```
package com.principle.segregation.improve;

public class Segreration2 {
    public static void main(String[] args) {
        // 使用一把
        A a = new A();
        a.depend1(new B()); // A 类通过接口去依赖 B 类
        a.depend2(new B());
        a.depend3(new B());

        C c = new C();

        c.depend1(new D()); // C 类通过接口去依赖(使用)D 类
        c.depend4(new D());
        c.depend5(new D());
    }
}


// 接 口 1
interface Interface1 {
    void operation1();
}

// 接 口 2
interface Interface2 {
    void operation2();
    void operation3();
}

// 接 口 3
interface Interface3 {
    void operation4();
    void operation5();
}

class B implements Interface1,Interface2 {
    public void operation1() {
        System.out.println("B 实现了 operation1");
    }

    public void operation2() {
        System.out.println("B 实现了 operation2");
    }

    public void operation3() {
        System.out.println("B 实现了 operation3");
    }


}


class D implements Interface1,Interface3 {

    public void operation1() {
        System.out.println("D 实现了  operation1");
    }

    public void  operation4()  {
        System.out.println("D 实现了  operation4");
    }

    public void operation5() {
        System.out.println("D 实现了 operation5");
    }
}

class A { // A 类通过接口 Interface1,Interface2 依赖(使用) B 类，但是只会用到 1,2,3 方
法
    public void depend1(Interface1 i) {
        i.operation1();
    }


    public void depend2(Interface2 i) {
        i.operation2();
    }


    public void depend3(Interface2 i) {
        i.operation3();
    }
}

class C { // C  类通过接口 Interface1,Interface3  依赖(使用) D 类，但是只会用到 1,4,5 方
法
    public void depend1(Interface1 i) {
        i.operation1();
    }


    public void depend4(Interface3 i) {
        i.operation4();
    }


    public void depend5(Interface3 i) {
        i.operation5();
    }
}
```