Topics: Method overloading, OOP review

Reading (GG): Sec 3.4

Method overloading

Each method has a signature: the method name and the parameter types (including the order). In a class, methods can have the same name as long as the signatures are different. The constructors in a class are often overloaded.

class Interval {
    private double base;                      // low end
    private double width;                     // interval width
    public final static double MAXwidth = 5;  // max width of interval

    public Interval(double b, double w) {
        setBase(b);
        setWidth(w);
    }

    public Interval() {}

    /* An Interval with base b and MAXwidth */
    public Interval(double b) {
        setBase(b);
        setWidth(MAXwidth);
    }

    public double setWidth(…)  { … }
    public boolean isIn(Interval o)  { … }
    public static Interval overlap(Interval a, Interval b) { … }
    //… other methods
}

Invoking methods

Assume three Intervals have been instantiated: i1, i2, i3. Assume i1 and i2 overlap. Write code to find if the overlapped Interval of i1 and i2 is in Interval i3.

Interval i1 = new Interval(…);
Interval i2 = new Interval(…);
Interval i3 = new Interval(…);
// Assume i1 and i2 overlap

if (                          )
    System.out.println(…);
else
    System.out.println(…);
Why doesn’t this swap work?

/* Try to swap two values */
public class BadSwap {
    public static void main(String[] args) {
        int x = Keyboard.readInt();  int y = Keyboard.readInt();
        swap(x, y);
    }
    public static void swap(int x, int y) {
        int tmp;
        tmp = x;
        x = y;
        y = tmp;
    }
} //class BadSwap

A different class

/** Organize data for any Person: name, age, best friend */
public class Person {
    private String name;
    private int age;

    public static final int LEGALAge=18;
    /** Constructor */
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    /** =This Person is an adult */
    public boolean isAdult() {  return age >= MATURE; }
    /** Make a friend with p */

    /** =String description of this Person */
    public String toString() {
        return name + " is " + age;
    }
}

public static void main(String[] args) {
    Person a = new Person("AJ",9);
    Person b = new Person("BP",7);
    a.beFriendOf(b);
    a.makeFriend(b);
} // class Person