```java
public class Tahiti {

    Tahiti t;

    Tahiti go(Tahiti t) {

        Tahiti t1 = new Tahiti();
        Tahiti t2 = new Tahiti();

        t1.t = t2;
        t2.t = t1;
        t.t = t2;
        return t1;
    }
    public static void main(String[] args) {
        Tahiti t = new Tahiti();
        Tahiti t2 = t.go(t);
        t2 = null;
        // more code here 11
    }

}
```
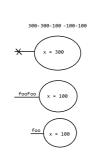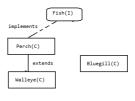
t1   t=null t1
t2   t=null t1
t   t=null
t2

Tahiti

```java
class Foo {
    private int x;
    public Foo( int x ){ this.x = x;}
    public void setX( int x ) { this.x = x; }
    public int getX(){ return x;}
}

public class Gamma {
    static Foo fooBar(Foo foo) {
        foo = new Foo(100);
        return foo;
    }
    public static void main(String[] args) {
        Foo foo = new Foo( 300 );
        System.out.println( foo.getX() + "-");
        Foo fooFoo = fooBar(foo);
        System.out.println(foo.getX() + "-");
        System.out.println(fooFoo.getX() + "-");
        foo = fooBar( fooFoo);
        System.out.println( foo.getX() + "-");
        System.out.println(fooFoo.getX());
    }
}
```

300-300-100 -100-100

x = 300

fooFoo   x = 100

foo   x = 100

```java
Q>
interface Fish {}
class Perch implements Fish {}
class Walleye extends Perch {}
class Bluegill {}

public class Fisherman {
 public static void main(String[] args) {
    Fish f = new Walleye();
    Walleye w = new Walleye();
    Bluegill b = new Bluegill();

    if (f instanceof Perch)
        System.out.print("f-p ");    ✓
    if (w instanceof Fish)
        System.out.print("w-f ");    ✓

    if (b instanceof Fish)
        System.out.print("b-f ");    ✗
 }
}
```

Fish(I)

implements

Perch(C)

extends

Walleye(C)

Bluegill(C)

X instanceof Y
|=> interface type means no compiletime error