多重继承

class [类名] : [[继承权限] [类名] , [继承权限] [类名], ...]

`class A : public B, public C`

示例代码:

```
// 1. 多重继承无虚函数.cpp : 定义控制台应用程序的入口点。
//

#include "stdafx.h"
#include <iostream>
using namespace std;

// 床类
class CBed {
public:
    CBed() {
        printf("CBed::CBed()\r\n");
        m_nBed = 1;
    }

    ~CBed() {
        printf("CBed::~CBed()\r\n");
        m_nBed = 0;
    }

    void sleep() {
        printf("CBed::sleep()\r\n");
    }

private:
    int m_nBed;
};

// 沙发类
class CSofa {
public:
    CSofa() {
        printf("CSofa::CSofa()\r\n");
        m_nSofa = 2;
```

```
    }

    ~CSofa() {
        printf("CSofa::~CSofa()\r\n");
        m_nSofa = 0;
    }

    void sit() {
        printf("CSofa::sit()\r\n");
    }

private:
    int m_nSofa;
};

// 沙发床多重继承，既继承了沙发的特点，又继承了床的特点。
class CSofaBed : public CSofa, public CBed {
public:
    CSofaBed() {
        printf("CSofaBed::CSofaBed()\r\n");
        m_nSofaBed = 3;
    }

private:
    int m_nSofaBed;
};

int main()
{
    CSofa sofa;
    CBed bed;
    CSofaBed sofabed;
    //对象大小
    cout << sizeof(CSofa) << endl; // 4
    cout << sizeof(CBed) << endl; // 4
    cout << sizeof(sofabed) << endl; // 12
}
/*
output:

CSofa::CSofa()
CBed::CBed()
```

```
CSofa::CSofa()
CBed::CBed()
CSofaBed::CSofaBed()
4
4
12
CBed::~CBed()
CSofa::~CSofa()
CBed::~CBed()
CSofa::~CSofa()
*/
```

对象大小

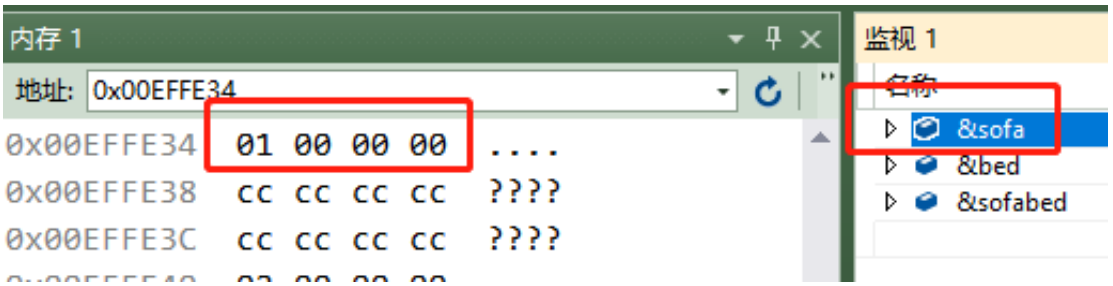
```
    //对象大小
    cout << sizeof(CSofa) << endl; // 4
    cout << sizeof(CBed) << endl; // 4
    cout << sizeof(sofabed) << endl; // 12
```

对象创建

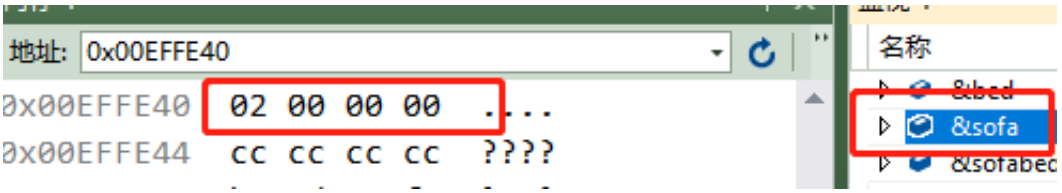
执行流程：
执行自身构造函数，如果有父类，先执行父类构造函数，执行完毕再执行自身构造函数。
这里有必要说明的一点就是，多继承，也是按顺序执行。
`class A : public B, public C`，假设都有构造函数，先执行B，再执行C，然后执行A。

对象内存

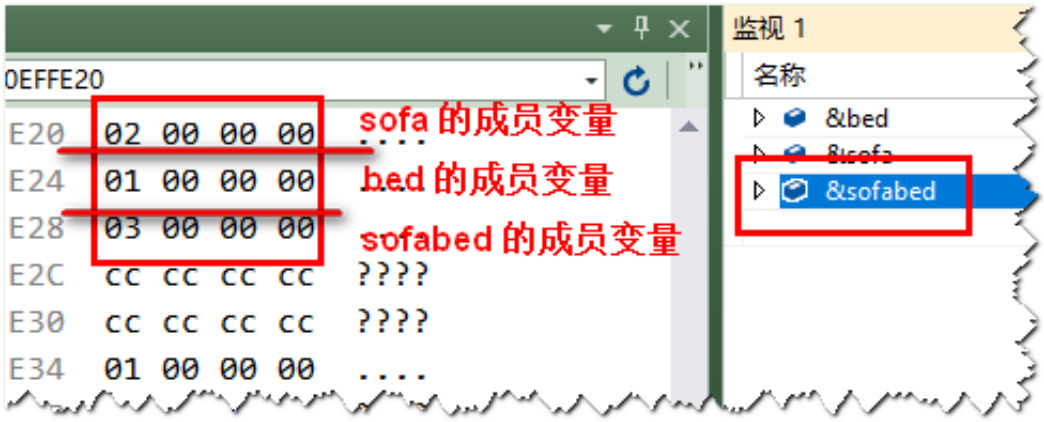
`CBed bed;`




```
CSofa sofa;
```




```
CSofaBed sofabed;
```



函数调用

都是直接调用。