```
package com.example;

public abstract class Animal {

    public abstract void eat();

}
```

```
package com.example;

public class Spider extends Animal implements Ambulatory {

    private Ambulatory ambulatory;

    public Spider() {
        ambulatory = new AmbulatoryImpl(8);
    }

    @Override
    public void eat() {
        System.out.println("The spider eats a fly.");
    }

    @Override
    public void walk() {
        ambulatory.walk();
    }

}
```

```
package com.example;

public class Cat extends Animal implements Pet, Ambulatory {

    private Nameable nameable = new NameableImpl();
    private Ambulatory ambulatory;

    public Cat() {
        this("Fluffy");
    }

    public Cat(String name) {
        ambulatory = new AmbulatoryImpl(4);
        setName(name);
    }

    @Override
    public void eat() {
        System.out.println("Cats like to eat spiders and fish.");
    }

    @Override
    public String getName() {
        return nameable.getName();
    }

    @Override
    public void setName(String name) {
        nameable.setName(name);
    }

    @Override
    public void play() {
        System.out.println(getName() + " likes to play with string.");
    }

    @Override
```

```
    public void walk() {
        ambulatory.walk();
    }

}

package com.example;

public class Fish extends Animal implements Pet {

    private Nameable nameable = new NameableImpl();

    @Override
    public void eat() {
        System.out.println("Fish eat pond scum.");
    }

    @Override
    public void setName(String name) {
        nameable.setName(name);
    }

    @Override
    public String getName() {
        return nameable.getName();
    }

    @Override
    public void play() {
        System.out.println("Just keep swimming.");
    }

}

package com.example;

interface Pet extends Nameable {

    public void play();

}

package com.example;

public class PetMain {

    public static void main(String[] args) {
        Animal a;
        //test a spider with a spider reference
        Spider s = new Spider();
        s.eat();
        s.walk();
        //test a spider with an animal reference
        a = new Spider();
        a.eat();
        //a.walk();

        Pet p;

        Cat c = new Cat("Tom");
        c.eat();
        c.walk();
        c.play();
        a = new Cat();
        a.eat();
        //a.walk();
        p = new Cat();
```

```
        p.setName("Mr. Whiskers");
        p.play();

        Fish f = new Fish();
        f.setName("Guppy");
        f.eat();
        //f.walk();
        f.play();
        a = new Fish();
        a.eat();
        //a.walk();


        playWithAnimal(s);
        playWithAnimal(c);
        playWithAnimal(f);
    }

    public static void playWithAnimal(Animal a) {
        if (a instanceof Pet) {
            Pet p = (Pet) a;
            p.play();
        } else {
            System.out.println("Danger! Wild Animal");
        }
    }
}
```

```
package com.example;

public interface Nameable {

    public void setName(String name);

    public String getName();

}
```

```
package com.example;

public class NameableImpl implements Nameable {

    private String name;

    @Override
    public void setName(String name) {
        if(name.length() < 20) {
            this.name = name;
        } else {
            System.out.println("Name too long");
        }
    }

    @Override
    public String getName() {
        return name;
    }

}
```

```
package com.example;

public interface Ambulatory {

    public void walk();

}
```

```java
package com.example;

public class AmbulatoryImpl implements Ambulatory {

    private int legs;

    public AmbulatoryImpl(int legs) {
        this.legs = legs;
    }

    public void walk() {
        System.out.println("This animal walks on " + legs + " legs.");
    }
}
```