

## 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 // 1. 多重继承无虚函数.cpp : 定义控制台应用程序的入口点。
2 //
3
4 #include "stdafx.h"
5 #include <iostream>
6 using namespace std;
7
8 // 床类
9 class CBed {
10 public:
11     CBed() {
12         printf("CBed::CBed()\r\n");
13         m_nBed = 1;
14     }
15
16     ~CBed() {
17         printf("CBed::~~CBed()\r\n");
18         m_nBed = 0;
19     }
20
21     void sleep() {
22         printf("CBed::sleep()\r\n");
23     }
24
25 private:
26     int m_nBed;
27 };
28
29 // 沙发类
30 class CSofa {
31 public:
32     CSofa() {
33         printf("CSofa::CSofa()\r\n");
34         m_nSofa = 2;
```

```

35     }
36
37     ~CSofa() {
38         printf("CSofa::~CSofa()\r\n");
39         m_nSofa = 0;
40     }
41
42     void sit() {
43         printf("CSofa::sit()\r\n");
44     }
45
46 private:
47     int m_nSofa;
48 };
49
50 // 沙发床多重继承，既继承了沙发的特点，又继承了床的特点。
51 class CSofaBed : public CSofa, public CBed {
52 public:
53     CSofaBed() {
54         printf("CSofaBed::CSofaBed()\r\n");
55         m_nSofaBed = 3;
56     }
57
58 private:
59     int m_nSofaBed;
60 };
61
62 int main()
63 {
64     CSofa sofa;
65     CBed bed;
66     CSofaBed sofabed;
67     //对象大小
68     cout << sizeof(CSofa) << endl; // 4
69     cout << sizeof(CBed) << endl; // 4
70     cout << sizeof(sofabed) << endl; // 12
71 }
72 /*
73 output:
74
75 CSofa::CSofa()
76 CBed::CBed()

```

```
77 CSofa::CSofa()
78 CBed::CBed()
79 CSofaBed::CSofaBed()
80 4
81 4
82 12
83 CBed::~~CBed()
84 CSofa::~~CSofa()
85 CBed::~~CBed()
86 CSofa::~~CSofa()
87 */
```

## 对象大小

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

## 对象创建

执行流程：

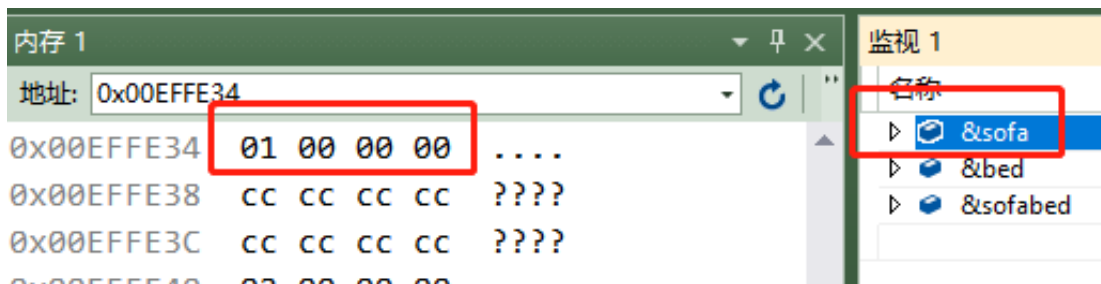
执行自身构造函数，如果有父类，先执行父类构造函数，执行完毕再执行自身构造函数。

这里有必要说明的一点就是，多继承，也是按顺序执行。

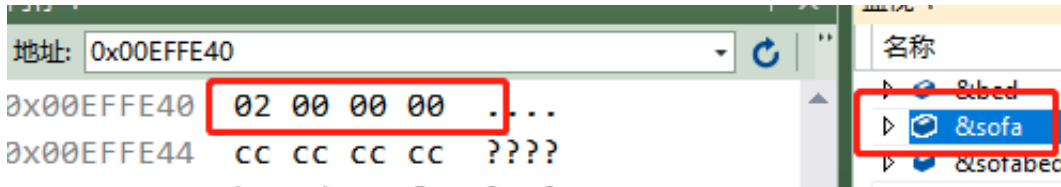
`class A : public B, public C`，假设都有构造函数，先执行B，再执行C，然后执行A。

## 对象内存

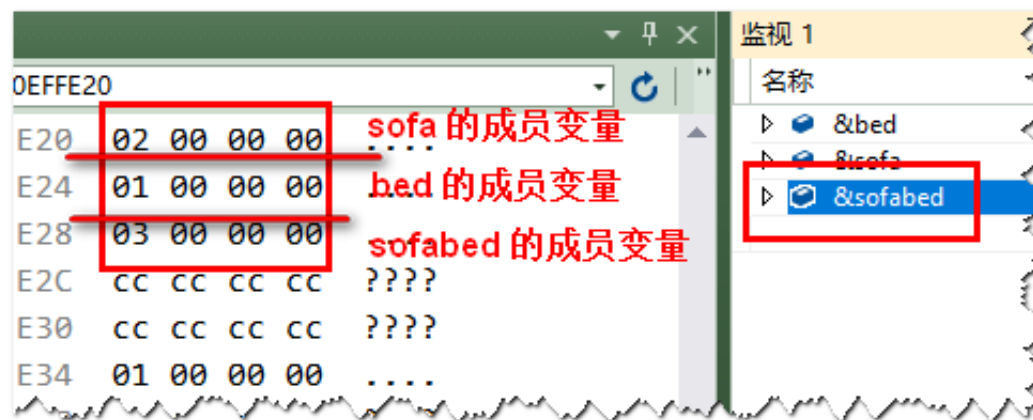
```
CBed bed;
```



CSofa sofa;



CSofaBed sofabed;



## 函数调用

都是直接调用。