

```

class Aufgabe12 {

    private int[] feld = new int[100];

    public int[] fuelle() {
        for (int i = 0; i < 100; i++) {
            feld[i] = (int) (Math.random()*1000);
        }
        return feld;
    }

    public int[] sortiere(int[] unsortiert) {
        for (int j = 0; j < 99; j++)
            for (int i = 0; i < 99; i++) {
                if (unsortiert[i] > unsortiert[i+1]) {
                    int buffer = unsortiert[i];
                    unsortiert[i] = unsortiert[i+1];
                    unsortiert[i+1] = buffer;
                }
            }
        return unsortiert;
    }

    public static void main(String[] args) {

        Aufgabe12 Feldsortierer = new Aufgabe11();
        int[] FeldUnsortiert = Feldsortierer.fuelle();
        int[] FeldSortiert = Feldsortierer.sortiere(FeldUnsortiert);

        for(int i=0; i<100; i++)
            System.out.print(FeldSortiert[i] + "\t");

    }
}

```

```

public class Complex {

    private double re, im;          // Real- und Imaginärteil

    public Complex(double r, double i) { re = r; im = i; }

    public Complex(double r) { this(r,0.0); }

    public static final Complex ZERO = new Complex(0.0, 0.0);
    public static final Complex ONE = new Complex(1.0, 0.0);
    public static final Complex I = new Complex(0.0, 1.0);

    public double reTeil() { return re; }

    public double imTeil() { return im; }

    public String toString() {
        String s = "" + re;
        if ( im < 0 ) s += " - " + (-im) + "i";
        else s += " + " + im + "i";
        return s;
    }

    public Complex add(Complex b) {
        return new Complex( re + b.reTeil(), im + b.imTeil() );
    }

    public Complex subtract(Complex b) {
        return new Complex( re - b.reTeil(), im - b.imTeil() );
    }

    public Complex negate() { return new Complex(-re,-im); }

    public Complex conjugate() { return new Complex(re,-im); }

    public double abs() { return ( re * re + im * im ); }

    public Complex multiply(Complex b) {

```

```

return new Complex(re * b.reTeil() - im * b.imTeil(),
                   re * b.imTeil() + im * b.reTeil() );
}

public boolean equals(Complex b) {
return ( re == b.reTeil() && im == b.imTeil() );
}
}

```

```

public class ComplexTest {

    public static void main(String[] args) {

        Complex a = new Complex(2.0,3.0);
        System.out.println("a = " + a);

        System.out.println("Null = " + Complex.ZERO);

        Complex b = Complex.ONE;
        b = b.add(a);      System.out.println("b = " + b);

        Complex c = new Complex(3.0,3.0);      System.out.println("c = " + c);
        boolean t = c.equals(b);      System.out.println("t = " + t);

        Complex d = c.conjugate();      System.out.println("d = " + d);

        double e = c.abs();      System.out.println("e = " + e);

        Complex f = c.multiply(d);      System.out.println("f = " + f);

        t = e == f.reTeil();      System.out.println("t = " + t);

        Complex g = Complex.I.multiply(Complex.I);
        System.out.println("g = " + g);
    }
}

```