

Musterlösung 10

Aufgabe 1 (Infix/Postfix)

Diese Lösung fängt einige aber nicht alle Fehler im Aufbau ab. Z.B. wird ab+ auch akzeptiert. Haben Sie Testfälle mit abgeben?

```
package U10.A1;

import java.util.EmptyStackException;
import java.util.Stack;

public class Infix {

    /**
     * liefert zu input äquivalenten Postfixausdruck bzw. SyntaxErrorException
     * analog zu 5.1.3.2
     */
    static String infix2postfix(String input) throws SyntaxErrorException {

        Stack<Character> operators = new Stack<Character>();
        char c;
        int operands = 0; // Zähler für Operanden, zur Fehlererkennung
        StringBuilder postfix = new StringBuilder();
        operators.push('('); // Ausdruck künstlich einklammern

        for (int i = 0; i <= input.length(); i++) {

            if (i == input.length()) {
                c = ')';
            } else {
                c = input.charAt(i);
            }

            if (Character.isLetter(c)) {
                postfix.append(c);
                operands++;
            }

            } else if (OPS.indexOf(c) >= 0) { // OPS enthält c
                while (!weaker(operators.peek(), c)) {
                    try {
                        char op = operators.pop();
                        if (operands < 2)
                            throw new SyntaxErrorException(
                                "Missing operand near " + c + " at "
                                    + (i + 1));
                        postfix.append(op);
                        operands--;
                    } catch (EmptyStackException e) {
                        throw new SyntaxErrorException("Missing operator near "
                            + c + " at " + (i + 1));
                    }
                }
            }
        }
    }
}
```

```

        }
    }

    if (c == ')') {
        operators.pop(); // '(' verwerfen
        if (i != input.length() && operators.size() == 0)
            throw new SyntaxErrorException(
                "Misaligned paranthesis.");
    } else
        operators.push(c);
} else {
    throw new SyntaxErrorException("invalid char '\" + c + "\" at "
        + i);
}
}

if (operators.size() != 0) {
    throw new SyntaxErrorException("Too many operators.");
}

if (operands != 1) {
    throw new SyntaxErrorException("Missing operator(s).");
}

return postfix.toString();
}

/** gültige Operatoren */
private static final String OPS = "+-*/^()";

/** liefert Operatorvorrang gemäß 5.1 Seite 20 (bis auf ^) */
private static boolean weaker(char a, char b) {
    if (a == '(')
        return true;
    if (b == ')')
        return false;
    if (a == '^' && b == '^')
        return true; // rechtsassoziativ
    return WEAKER.indexOf(simplify(a)) < WEAKER.indexOf(simplify(b));
}

private static final String WEAKER = "+*^(";

private static char simplify(char c) {
    if (c == '-')
        return '+';
    if (c == '/')
        return '*';
    return c;
}
}

```

Aufgabe 2 (Iteratoren)

```
7 interface Traversable<T> {
8     void map(UnaryFunction<T, T> f);
9
10    <V> V fold(BinaryFunction<T, V, V> f, V init);
11
12    List<T> filter(Predicate<T> p);
13 }
14
15 interface UnaryFunction<X, Y> {
16     Y of(X x);
17 }
18
19 interface BinaryFunction<X, Y, Z> {
20     Z of(X x, Y y);
21 }
22
23 interface Predicate<T> {
24     boolean satisfied(T x);
25 }
26
27 class L33tSpeak implements UnaryFunction<String, String> {
28     public String of(String x) {
29         return x.replaceAll("(?i)elite", "l33t").replaceAll("[eE]", "3")
30             .replaceAll("[iI]", "1").replaceAll("[aA]", "4");
31     }
32 }
33
34 class CountChars implements BinaryFunction<String, Integer, Integer> {
35     public Integer of(String x, Integer y) {
36         return y + x.length();
37     }
38 }
39
40 class NoSpace implements Predicate<String> {
41     public boolean satisfied(String x) {
42         return !x.contains(" ");
43     }
44 }
```

```

46 public class TraversableStack<T> extends Stack<T> implements Traversable<T> {
47
48     public List<T> filter(Predicate<T> p) {
49         Vector<T> res = new Vector<T>();
50         for (int i = 0; i < elementCount; i++) {
51             T t = (T) elementData[i]; // Cast OK, Elemente über korrekt ge-
52             // typpte Methoden hinzugefügt
53             if (p.satisfied(t))
54                 res.add(t);
55         }
56         return res;
57     }
58
59     public <V> V fold(BinaryFunction<T, V, V> f, V init) {
60         for (int i = 0; i < elementCount; i++)
61             init = f.of((T) elementData[i], init); // Cast OK
62         return init;
63     }
64
65     public void map(UnaryFunction<T, T> f) {
66         for (int i = 0; i < elementCount; i++)
67             elementData[i] = f.of((T) elementData[i]); // Cast OK
68     }
69
70
71     public static void main(String[] args) {
72         TraversableStack<String> ts = new TraversableStack<String>();
73         ts.add("Testing our Elite Map");
74         ts.add("NoSpace");
75         System.out.println("Before: " + ts);
76         ts.map(new L33tSpeak());
77         System.out.println("L33t  : " + ts);
78         System.out.println("Chars : " + ts.fold(new CountChars(), 0));
79         System.out.println("Spaces: " + ts.filter(new NoSpace()));
80     }
81 }

```

Aufgabe 3 (Konfigurationen mit Java)

<http://www.inf.fu-berlin.de/lehre/WS05/ALP3/loesungen/U10.A3.zip>

Model.java:

<pre> /** Exchange rate. */ private float exchangeRate; /** * Initializes the model of the * {@linkplain Converter currency converter component}. */ public Model() { // set default exchange rate this.exchangeRate = 1; } </pre>		<pre> /** * Initializes the model of the * {@linkplain Converter currency converter component}. */ public Model() { // determine preferences Preferences pref = Preferences.userNodeForPackage(Main.class); // set default exchange rate this.exchangeRate = pref.getFloat("rate", 1); } </pre>
--	--	--

Controller.java

<pre> public QuitController(Converter converter) { // set attribute this.converter = converter; } @Override public void windowClosing(WindowEvent event) { // close frame and release its resources this.converter.getView().getFrame().dispose(); } } </pre>		<pre> @Override public void windowClosing(WindowEvent event) { // determine preferences Preferences pref = Preferences.userNodeForPackage(Main.class); // save exchange rate pref.putFloat("rate", this.converter.getModel().getExchangeRate()); // save window position Point pos = this.converter.getView().getFrame().getLocation(); pref.putInt("left", pos.x); pref.putInt("top", pos.y); // close frame and release its resources this.converter.getView().getFrame().dispose(); } </pre>
--	--	--

View.java:

<pre> /** Status bar at the bottom of the frame. */ private JLabel statusBar; /** * Initializes the view of the * {@linkplain Converter currency converter component}. */ public View() { // create and configure frame this.frame = new JFrame(); this.frame.setTitle("Currency converter"); this.frame.setLocation(30, 40); this.frame.setResizable(false); this.frame.setDefaultCloseOperation(DO_NOTHING_ON_CLOSE); this.frame.getContentPane().setLayout(new BorderLayout()); } </pre>		<pre> /** * Initializes the view of the * {@linkplain Converter currency converter component}. */ public View() { // determine preferences Preferences pref = Preferences.userNodeForPackage(Main.class); // create and configure frame this.frame = new JFrame(); this.frame.setTitle("Currency converter"); this.frame.setLocation(pref.getInt("left", 30), pref.getInt("top", 40)); this.frame.setResizable(false); this.frame.setDefaultCloseOperation(DO_NOTHING_ON_CLOSE); this.frame.getContentPane().setLayout(new BorderLayout()); } </pre>
---	--	--