示例代码:

```
// 5 3 菱形继承有虚继承有虚函数.cpp: 定义控制台应用程序的入口点。
  //
2
4 #include "stdafx.h"
5 #include <iostream>
   using namespace std;
7
8 class A {
9
   public:
10
       A() {
       _a = 1;
11
12
      };
13
    virtual void funA() {
14
           printf("A::funA()\r\n");
15
16
       }
17
18 public:
19
       int _a;
20 };
21
22 class B : virtual public A {
23 public:
24
       B() {
          b = 2;
25
26
       }
27
   virtual void funB() {
28
           printf("B::funB()\r\n");
29
       }
30
31 public:
       int b;
32
33 };
34
35 class C : virtual public A {
36 public:
       C() {
37
          _{c} = 3;
38
```

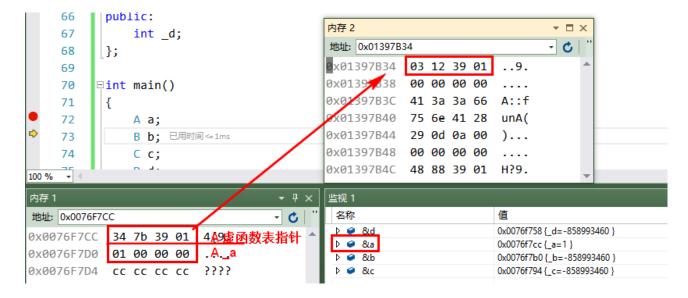
```
39
       }
40
       virtual void funC() {
41
            printf("C::funC()\r\n");
42
       }
43
   public:
44
       int _c;
45
46 };
47
48 class D : virtual public B, virtual public C {
   public:
49
       D() {
50
           _d = 4;
51
52
       }
53
       virtual void funA() {
54
55
            printf("D::funA()\r\n");
56
       }
57
58
       virtual void funB() {
            printf("D::funB()\r\n");
59
       }
60
61
       virtual void funC() {
62
63
            printf("D::funC()\r\n");
64
       }
65
66 public:
       int _d;
67
68 };
69
70 int main()
71 {
72
       A a;
73
       B b;
74
       C c;
75
       D d;
76
       d.B::_a = 10; // 通过指定是哪个类中的_a来消除二义性的目的
77
78
       cout << d.B::_a << endl;</pre>
79
       // 函数调用
80
```

```
81
        // 将派生类对象赋值给基类对象
        // a->b a->c
82
        a = b;
83
        a.funA(); // A::funA()
84
85
        a = c;
        a.funA(); // A::funA()
86
                    // b->d c->d
87
        b = d;
88
        b.funA(); // A::funA()
89
        b.funB(); // B::funB()
90
91
        c = d;
        c.funA(); // A::funA()
92
93
        c.funC(); // C::funC()
94
        printf("\r\n");
95
        // 将派生类指针赋值给基类指针
96
        // a->b a->c
97
        A* pa = &b;
98
        pa->funA(); // A::funA()
99
100
        A* pa2 = &c;
        pa2->funA(); // A::funA()
101
102
                      // b \rightarrow d c \rightarrow d
103
        B* pb = &d;
104
        pb->funA(); // D::funA()
105
        pb->funB(); // D::funB()
106
        B* pb2 = &d;
107
        pb2->funA(); // D::funA()
        pb2->funB(); // D::funB()
108
        printf("\r\n");
109
110
        // 将派生类引用赋值给基类引用
111
112
        // a->b a->c
113
        A &ra = b;
        ra.funA(); // A::funA()
114
115
        A &ra2 = c;
116
        ra2.funA(); // A::funA()
117
                     // b \rightarrow d c \rightarrow d
118
        B \& rb = d;
        rb.funA(); // D::funA()
119
        rb.funB(); // D::funB()
120
121
        B \& rb2 = d;
        rb2.funA(); // D::funA()
122
```

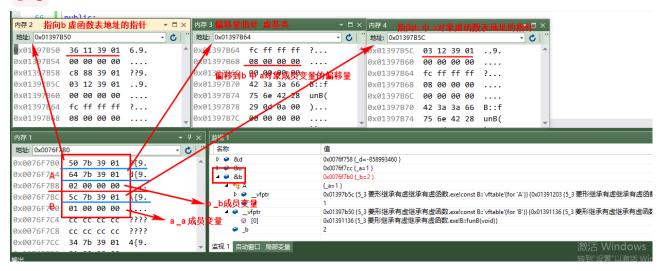
```
rb2.funB(); // D::funB()
123
124
125
        return 0;
126 }
127 /*
128 output:
129
130 10
131 A::funA()
132 A::funA()
133 A::funA()
134 B::funB()
135 A::funA()
136 C::funC()
137
138 A::funA()
139 A::funA()
140 D::funA()
141 D::funB()
142 D::funA()
143 D::funB()
144
145 A::funA()
146 A::funA()
147 D::funA()
148 D::funB()
149 D::funA()
150 D::funB()
151 */
```

对象内存分析

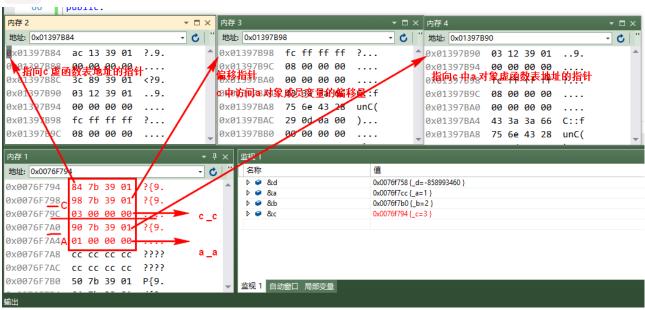
A a

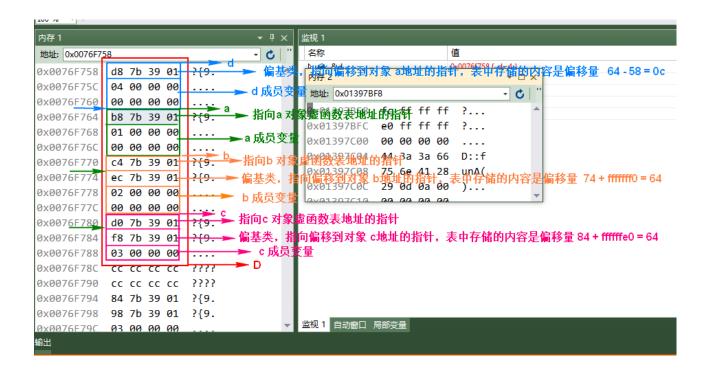


B b









函数调用

参考虚函数调用。