

## 1 Arbeiten mit den Standardklassen String und StringBuilder

*Ziel: Erzeugen von Objekten der Klassen String und StringBuilder; Benutzung von Methoden der Klassen String und StringBuilder; Benutzung der Java-Dokumentation*

1.1. Schreiben Sie ein Java-Programm, das verschiedene String-Methoden am Beispiel des Strings "Hurra, jetzt bin ich ein Java-Programmierer!" anwendet:.

- Geben Sie nur die ersten 5 Zeichen des Strings am Bildschirm aus.
- Setzen Sie den gesamten Text in Großbuchstaben um und geben ihn aus.
- Ersetzen Sie alle 'i' durch '\_' und zeigen das Ergebnis an.
- Geben Sie nur die Hälfte des Strings aus.
- Geben Sie den Text vom ersten 'j' bis zum letzten 'r' aus.

```
2* * A0101StringOps.java
5 package oopProgStdClass;
6
7 /**
8  * @author stk
9  * Kurzbeschreibung: Anwendung verschiedener String Objektmethoden
10 */
11 public class A0101StringOps
12 {
13     * @param args
14     public static void main(String[] args)
15     {
16         String sText = "Hurra jetzt bin ich eine Java-Programmierer!";
17         String sErg;
18
19         sErg = sText.substring(0, 5);
20         System.out.println(sErg);
21         sErg = sText.toUpperCase();
22         System.out.println(sErg);
23         sErg = sText.replace('i', '_');
24         System.out.println(sErg);
25         sErg = sText.substring(0, sText.length()/2);
26         System.out.println(sErg);
27         sErg = sText.substring(sText.indexOf('j'), sText.lastIndexOf('r') + 1);
28         System.out.println(sErg);
29     }
30 }
31
32
33
34 }
```

1.2. Lesen Sie 2 Strings von der Tastatur ein.

- Wenn die beiden Strings identisch sind, geben Sie aus „Die Strings sind identisch“.
- Wenn sie ungleich sind, geben Sie die Strings lexikalisch sortiert aus.

```

2* * A0102StringVergleich.java
5 package oopProgStdClass;
6
7 import input.Eingabe;
8
9 /**
10  * @author stk
11  * Kurzbeschreibung: Vergleich von Strings
12  */
13 public class A0102StringVergleich
14 {
15     * @param args
16     public static void main(String[] args)
17     {
18         String s1;
19         String s2;
20
21         s1 = Eingabe.getString("String1 = ");
22         s2 = Eingabe.getString("String2 = ");
23
24         if (s1.equals(s2))
25         {
26             System.out.println("Die Strings sind identisch!");
27         }
28         else
29         {
30             if (s1.compareTo(s2) < 0)
31             {
32                 System.out.println(s1 + ", " + s2);
33             }
34             else
35             {
36                 System.out.println(s2 + ", " + s1);
37             }
38         }
39     }
40 }

```

### 1.3. Programm "Passworttest":

**Problemstellung:** Ein sicheres Passwort sollte mindestens 10 Zeichen lang sein und sowohl Buchstaben und Ziffern als auch Sonderzeichen enthalten. Bei manchen Systemen wird diese Regel beim Anlegen eines Passworts geprüft. Zu Testzwecken soll in einem Konsolprogramm die Sicherheit eines Passworts geprüft werden.

Im Programm wird das zu prüfende Passwort von der Tastatur eingelesen. Das Programm gibt je nach Güte des Passworts folgendes auf dem Bildschirm aus:

```

Programm zum Testen der Güte eines Passwortes

Wie lautet das zu prüfende Passwort ? Passwort
|

Ihr Passwort ist unsicher!
Es sollte mindestens 10 Zeichen lang sein
und mindestens 1 Ziffer, 1 Buchstabe
und 1 Sonderzeichen enthalten!
"Passwort" enthält 8 Zeichen, 0 Ziffern,
8 Buchstaben und 0 Sonderzeichen

```

```

Programm zum Testen der Güte eines Passwortes

Wie lautet das zu prüfende Passwort ? SeTP@W#2018

Ihr Passwort "SeTP@W#2018" ist o.k.

```

```

2* * A0103PasswortTest.java
5 package oopProgStdClass;
6
7 import input.Eingabe;
8
9 /**
10  * @author stk
11  * Kurzbeschreibung: Test die Passwort Qualität
12  */
13 public class A0103PasswortTest
14 {
15     * @param args
16     public static void main(String[] args)
17     {
18         final int MINLAENGE = 10; // Minimale Passwortlänge
19         int i=0; // Schleifenzähler
20         int ilaenge = 0;
21         int iZiffern = 0; // Anzahl der Ziffern
22         int iBuchst = 0; // Anzahl der Buchstaben
23         String iPw; // Verweisvariable auf Stringobjekt
24
25         System.out.println("Programm zum Testen der Güte eines Passwortes\n");
26
27         System.out.printf("Wie lautet das zu prüfende Passwort ? ");
28
29         iPw = Eingabe.getString();
30
31         ilaenge = iPw.length();
32
33         for (i = 0; i < ilaenge; i++)
34         {
35             if (Character.isDigit(iPw.charAt(i))) iZiffern++;
36             else
37                 if (Character.isLetter(iPw.charAt(i))) iBuchst++;
38         }
39
40         if (ilaenge >= MINLAENGE && iZiffern >= 1 && iBuchst >= 1
41             && (ilaenge - iBuchst - iZiffern) >= 1)
42             System.out.printf("\n\nIhr Passwort \"%s\" ist o.k.\n", iPw);
43         else
44         {
45             System.out.printf("\n\nIhr Passwort ist unsicher!\n" +
46                 "Es sollte mindestens 10 Zeichen lang sein\n" +
47                 "und mindestens 1 Ziffer, 1 Buchstabe\n" +
48                 "und 1 Sonderzeichen enthalten!\n");
49             System.out.printf("\n\n%s" enthält %d Zeichen, %d Ziffern, %d Buchstaben" +
50                 "und %d Sonderzeichen",
51                 iPw, ilaenge, iZiffern, iBuchst, ilaenge - iBuchst - iZiffern);
52         }
53     }
54 }

```

1.4. Die Klassenmethode `zaehleWoerter` soll erstellt werden. Sie bekommt als Parameter einen String übergeben und soll dann die darin enthaltenen Wörter zählen. Die Anzahl wird als Rückgabewert zurückgegeben. Als Trennzeichen zwischen 2 Wörtern zählen die sogenannten „White-Space-Characters“, das sind  (Leerzeichen) und `\t` (Tabulatorzeichen).

Berücksichtigen Sie auch folgende Fälle:

- Zwischen zwei Wörtern sind mehrere Trennzeichen
- die Zeichenkette ist leer

- die Zeichenkette beginnt mit einem oder mehreren Trennzeichen
- die Zeichenkette endet mit einem oder mehreren Trennzeichen

Testen Sie die Methode, indem Sie in Main einen Text von der Tastatur einlesen und mit Hilfe der Methode die Anzahl der Wörter bestimmen.

```
2* * A0104WoerterZaehlen.java
5 package oopProgStdClass;
6 import input.Eingabe;
7
8 /**
9  * @author stk
10  * Kurzbeschreibung: Wörter in einem String zählen
11  */
12 public class A0104WoerterZaehlen
13 {
14     * @param args
15     public static void main(String[] args)
16     {
17         String sText;
18
19         sText = Eingabe.getString("Geben Sie einen Text ein damit die Wörter "
20             + "gezählt werden können");
21
22         System.out.printf("Es waren %d Wörter im Text", zaehleWoerter(sText));
23     }
24
25     public static int zaehleWoerter(String sText)
26     {
27         int iWoerter = 0;
28         Boolean bWarleer = true;
29
30         for (int i = 0; i < sText.length(); i++)
31         {
32             if (Character.isWhitespace(sText.charAt(i)))
33             {
34                 bWarleer = true; //leerzeichen ist da
35             }
36             else
37             {
38                 if (bWarleer == true)
39                 {
40                     bWarleer = false;
41                     iWoerter++; // zählen bei Wortbeginn
42                 }
43             }
44         }
45         return iWoerter;
46     }
47 }
48
49 }
```



- 1.5. Ein Programm soll eine Zeichenkette von der Tastatur einlesen und die Zeichenkette in einer zweiten String-Variablen in umgekehrter Reihenfolge abspeichern. Die umgekehrte Zeichenfolge soll dann am Bildschirm angezeigt werden. Tipp: Initialisieren Sie die zweite Variable mit einem leeren String und erstellen dann den umgekehrten String durch Konkatination (Operator '+').

Begründen Sie, warum der Datentyp String für dieses Programm aus Performanzgründen sehr schlecht geeignet ist.

```

2* * A0105StringReverse.java
5 package oopProgStdClass;
6 import input.Eingabe;
7
8 /**
9  * @author stk
10  * Kurzbeschreibung: Text rückwärts speichern und
11  *                   anzeigen
12  */
13 public class A0105StringReverse
14 {
15     * @param args
16     public static void main(String[] args)
17     {
18         String s1;
19         String s2 = "";
20         int i;
21
22         s1 = Eingabe.getString("Text = ");
23
24         for (i = 0; i < s1.length(); i++)
25         {
26             s2 = s1.charAt(i) + s2;
27         }
28
29         System.out.println(s2);
30     }
31 }

```

- 1.6. Die Klassenmethode mit der Signatur  
`public static String generiereAnmeldeName(String sIn)`  
soll programmiert werden. Der Parameter String enthält einen Vor- und Nachname, Bsp.  
„Otto Klotz“.

Der Rückgabewert ist der gewünschte Anmeldename:

- Der Anmeldename besteht immer aus acht Zeichen und enthält nur standard ASCII Zeichen, d.h. keine deutschen Umlaute oder 'ß'. Der Anmeldename enthält nur Kleinbuchstaben.
- Die ersten sechs Zeichen des Anmeldenames sind identisch mit den ersten sechs Zeichen des Nachnamens.
- Falls der Nachname kürzer als sechs Zeichen ist, werden die fehlenden Zeichen mit dem Zeichen 'x' aufgefüllt.
- Das 7. und 8. Zeichen des Login-Namens ist der erste und zweite Buchstabe des Vornamens. Annahme: Vornamen haben immer mindestens drei Zeichen.
- Bsp. Rückgabewert: „klotzot“