派生类对象到基类的转换，见下例：

#include <iostream>

using namespace std;

class A // 基类

{

public:

virtual A\* f()

{

cout << "A的f()" << endl;

return this

}

};

class B : public A // 派生类

{

public:

virtual B\* f()

{

cout << "B的f()" << endl;

return this;

}

};

int main(int argc, char\* argv[])

{

A\* a\_ptr = new A;

a\_ptr->f(); // 执行A的f()

delete a\_ptr;

a\_ptr = NULL;

a\_ptr = new B; // 基类A指针指向派生类B对象

a\_ptr->f(); // 执行B的f()

delete a\_ptr;

a\_ptr = NULL;

B b;

A& a\_ref = b; // 基类A引用绑定派生类B对象

a\_ref.f(); // 执行B的f()

A a = b;

a.f(); // 执行A的f()，基类A的对象直接调用，不触发动态绑定

B\* b\_ptr = NULL;

a\_ptr = new A;

b\_ptr = (B\*)a\_ptr; // 这样做是不安全的。

b\_ptr->f(); // 执行A的f()，b\_ptr只能调用基类A的函数

delete a\_ptr;

a\_ptr = NULL;

b\_ptr = NULL;

return 0;

}

基类指针指向派生类对象或基类引用绑定到派生类对象上，实际对象仍然是派生类对象。

用派生类对象对基类对象进行初始化或赋值，则是将派生类转换为基类，对象的类型是固定的，静态类型和动态类型都是基类类型。

派生类到基类转换的可访问性：

public继承：用户代码和派生类成员都可以使用派生类到基类的转换。

protected继承：派生类成员可以使用派生类到基类的转换。

private继承：派生类不能转换为基类。

基类到派生类的转换：

基类对象不能转换为派生类。

不能将派生类的引用绑定到基类对象上。

可以将派生类的指针强制指向（采用static\_cast或dynamic\_cast）基类对象。