#include<iostream>

using namespace std;

class Base

{

protected:

    int a;

public:

    Base() {a = 0;}

};

class Derived1:  public Base

{

public:

    int c;

};

class Derived2:  public Base

{

public:

    int c;

};

class DerivedDerived: public Derived1, public Derived2

{

public:

    void show()  {   cout << a;  }

};

int main(void)

{

    DerivedDerived d;

    d.show();

    return 0;

}

Compiler Error in Line "cout << a;"

This is a typical example of [diamond problem of multiple inheritance](http://www.geeksforgeeks.org/multiple-inheritance-in-c/). Here the base class member 'a' is inherited through both Derived1 and Derived2. So there are two copies of 'a' in DerivedDerived which makes the statement "cout << a;" ambiguous. The solution in C++ is to use virtual base classes.

**Solution to the problem:**

#include<iostream>

using namespace std;

class Base

{

protected:

    int a;

public:

    Base() {a = 0;}

};

class Derived1: virtual public Base

{

public:

    int c;

};

class Derived2: virtual public Base

{

public:

    int c;

};

class DerivedDerived: public Derived1, public Derived2

{

public:

    void show()  {   cout << a;  }

};

int main(void)

{

    DerivedDerived d;

    d.show();

    return 0;

}