Data och Informationsteknik / Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
Ansvarig lärare: Magnus Myréen

DAT050 Objektorienterad programmering

Modellsvar för tentamen, torsdag, 2020-10-29, 08:30-12:30

Vitsordsgränser: 3=24p, 4=36p, 5=48p, max 60p.

Kom ihåg att inte fastna på en uppgift. Bestäm i förväg din egen tidsgräns per uppgift. ***Lycka till!***

Modellsvar för Uppgift 1: [10 poäng totalt] *att skriva klass, equals, interface*

```
public class Nat implements java.lang.Comparable<Nat> {

    private int value;

    public Nat(int n) {
        if (n < 0) { throw new IllegalArgumentException("Cannot create Nat."); }
        value = n;
    }

    public Nat add(Nat other) {
        // new Nat throws exception in case of overflow in integer arithmetic
        return new Nat(this.value + other.value);
    }

    public Nat sub(Nat other) {
        int k = this.value - other.value;
        if (k < 0) { k = 0; }
        return new Nat(k);
    }

    public boolean equals(Object obj) {
        if (obj == this) { return true; }
        if (obj == null || obj.getClass() != this.getClass()) { return false; }
        Nat other = (Nat)obj;
        return (other.value == this.value);
    }

    public String toString() {
        return String.valueOf(value);
    }

    public int compareTo(Nat other) {
        return this.value - other.value;
    }

}
```

Modellsvar för Uppgift 2: [6 poäng totalt] *testning, exceptions*

```
// returns true if createRandomArray seems to be correct
public boolean isOK() {
    try {
        int[] a = createRandomArray();
        if (a == null) { return false; }
        for (int i=1; i<100; i++) {
            int[] b = createRandomArray();
            if (b == null) { return false; }
            if (!isSame(a,b)) { return true; }
        }
        return false;
    } catch (Exception e) {
        return false;
    }
}

// assumes a and b are not null
private boolean isSame(int[] a, int[] b) {
    if (a.length != b.length) { return false; }
    for (int i=0; i<a.length; i++) {
        if (a[i] != b[i]) { return false; }
    }
    return true;
}
```

Modellsvar för Uppgift 3: [15 poäng totalt] *användning av standard klasser, testning*

Del a)

```
public double distance(Place a, Place b) {
    Set<RoadSegment> roads = getRoadsFrom(a);
    Iterator<RoadSegment> it = roads.iterator();
    while (it.hasNext()) {
        RoadSegment rs = it.next();
        Place end = rs.getEndPoint();
        if (end.equals(b)) {
            return rs.getLength();
        }
    }
    throw new IllegalArgumentException("No such route");
}
```

Del b)

```
public double distance(Place[] path) {
    if (path == null || path.length < 1) {
        throw new IllegalArgumentException("Bad input");
    }
    double total = 0.0;
    for (int i=0; i<path.length-1; i++) {
        total = total + distance(path[i], path[i+1]);
    }
    return total;
}
```

Del c)

```
public Set<Place> reachableWithin(Place origin, double d) {
    Set<Place> res = new HashSet<Place>();
    if (d >= 0) { addWithin(origin, d, res); }
    return res;
}

private void addWithin(Place origin, double d, Set<Place> res) {
    res.add(origin);
    Set<RoadSegment> roads = getRoadsFrom(origin);
    Iterator<RoadSegment> it = roads.iterator();
    while (it.hasNext()) {
        RoadSegment rs = it.next();
        double len = rs.getLength();
        if (len <= d) {
            addWithin(rs.getEndPoint(), d-len, res);
        }
    }
}
```

En alternativ lösning till Del c)

```
public Set<Place> reachableWithin(Place origin, double d) {
    Set<Place> res = new HashSet<Place>();
    if (d < 0) { return res; }
    for (RoadSegment rs : getRoadsFrom(origin)) {
        res.addAll(reachableWithin(rs.getEndPoint(), d - rs.getLength()));
    }
    return res;
}
```

Modellsvar för Uppgift 4: [23 poäng totalt] *interface, kopiering, Java Generics*

Del a)

```
public class ThreesIter implements Iter<Integer> {

   public boolean hasNext() {
       return true;
   }

   public Integer next() {
       return 3;
   }

   public Iter<Integer> clone() {
       return new ThreesIter();
   }

}
```

Del b)

```
public class OneTwoThreeIter implements Iter<Integer> {

    private int n;

    public boolean hasNext() {
        return (n < 3);
    }

    public Integer next() {
        n = n+1;
        return n;
    }

    public OneTwoThreeIter() {
        n = 0;
    }

    private OneTwoThreeIter(OneTwoThreeIter other) {
        n = other.n;
    }

    public Iter<Integer> clone() {
        return new OneTwoThreeIter(this);
    }

}
```

Del c)

```
public class EmptyIter<A> implements Iter<A> {

   public boolean hasNext() {
       return false;
   }

   public A next() {
       return null;
   }

   public Iter<A> clone() {
       return new EmptyIter<A>();
   }

}
```

Del d)

```
public class AppendIter<A> implements Iter<A> {

    private Iter<A> fst;
    private Iter<A> snd;

    public AppendIter(Iter<A> first, Iter<A> second) {
        fst = first.clone();
        snd = second.clone();
    }

    public boolean hasNext() {
        return (fst.hasNext() || snd.hasNext());
    }

    public A next() {
        if (fst.hasNext()) {
            return fst.next();
        } else {
            return snd.next();
        }
    }

    public Iter<A> clone() {
        return new AppendIter<A>(fst,snd);
    }

}
```

Del e)

```
    public static Iter appendAll(Iter[] array) {
        Iter res = new EmptyIter();
        if (array == null) { return res; }
        for (int i=0; i<array.length; i++) {
            if (array[i] != null) {
                res = new AppendIter(res,array[i]);
            }
        }
        return res;
    }
```

Del f)

```
public class OnlyEvenIter implements Iter<Integer> {

    private Iter<Integer> it;
    private boolean hasValue;
    private int value;

    public OnlyEvenIter(Iter<Integer> it) {
        this.it = it.clone();
        hasValue = false;
    }

    public boolean hasNext() {
        while (!hasValue) {
            if (!it.hasNext()) { return false; }
            value = it.next();
            if (value % 2 == 0) {
                hasValue = true;
            }
        }
        return true;
    }

    public Integer next() {
        hasNext(); // loads correct next into this.value
        return value;
    }

    private OnlyEvenIter(OnlyEvenIter other) {
        it = it.clone();
        hasValue = other.hasValue;
        value = other.value;
    }

    public Iter<Integer> clone() {
        return new OnlyEvenIter(this);
    }

}
```

Modellsvar för Uppgift 5: [6 poäng totalt] *händelsehantering, GUI programmering*

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class ShakeFrame extends JFrame implements MouseMotionListener {

    public ShakeFrame() {
        super();
        addMouseMotionListener(this);
        setLocation(50,50);
    }

    public void mouseMoved(MouseEvent e) {
        setLocation(50 + (int)(Math.random()*10.0) - 5,
                    50 + (int)(Math.random()*10.0) - 5);
    }

    public void mouseDragged(MouseEvent e) {}

}
```