

Inheritance (reusing implementation by using inheritance)

```
public class A {  
    private int i = 0; // hidden from all other objects, except instances  
of A  
  
    protected double d1 = 0.0D; // subclasses of A will inherit this  
    protected double d2 = 0.0D; // subclasses of A will inherit this  
  
    public void setI(int in) {  
        i = in;  
    }  
  
    public void setd1(double din) {  
        d1 = din;  
    }  
  
    public void setd2(double din) {  
        d2 = din;  
    }  
  
    public void print() {  
        System.out.println("A's print() called");  
        System.out.println("d1: "+d1+", d2:"+d2);  
    }  
  
    public double acc(double d1) {  
        System.out.println("A's acc(double) called");  
        return this.d1+this.d2+d1;  
    }  
}
```

```

public class B extends A {

    protected double d3 = 0.0D;

    public void setd3(double din) {
        d3 = din;
    }

    // print() overrides print() in class A
    public void print() {
        System.out.println(" B's print() called");
        //System.out.println("i1: "+i); i is a private data member of A,
        // compile time error
        System.out.println("d1: "+d1+", d2: "+d2+", d3: "+d3);
        // d1, d2, are protected data members of A, and are inherited by B
    }

    // acc(double) overrides acc(double) in class A
    public double acc(double x1) {
        System.out.println(" B's acc(double) called");
        return super.acc(x1)+d3;
    }

    // acc(double, double) overloads acc(double) in A
    public double acc(double x1, double x2) {
        System.out.println(" B's acc(double, double) called");
        return d1+d2+d3+x1+x2;
    }
}

```

```

public class MainClass {

    public static void main(String[] args) {

        output

        B myB = new B();
        myB.setd1(3.4D); // invoking setd1(double) inherited from A
        myB.setd2(4.4D); // invoking setd2(double) inherited from A
        myB.setd3(5.5D);
        myB.print(); B's print() called
                    d1: 3.4, d2: 4.4, d3: 5.5

        System.out.println(myB.acc(5.5D));
                    B's acc(double) called
                    A's acc(double) called
                    18.8

        System.out.println(myB.acc(5.5D, 6.5D));
                    B's acc(double, double) called
                    19.8

        A myA = new B(); // myA is declared as a reference to an object of
                        // type A, but is actually "pointing" to an object
                        // of type B

        myA.setd1(2.3D);
        myA.setd2(2.6D);
        // myA.setd3(6.6D); setd3 not defined in A. Compile time error!
        myA.print(); // polymorphic B's print() called
                    d1: 2.3, d2: 2.6, d3: 0.0

        myA.acc(5.6D); // polymorphic B's acc(double) called
                    A's acc(double) called

        // myA.acc(3.4D, 5.6D); acc(double, double) not defined in A
        // compile time error!

    }
}

```
