**BK1000. 猫科动物和老虎**

**Description**

下面是不完整的继承类定义，  
  
class Cat  
{  
    int data;  
public:  
    Cat();  
};  
  
class Tiger : public Cat  
{  
    int data;  
public:  
    Tiger(int);  
};  
  
试完成其定义(你可以根据需要增加必要的构造函数和析构函数)，使得主函数main运行后能得到其后结果：  
  
int main()  
{  
    Cat bc;  
    Tiger dc(6);  
    Tiger dc1(18);  
  
    return 0;  
}

运行结果：  
Default Constructor of Cat is Running  
Constructor of Cat is Running 6  
Constructor of Tiger is Running 6  
Constructor of Cat is Running 18  
Constructor of Tiger is Running 18  
Destructor of Tiger is Running 18  
Destructor of Cat is Running 18  
Destructor of Tiger is Running 6  
Destructor of Cat is Running 6  
Destructor of Cat is Running 0  
  
注意：你只需要提交类定义。

**BK1001. Extending MyPoint class**

**Description**

The **MyPoint** class below is created to model a point in a two-dimensional space.  
Create a class named **ThreeDPoint** to model a point in a three-dimensional space. Let **ThreeDPoint** be derived from **MyPoint**.  
x, y, z represent x-, y- and z-coordinates.  
class MyPoint  
{  
  private:  
    double x, y;  
  public:  
    // The no-arg constructor that contruccts a point with coordinates(0,0)  
    MyPoint();  
  
    MyPoint(double x, double y);  
    double getX() const;  
    double getY() const;  
  
    //display the distance between two points in two-dimensional space.  
    double distance(const MyPoint &point);   
};  
  
class ThreeDPoint : public MyPoint  
{  
private:  
  double z;  
public:  
  // The no-arg constructor that contruccts a point with coordinates(0,0,0)    
  ThreeDPoint();  
  
  ThreeDPoint(double x, double y, double z);  
  double getZ() const;  
  
  //display the distance between two points in Three-dimensional space.  
  double distance(const ThreeDPoint &point);    
};

**BK1002. Multiple-inheritance**

**Description**

Below are two classes A and C:

class A  
{  
public:  
    A(int a0):a(a0) {}  
    void showA() { cout<<a<<endl; }  
protected:  
    int a;  
};

class C  
{  
public:  
    C(int c0):c(c0) {}  
    void showC() { cout<<c<<endl; }  
protected:  
    int c;  
};

You are required to derive a class B from both A and C. The B class has only one constructor, which has two parameters of the type int, with the first one for a and the second for c. The default values for the two parameters are 11 and 22 respectively.

Your submitted source code should include the whole implementation of the B class, but without the A and C classes.

No main() function should be included.

**BK1003. Constructor chains**

**Description**

There are four classes A, B, C and D. They all have the same form as follows:

class X

{  
public:  
    X() { cout << "In X()" << endl; }  
    ~X() { cout << "In ~X()" << endl; }  
};

X belongs to { A, B, C, D }. These four classes have some kind of inheritance relationship. You are to find the right one to generate the desired output, which should come from the constructors and destructors.

Your submitted source code should include all the implementation of the A, B, C and D classes.

No main() function should be included.

**Note**: the main() function of the test framework looks like this:

------------------------------------------------------------------------------

int main()  
{  
    D d;  
    return 0;  
}

------------------------------------------------------------------------------

**Output**

In B()  
In C()  
In B()  
In A()  
In D()  
In ~D()  
In ~A()  
In ~B()  
In ~C()  
In ~B()