TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen
Praktikum Grundlagen der Programmierung
F. Forster, T. Gawlitza, A. Flexeder

WS 2007/2008
Lösungsvorschläge zu Blatt 13
25. Januar 2008

Übungen zu Praktikum Grundlagen der Programmierung

Aufgabe 58 Applets: Album (Lösungsvorschlag)

a) HTML-Code: `album.html`

```
<html>
  <head>
    <title>Album</title>
  </head>
  <body>
    <h1>Album</h1>
    <applet codebase="." code="Album.class" width=500  height=300
        alt="Your␣browser␣has␣to␣be␣Java-enabled␣to␣see␣the␣applet!">
        <param name="urls" value="mac.jpeg:tux.jpeg:windows.jpeg"/>
    </applet>
  </body>
</html>
```

b) Applet-Code: `Album.java`

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.util.List;

public class Album extends Applet implements ActionListener {
    int crt = 0;
    Label label;
    Button prev, next;
    String[] names;
    Image[] imgs;

    public void init() {
        List<String> urls = new LinkedList<String>();
        StringTokenizer t = new StringTokenizer(getParameter("urls"), ":");
        while (t.hasMoreTokens())
            urls.add(t.nextToken());

        setBackground(Color.white);
        setFont(new Font("SansSerif", Font.BOLD, 18));
        label = new Label();
```

```
        add(label);

        prev = new Button("Prev");
        prev.setBackground(Color.orange);
        prev.addActionListener(this);
        add(prev);

        next = new Button("Next");
        next.setBackground(Color.orange);
        next.addActionListener(this);
        add(next);

        names = new String[urls.size()];
        imgs = new Image[urls.size()];
        int id = 0;
        for(String url : urls) {
            names[id] = url;
            id++;
        }
        label.setText(names[0]);
    }

    public void paint(Graphics page) {
        if (imgs[crt] == null)
            imgs[crt] = getImage(getDocumentBase(), names[crt]);
        page.drawImage(imgs[crt],100,100,this);
    }

    public void actionPerformed(ActionEvent e) {
        if(e.getSource()==next)
            crt = (crt+1)%imgs.length;
        else
            crt = (crt+imgs.length-1)%imgs.length;
        label.setText(names[crt]);
        repaint();
    }
}
```

Test durch Aufrufen der HTML-Datei im WWW-Browser oder mit `appletviewer`.

Aufgabe 59 (Ü) Tabellenkalkulation

a)

```
public abstract class Expr {
  public abstract int eval();
}

public class Const extends Expr {
  private int value;

  public Const(int v) {
    value = v;
  }

  public int eval() {
    return value;
  }

}

public class Add extends Expr {

  private Expr lhs, rhs;

  public Add(Expr x, Expr y) {
            lhs=x;
        rhs=y;
  }

  public int eval() {
    return lhs.eval() + rhs.eval();
  }
}

public class UnMinus extends Expr {
    private Expr e;
    public UnMinus(Expr x) {
        e=x;
    }

    public int eval() {
        return -e.eval();
    }
}

public class Ref extends Expr {

    private Tabelle tab;
    private int col;
    private int row;

    public Ref(Tabelle t, int r, int c) {
        tab = t;
        col = c;
        row = r;
    }
```

```
    public int eval() {
        return tab.evalZelle(row,col);
    }
}
```

b)

```
public class Tabelle {

    private final int ROWS = 30;
    private final int COLS = 30;
    private Expr[][] zellen = new Expr[ROWS][COLS];
    private int[][] values = new int[ROWS][COLS];
    private boolean[][] isValid = new boolean[ROWS][COLS];

    public Expr get(int i, int j) {
        if (i >= 0 && i < ROWS && j >= 0 && j < COLS)
            return zellen[i][j];
        else
            return null;
    }

    public int evalZelle(int i, int j) {
        if (i >= 0 && i < ROWS && j >= 0 && j < COLS && zellen[i][j] != null) {
            if (!isValid[i][j]) {
                values[i][j] = zellen[i][j].eval();
                isValid[i][j] = true;
            }
            return values[i][j];
        }
        throw new IllegalArgumentException("Cell␣not␣set");
    }

    public void setExpression(int i, int j, Expr e) {
        // set cell
        if (i >= 0 && i < ROWS && j >= 0 && j < COLS)
            zellen[i][j] = e;

        // invalid cache
        for (int a = 0; a < ROWS; a++) {
            for (int b = 0; b < COLS; b++) {
                isValid[a][b] = false;
            }
        }
    }

    public static void main(String[] args) {
        Tabelle t = new Tabelle();
        t.setExpression(1,1, new Const(1));
        t.setExpression(2, 3, new Ref(t,1,1));
        System.out.println(t.evalZelle(2,3));
        t.setExpression(1,1, new Const(3));
        System.out.println(t.evalZelle(2,3));
    }

}
```