Topics: OOP review, OO Design

Reading: Text Sec 3.4 or PL activities on Lesson Page 3-8

Class `Interval`

We complete our class `Interval`. The java file is posted on the Notes page. Be sure to analyze the code, read the comments, and experiment with the class by changing/adding methods and by writing client code.

```
/** Numeric interval -- closed intervals
 *  Intervals have a maximum width (MAXwidth)
 */
class Interval {

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

  /** Constructor: An Interval has base b and width w _____________________*/
  public Interval(double base, double w) {
    this.base= base;
    width=
  }

  /** Constructor: An Interval with base 0 and width w */
  public Interval(double w) { this(0,w); }

  /** Constructor: An Interval with initial values given by declarations */
  public Interval() {}

  /** =Get left end of this Interval */
  public double getBase() { return base; }

  /** =Get right end of this Interval */
  public double getEnd() { return base + width; }

  /** Set left end of this Interval to base */
  public void setBase(double base) {
    this.base= base;
  }

  /** Set width of this Interval to w ___________________________________*/
  public void setWidth(double w) {
    width=
  }

  /** =String description of this Interval */
  public String toString(){
    return "[" + getBase() + "," + getEnd() + "]";
  }

  /** Expand this Interval by a factor of f (expand to the right) */
  public void expand(double f) {
    setWidth(width*f);
  }

  /** =This Interval is in i
   *  If the ends of this Interval and i are exactly equal, consider
   *  this Interval to be in i
   */
```

```
  public boolean isIn(Interval i) {
    return ( getBase()>=i.getBase() && getEnd()<=i.getEnd() );
  }

  /** =The overlapped Interval between this Interval and Interval b */
  public Interval overlap(Interval b) {
    Interval olap;      // overlapped interval
    double left, right; // olap's left & right

    left= Math.max(getBase(),b.getBase());
    right= Math.min(getEnd(),b.getEnd());
    if ((right-left)<=0 ) //treat overlap of width 0 as no overlap
      olap= null;
    else
      olap= new Interval(left, right-left);
    return olap;
  }

  /** =The overlapped Interval between Intervals a and b */
  public static Interval overlap(Interval a, Interval b) {
    Interval olap;      // overlapped interval
    double left, right; // olap's left & right

    left= Math.max(a.getBase(),b.getBase());
    right= Math.min(a.getEnd(),b.getEnd());
    if ((right-left)<=0 ) //treat overlap of width 0 as no overlap
      olap= null;
    else
      olap= new Interval(left, right-left);
    return olap;
  }

}
```

Invoking methods

Assume three **`Interval`**s 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`**.

A different class

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


	public final static 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 */




	/** Become a friend of 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
```