

虚析构函数

赵英良

虚函数——特征：virtual

```
class A    // 基类
{
public:
    virtual void f(){...} // 虚函数
};
class B:public A // 派生类
{
public:
    virtual void f(){...}
};
```

```
int main    // 主函数
{
    B x;
    A *p;
    p=&x;
    p->f(); // 访问派生类的f
    return 0;
}
```

实例

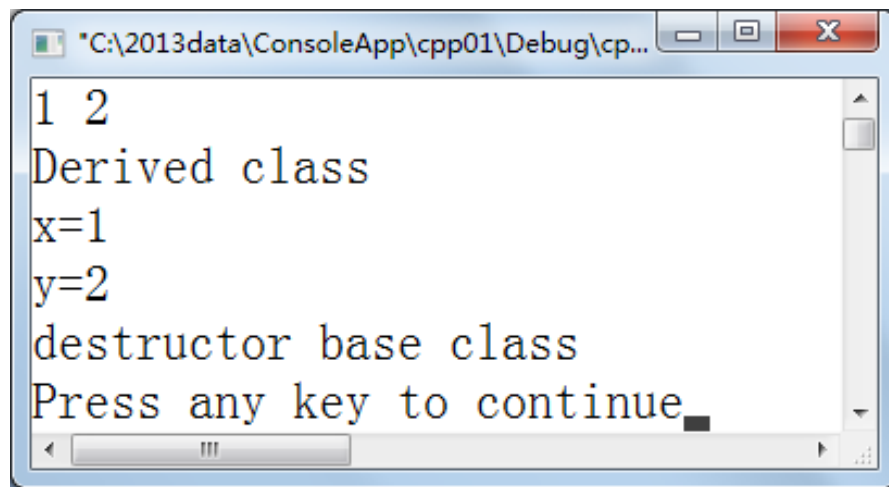
```
#include<iostream>
using namespace std;
class Base  //基类
{ public:
    int x;
    virtual void f(){cout<<"base class\n";};
    virtual void show(){cout<<"x="<<x<<endl;};
    ~Base(){cout<<"destructor base class\n";}
};
```

派生类

```
class Derived : public Base
{
public:
    int y;
    virtual void f(){
        cin>>y;
        cout<<"Derived class\n";
    }
    virtual void show(){Base::show ();cout<<"y="<<y<<endl;}
    ~Derived() { cout<<"destructor derived class\n";}
};
```

主函数

```
int main()
{
    Base *p;
    p=new Derived;
    cin>>p->x;
    p->f();
    p->show();
    delete p;
    return 0;
}
```



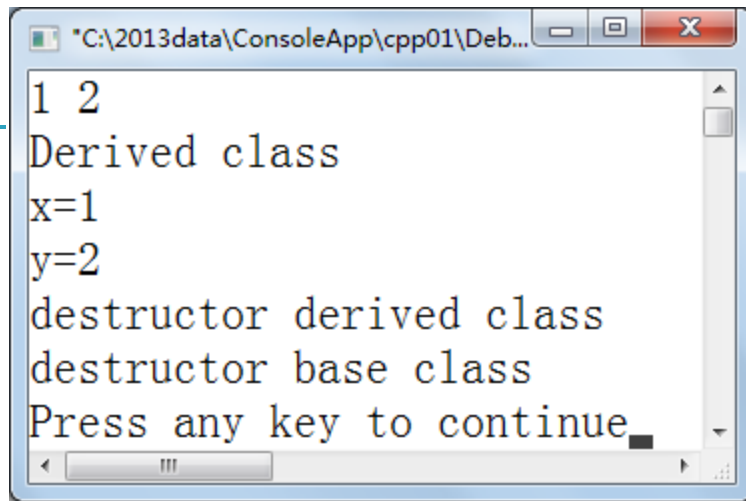
A screenshot of a Windows console window. The title bar shows the file path: "C:\2013data\ConsoleApp\cpp01\Debug\cp...". The console output is as follows:

```
1 2
Derived class
x=1
y=2
destructor base class
Press any key to continue
```

通过基类指针释放派生类对象空间
执行的是基类析构函数！！！！

修改基类析构函数

```
#include<iostream>
using namespace std;
class Base    //基类
{ public:
    int x;
    virtual void f(){cout<<"base class\n";};
    virtual void show(){cout<<"x="<<x<<endl;};
    virtual ~Base(){cout<<"destructor base
class\n";}
};
```



```
*C:\2013data\ConsoleApp\cpp01\Deb...
1 2
Derived class
x=1
y=2
destructor derived class
destructor base class
Press any key to continue.
```

虚函数的声明位置

```
class Base
```

```
{
```

```
public:
```

```
    int x;
```

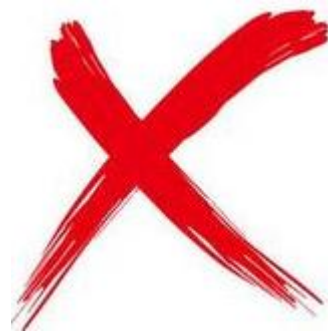
```
    virtual void f(){cout<<"base class\n";};
```

```
    void show();
```

```
    virtual ~Base(){cout<<"destructor base class\n"
```

```
};
```

```
virtual void Base::show(){cout<<"x="<<x<<endl;}
```



```
class Base
```

```
{
```

```
public:
```

```
    int x;
```

```
    virtual void f(){cout<<"base class\n";};
```

```
    virtual void show();
```

```
    virtual ~Base(){cout<<"destructor base class\n";}
```

```
};
```

```
void Base::show(){cout<<"x="<<x<<endl;}
```

