```
#include<iostream>

using namespace std;

//动物类
class Animal{
public:
    //虚函数
    virtual void speak(){
        cout<<"动物在说话"<<endl;
    }

    virtual void action(){
        cout<<"动物在动作"<<endl;
    }
};

class Cat:public Animal{
public:
    void speak(){
        cout<<"小猫在说话"<<endl;
    }
};

class Dog:public Animal{
public:
    int age;

    Dog(int age){
        this->age=age;
    }
    void action(){
        cout<<"小狗在动作"<<endl;
    }

};

//执行说话的函数
//地址早绑定 在编译阶段确定函数地址
//如果想执行让猫说话,那么这个函数地址就不能提前绑定,需要在运行阶段进行绑定,地址晚绑定

//动态多态满足条件
//1.有继承关系
//2.子类重写父类函数
```

```
//动态多态使用
//父类的指针或者引用 执行子类对象
void doSpeak(Animal& animal){
	animal.speak();
}


int main(){
	Cat cat;
	doSpeak(cat);
	Dog dog(10);
	Animal& dogt=dog;
	dogt.action();
}
```