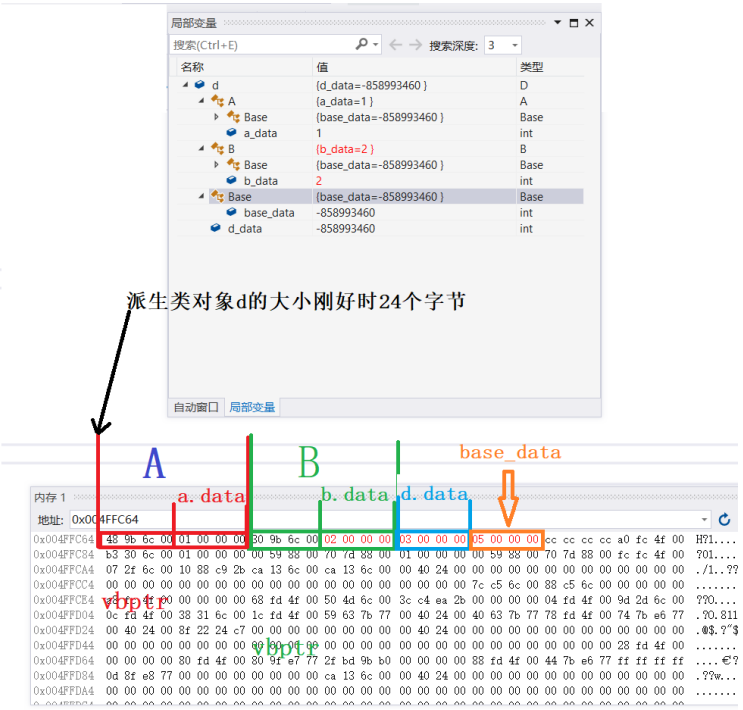
```cpp
//菱形继承(钻石继承) right example
class Base
{
public:
    int base_data;
};

class A : virtual public Base
{
public:
    int a_data;
};

class B : virtual public Base
{
public:
    int b_data;
};

class D :public A, public B  //多继承
{
public:
    int d_data;
};

void main()
{
    cout<<sizeof(D)<<endl;
    D d;
    d.a_data = 1;
    d.b_data = 2;
    d.d_data = 3;
    d.A::base_data = 5;
    d.B::base_data = 6;
    d.base_data = 0;
    cout << "sizeof(d) is "<< sizeof(d) << endl;  //24
    cout<<&(d.A::base_data)<<endl;
    cout<<&(d.B::base_data)<<endl;
}
```



vbptr指的是虚基类表指针（virtual base table pointer），该指针指向了一个虚基类表（virtual table），虚表中记录了虚基类与本类的偏移地址；通过偏移地址，这样就找到了虚基类成员，而虚继承也不用像普通多继承那样维持着公共基类（虚基类）的两份同样的拷贝，节省了存储空间。