Lösningsförslag till tentamen

Kurs	**Objektorienterad programmering**
Tentamensdatum	**2012-12-18**
Program	**DAI2**
Läsår	**2012/2013, lp 2**
Examinator	**Uno Holmer**

Uppgift 1 (10 p)

Ingen lösning ges. Se kurslitteraturen.

Uppgift 2 (9+3 p)

a)

```
public class MinMaxThermometer implements DataReceiver {
    private double temperature;
    private double minTemperature;
    private double maxTemperature;
    private boolean firstValueReceived;

    public MinMaxThermometer() {
        firstValueReceived = false;
        temperature = 0;
        reset();
    }

    public double getTemperature() {
        return temperature;
    }

    public double getMinTemperature() {
        return minTemperature;
    }

    public double getMaxTemperature() {
        return maxTemperature;
    }

    public void reset() {
        minTemperature = maxTemperature = temperature;
    }

    @Override
    public void setValue(double value) {
        temperature = value;
        if ( ! firstValueReceived ) {
            reset();
            firstValueReceived = true;
        } else {
            if ( temperature < minTemperature )
                minTemperature = temperature;
            if ( temperature > maxTemperature )
                maxTemperature = temperature;
        }
    }
}
```

b)

```
public static void main(String[] args) {
    TempSensor sensor = new TempSensor(1000);
    MinMaxThermometer thermo = new MinMaxThermometer();
    sensor.addReceiver(thermo);
}
```

Uppgift 3 (12 p)

```
public class BirdClub {
    private TreeMap<String,Set<String>> observations;

    public BirdClub() {
        observations = new TreeMap<>();
    }

    public void addObservation(String member,String birdName) {
        if ( ! observations.containsKey(member) )
            observations.put(member,new TreeSet<String>());
        observations.get(member).add(birdName);
    }

    public void addObservation(List<Observation> l) {
        for ( Observation obs : l )
            addObservation(obs.getObserver(),obs.getBirdName());
    }

    public int getRank(String person) {
        if ( observations.containsKey(person) )
            return observations.get(person).size();
        else
            return 0;
    }

    public void print300() {
        for ( String member : observations.keySet() ) {
            int rank = getRank(member);
            if ( rank >= 300 )
                System.out.println(member + ": " + rank);
        }
    }
    // Alternativt:
    public void print300() {
        for ( Map.Entry<String,Set<String>> e : observations.entrySet() ) {
            int rank = e.getValue().size();
            if ( rank >= 300 )
                System.out.println(e.getKey() + ": " + rank);
        }
    }
}
```

Uppgift 4 (10 p)

Utskriften blir 252.0+(-210.0) = 42.☺

Objekt	Dyn. typ	Metod	Inparam.	Returvärde	Kommentar

`arr[0]`	`B`	`B.f`	`50.0`	`100-A.g(50.0) ->`
				`100-(-152.0) ->`
				`252.0`
					`g` ärvs från `A`

`arr[0]`	`B`	`A.g`	`50.0`	`A.h(50.0-205.0) ->` `-152.0`	`h` ärvs från `A` eftersom `B.h(`**`int`**`)` ej överskuggar `A.h(`**`float`**`)`
`arr[0]`	`B`	`A.h`	`-155.0`	`-155.0+3 -> -152.0`	

`arr[1]`	`C`	`C.f`	`50.0`	`100+A.g(50.0) ->`
				`100+(-310.0) ->`
				`-210.0`
					`g` ärvs från `A`

`arr[1]`	`C`	`A.g`	`50.0`	`C.h(50.0-205.0) ->` `-310.0`	`C.h(float)` överskuggar `A.h(float)`
`arr[1]`	`C`	`C.h`	`-155.0`	`-155.0*2 -> -310.0`	

Uppgift 5 (3+3+1+5+4 p)

a)
```
public final boolean equals(Object o) {
    if ( o == this )
        return true;
    else if ( o instanceof Point ) {
        Point other = (Point)o;
        return x == other.x && y == other.y;
    } else
        return false;
}
```

b)
```
public int hashCode() {
    return (123 + x)*37 + y;
}
```

c)
```
public class ImageTag implements Tag {
    private ImageViewer viewer;

    public ImageTag(String imageFileName) throws IOException {
        viewer = new ImageViewer(imageFileName);
    }

    public void play() {
         viewer.displayImage();
    }
}
```

d)

```
public class GeoTags {
    private HashMap<Point,List<Tag>> tagMap;

    public GeoTags() {
        tagMap = new HashMap<>();
    }

    public boolean hasPoint(Point p) {
        return tagMap.containsKey(p);
    }

    public void addTag(Point p,Tag t) {
        if ( ! hasPoint(p) )
            tagMap.put(p,new ArrayList<Tag>());
        tagMap.get(p).add(t);
    }

    public Iterator<Tag> getTags(Point p) throws UnknownPointException {
        if ( ! hasPoint(p) )
            throw new UnknownPointException("No such point: " + p);
        else
            return tagMap.get(p).iterator();
    }
}
```

e)

```
GeoTags tags = new GeoTags();
Point p = new Point(123,456);
try {
    tags.addTag(p,new ImageTag("kungsgatan.jpg"));
    tags.addTag(p,new CommentTag("Stort 5 m djupt hål i Kungsgatan"));

    Iterator<Tag> it = tags.getTags(p);
    while ( it.hasNext() )
        it.next().play();
}
catch (IOException | UnknownPointException e) {
    System.out.println(e);
}
```