程序设计实习

郭炜　微博 http://weibo.com/guoweiofpku
http://blog.sina.com.cn/u/3266490431
刘家瑛 微博 http://weibo.com/pkuliujiaying

public继承的赋值兼容规则

public继承的赋值兼容规则

```
class base {     };
class derived : public base {    };
base b;
derived d;
```

1）派生类的对象可以赋值给基类对象

```
b = d;
```

2）派生类对象可以初始化基类引用

```
base & br = d;
```

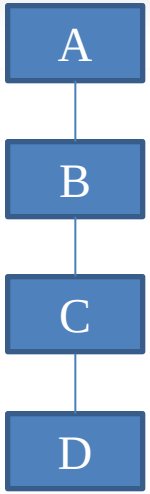
3）派生类对象的地址可以赋值给基类指针

```
base * pb = & d;
```

- 如果派生方式是 private或protected，则上述三条不可行。

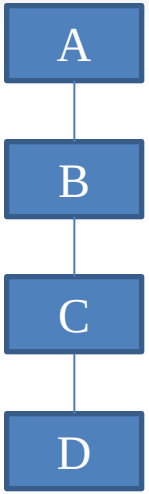
直接基类和间接基类

直接基类与间接基类



- 类A派生类B，类B派生类C，类C派生类D，……
  - 类A是类B的直接基类
  - 类B是类C的直接基类，类A是类C的间接基类
  - 类C是类D的直接基类，类A、B是类D的间接基类

直接基类与间接基类



- 在声明派生类时，只需要列出它的直接基类
  - 派生类沿着类的层次自动向上继承它的间接基类
  - 派生类的成员包括
    - 派生类自己定义的成员
    - 直接基类中的所有成员
    - 所有间接基类的全部成员

```
#include <iostream>
using namespace std;
class Base {
        public:
                int n;
                Base(int i):n(i)  {
                        cout << "Base " << n << " constructed" << endl;
                }
                ~Base() {
                        cout << "Base " << n << " destructed" << endl;
                }

};
```

```
class Derived:public Base
{
        public:
                Derived(int i):Base(i) {
                        cout << "Derived constructed" << endl;
                }
                ~Derived() {
                        cout << "Derived destructed" << endl;
                }

};
```

```
class MoreDerived:public Derived {
public:
        MoreDerived():Derived(4) {
                cout << "More Derived constructed" << endl;
        }
        ~MoreDerived() {
                cout << "More Derived destructed" << endl;
        }
};
int main()
{
        MoreDerived Obj;
        return 0;
}
```

输出结果：

Base 4 constructed
Derived constructed
More Derived constructed
More Derived destructed
Derived destructed
Base 4 destructed