函数体开始到名字出现之前的位置查找；
- ② 在类体内查找，然后在成员函数出现之前的作用域中查找；
- ③ 在成员函数出现之前的全局作用域中查找。

# 类成员定义中的名字查找

## ① 在成员函数作用域中查找（只考虑名字出现之前）

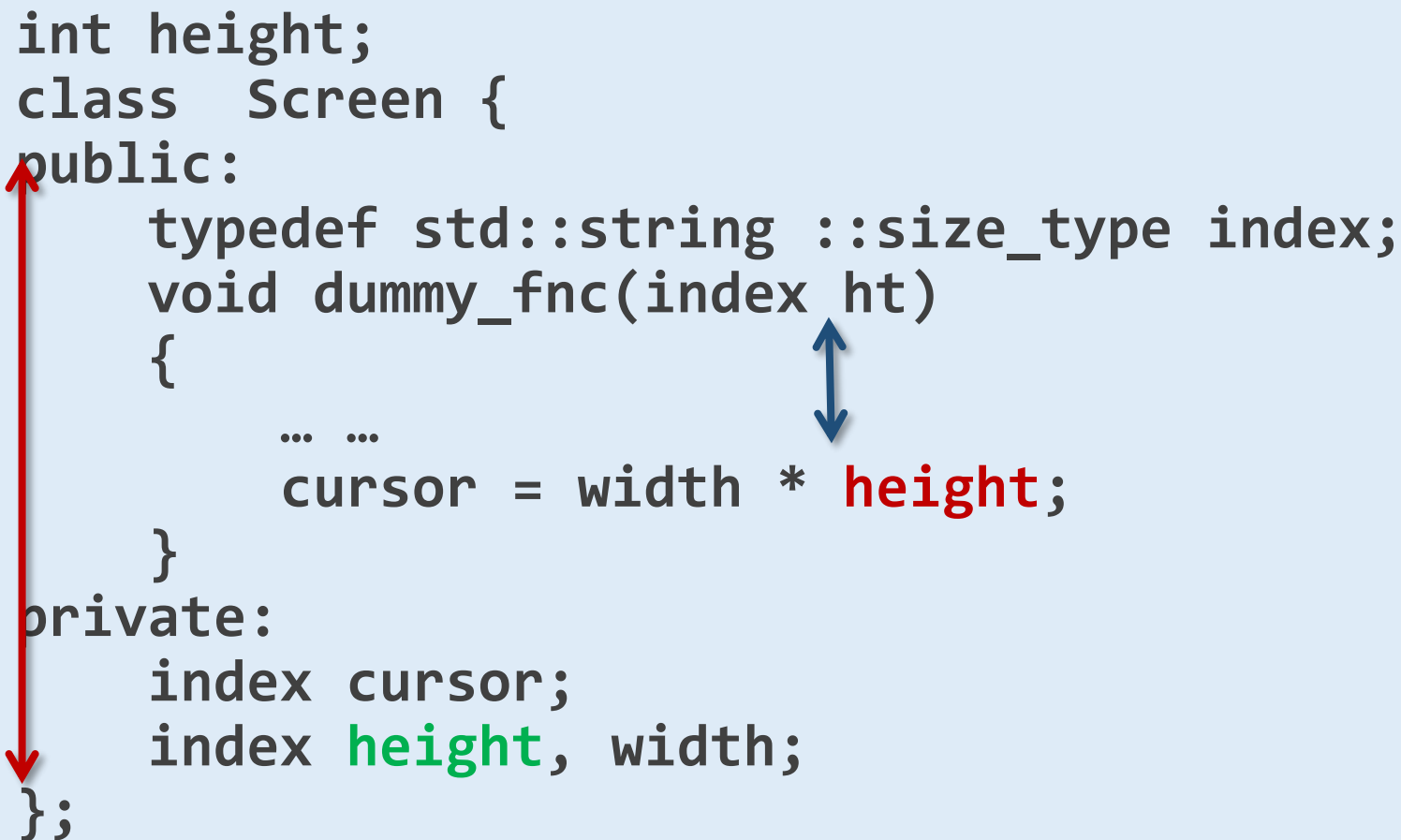
```
int height;
class Screen {
public:
    typedef std::string::size_type index;
    void dummy_fnc(index height)
    {
        ... ..
        cursor = width * height;
        ... ..
    }
private:
    index cursor;
    index height, width;
};
```



# 类成员定义中的名字查找

- ② 在类体中查找，然后在成员函数出现之前的作用域中查找

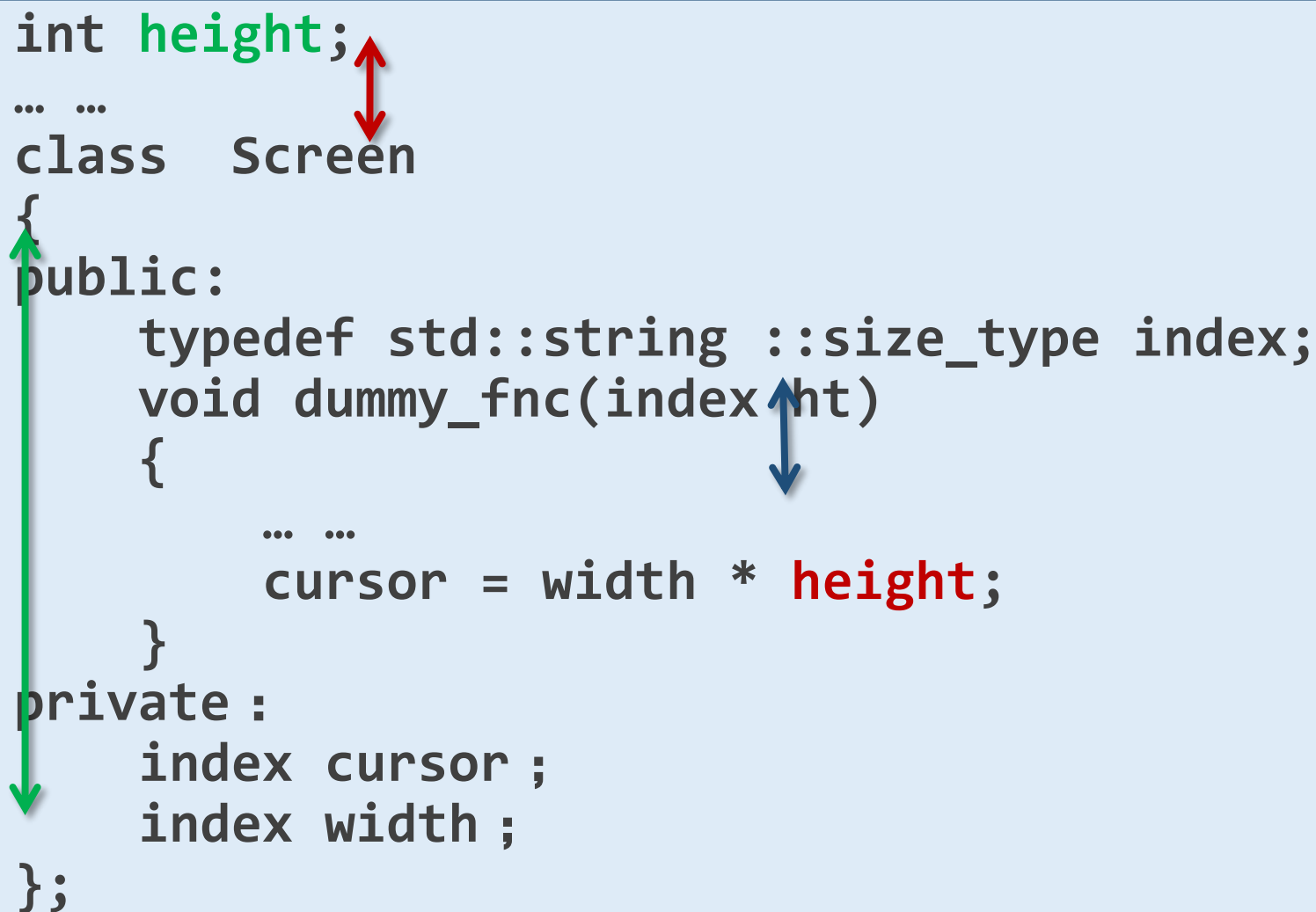
```
int height;
class Screen {
public:
    typedef std::string ::size_type index;
    void dummy_fnc(index ht)
    {
        ... ..
        cursor = width * height;
    }
private:
    index cursor;
    index height, width;
};
```



# 类成员定义中的名字查找

## ③ 在成员函数出现之前的全局作用域中查找

```
int height;
... ..
class Screen
{
public:
    typedef std::string ::size_type index;
    void dummy_fnc(index ht)
    {
        ... ..
        cursor = width * height;
    }
private :
    index cursor ;
    index width ;
};
```





THANKS

