Multiple Inheritance


```
#include <iostream>
using namespace std;
class A1 {
 public:
   A1() {cout << "calling constructor of A1" << endl;}
   void f1() { cout << "calling f1 in A1" << endl; }
   };
class A2{
 public:
   A2() {cout << "calling constructor of A2" << endl;}
   void f2() { cout << "calling f2 in A2" << endl; }
};
class A3:  public A1,  public A2 {
 public:
   A3() {cout << "calling constructor of A3" << endl;}
};
```

Multiple Inheritance


```
int main()
{
        A1 * a1=new A1();
        a1->f1();
        A2 * a2=new A2();
        a2->f2();
        A3 * a3=new A3();
        a3->f1();
        a3->f2();
}
```

```
calling constructor of A1
calling f1 in A1

calling constructor of A2
calling f2 in A2

calling constructor of A1
calling constructor of A2
calling constructor of A3
calling f1 in A1
calling f2 in A2
```

Multiple Inheritance


```
class A {
 private:
 public:
   A() {cout << "calling constructor of A" << endl;}
    void f() { cout << "calling f in A" << endl; }
};
class A1 : public A {
 public:
   A1() {cout << "calling constructor of A1" << endl;}
   void f1() { cout << "calling f1 in A1" << endl; }
   };
```

Multiple Inheritance


```
int main()
{
      A1 * a1=new A1();
       a1->f1();
       a1->f();
       A2 * a2=new A2();
       a2->f2();
       A3 * a3=new A3();
       a3->f();
       a3->f1();
       a3->f2();
}
```

```
calling constructor of A
calling constructor of A1
calling f1 in A1
calling f in A

calling constructor of A2
calling f2 in A2

calling constructor of A
calling constructor of A1
calling constructor of A2
calling constructor of A3
calling f in A
calling f1 in A1
calling f2 in A2
```

Multiple Inheritance


```
class A2: public A{
 public:
   A2() {cout << "calling constructor of A2" << endl;}
   void f2() { cout << "calling f2 in A2" << endl; }
};
```

Compiler error

[C++ Error] MultipleInheritance.cpp(31): E2014 Member is ambiguous: 'A::f' and 'A::f'.


```
class A1 : virtual public A {
 public:
   A1() {cout << "calling constructor of A1" << endl;}
   void f1() { cout << "calling f1 in A1" << endl; }
   };
class A2: virtual public A{
 public:
   A2() {cout << "calling constructor of A2" << endl;}
   void f2() { cout << "calling f2 in A2" << endl; }
};
```

Multiple Inheritance


```
int main()
{
      A1 * a1=new A1();
       a1->f1();
       a1->f();
       A2 * a2=new A2();
       a2->f2();
       a2->f();

       A3 * a3=new A3();
       a3->f();
       a3->f1();
       a3->f2();
}
```

```
calling constructor of A
calling constructor of A1
calling f1 in A1
calling f in A

calling constructor of A
calling constructor of A2
calling f2 in A2
calling f in A

calling constructor of A
calling constructor of A1
calling constructor of A2
calling constructor of A3
calling f in A
calling f1 in A1
calling f2 in A2
```