Tentamen Objectgeorienteerd Programmeren II IN1120

dinsdag 30 augustus 2005 9.00-12.00
Uitwerking

Opgave 1

a. compilerfout, length in String is een methode, geen (public) attribuut.

b. runtime fout: nullPointerException. d2 = null, dus er is geen object om tegen te praten

c. `true`**,** (Maak een object-diagram) . d4 is een alias van d2, en heeft dus dezelfde waarden als d2. Door de aanroep d2.setInhoud("inhoud3") heeft dit object dezelfde inhoud en dezelfde versie als d3. Dat is voor de methode equals voldoende om true terug te geven.

d. `false`. Weliswaar is de inhoud hetzelfde, maar de referenties zijn verschillend.

e. compilerfout: de operator <= is niet gedefinieerd voor Strings**.**

Opgave 2

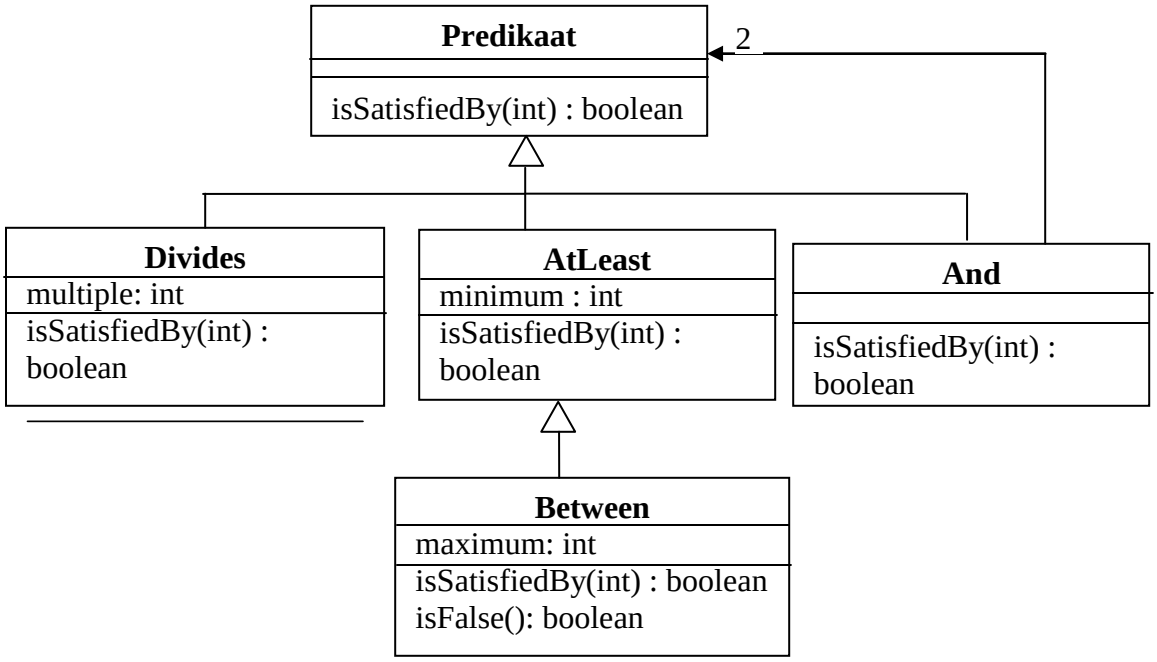
a.

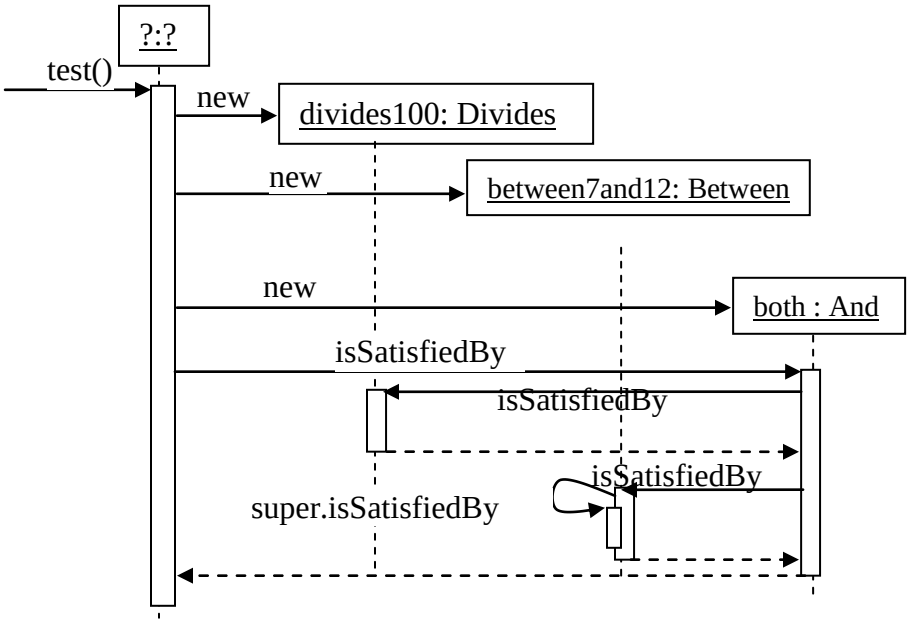
```
public Document langsteDocument(){
   Document kampioen = null;
   // als ondanks de preconditie de ordner leeg is,
   // wordt null teruggegeven
   if (aantal > 0){
      kampioen = document[0];
      for (int k = 1; k < aantal; k++)
         if (kampioen.getInhoud().length()
             <= document[k].getInhoud().length())
            kampioen = document[k];
   }
   return kampioen;
}
```

b.

```
public String overzicht(){
   StringBuffer buffer = new StringBuffer();
   for (int k = 0; k < aantal; k++){
       Document doc = document[k];
       buffer.append(doc.getTitel());
       buffer.append(',');
       buffer.append(doc.getVersie());
       if (k < aantal-1)
          buffer.append(",\n,");
   }
   return buffer.toString();
}
```

Opgave 3


Predikaat
isSatisfiedBy(int) : boolean
2
Divides
multiple: int
isSatisfiedBy(int) : boolean
AtLeast
minimum : int
isSatisfiedBy(int) : boolean
And
isSatisfiedBy(int) : boolean
Between
maximum: int
isSatisfiedBy(int) : boolean
isFalse(): boolean


?:?
test()
new
divides100: Divides
new
between7and12: Between
new
both : And
isSatisfiedBy
isSatisfiedBy
isSatisfiedBy
super.isSatisfiedBy

Opgave 4

a. Fout bij vertalen in de tweede regel: er wordt een AtLeast verwacht, maar p is (statisch) een Predikaat.
b. Vertalen gaat goed. Tijdens verwerken: ClassCastException: een Predikaat kan niet naar een Between worden gecast.
c. geen fouten. `bool ==false` (30 ligt niet tussen 5 en 10)
d. geen fouten. `bool ==true` (7 is niet kleiner dan 5)
e. Fout tijdens vertalen: een AtLeast heeft geen methode isFalse.

Opgave 5

```
import java.io.*;
...
public void execute(){
   try{
      FileReader in = new FileReader(INFILE);
      BufferedReader bin = new BufferedReader(in);
      String regel = bin.readLine();
      while (regel != null){
         StringTokenizer st = new StringTokenizer(regel);
         String token = st.nextToken();
         char richting = token.charAt(0);
         token = st.nextToken();
         int grootte = Integer.parseInt(token);
         stap(richting, grootte);
         regel = bin.readLine();
       }
       bin.close();
    }
    catch (IOException iox)
    {System.out.println(iox)}
}
```

Opgave 6

```
import java.awt.*;
public class SchildpadVenster extends Frame{
    ...
    public SchildpadVenster(){
       ... // create graphics
      startknop = new Button();
      add(startknop);
      schildpad = new Schildpad();
      startknop.addActionListener(schildpad);
    }

}
```

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

public class Schildpad implements ActionListener{
    private final static String INFILE = "program.txt" ;
    private Point hier = new Point(100,100);

    public void actionPerformed(ActionEvent event){
        execute();
    }
}
```

Opgave 7

```
import java.awt.*;
import java.util.*;
import java.io.*;
import java.awt.event.*;

public class Schildpad extends Observable implements
ActionListener{
    ...
    private void stap(char c, int step){
        switch (c){
           ...
        }
        setChanged();
        notifyObservers(hier);
    }
}

import java.awt.*;
import java.util.*;

public class TekenCanvas extends Canvas implements Observer{
    private Point hier;

    public void update(Observable o, Object arg){
        Point daar = (Point) arg;
        gaNaar(daar);
    }
}

public class SchildpadVenster extends Frame{
    public SchildpadVenster(){
        ...
        tekenCanvas = new TekenCanvas(new Point(100,100));
        add(tekenCanvas);
        schildpad = new Schildpad();
        schildpad.addObserver(tekenCanvas);
        startknop.addActionListener(schildpad);
    }
}
```

Opgave 8

1. De rij wordt verdeeld in een verwerkt en een onverwerkt gedeelte. Van de elementen in het verwerkte gedeelte is de frequentie (in dat gedeelte) geteld, en de grootst voorkomende frequentie wordt onthouden.
2.
   - int index: de index van het eerste niet-verwerkte element
   - int maxFreq: de grootste frequentie tot dusver gevonden
   - int currFreq: de frequentie van het laatst verwerkte element
3. *invariant:*zie onder 2.
4. stoppen als `index == aantal` (dan hebben we ze allemaal gehad)
5. omdat we achteruitkijken moeten we beginnen met index = 1. De lege rij moet dus apart worden behandeld.
   ```
   index = 1;
   maxFreq = 1;
   currFreq = 1;
   ```
6.
   ```
   public int hoogsteFrequentie(){
      if (aantal == 0)
        return 0;
      int index = 1;
      int maxFreq = 1;
      int currFreq = 1;
      while (index < aantal){
         if (rij[index] == rij[index-1])
            currFreq++;
         else
            currFreq = 1;
         if (currFreq > maxFreq)
               maxFreq = currFreq;
         index++;
      }
      return maxFreq;
   }
   ```

   Toelichting: De lege rij wordt apart behandeld. Het bijwerken van maxFreq kan ook bij het ophogen van currFreq. Het kan echter niet aan het begin van een nieuwe reeks, want dan wordt de laatste reeks gemist.
7.
   – Als alle elementen zijn verwerkt, geeft de methode maxFreq terug. Dit is volgens de invariant de frequentie van het meest voorkomende element.
     De code binnen de lus zorgt dat `curFreq` en `maxFreq` de juiste waarde behouden als `index` wordt opgehoogd.
   – De variabele index loopt van 1 tot aantal. Dat is een beperkt aantal stappen. Terminatie is dus zeker.