gds2 Gottlieb-Daimler-Schule 2
Technisches Schulzentrum Sindelfingen
mit Abteilung Akademie für Datenverarbeitung

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 letzen `'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 {
14      * @param args
17     public static void main(String[] args)
18     {
19         String sText = "Hurra jetzt bin ich eine Java-Programmierer!";
20         String sErg;
21
22         sErg = sText.substring(0, 5);
23         System.out.println(sErg);
24         sErg = sText.toUpperCase();
25         System.out.println(sErg);
26         sErg = sText.replace('i', '_');
27         System.out.println(sErg);
28         sErg = sText.substring(0, sText.length()/2);
29         System.out.println(sErg);
30         sErg = sText.substring(sText.indexOf('j'), sText.lastIndexOf('r') + 1);
31         System.out.println(sErg);
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.

```java
/* A0102StringVergleich.java
package oopProgStdClass;

import input.Eingabe;

/**
 * @author stk
 * Kurzbeschreibung: Vergleich von Strings
 */
public class A0102StringVergleich
{
    /**
     * @param args
     */
    public static void main(String[] args)
    {
        String s1;
        String s2;

        s1 = Eingabe.getString("String1 = ");
        s2 = Eingabe.getString("String2 = ");

        if (s1.equals(s2))
        {
            System.out.println("Die Strings sind identisch!");
        }
        else
        {
            if (s1.compareTo(s2) < 0)
            {
                System.out.println(s1 + ", " + s2);
            }
            else
            {
                System.out.println(s2 + ", " + s1);
            }
        }
    }
}
```

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 Buchstabenund 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.
```

```java
/* A0103PasswortTest.java
package oopProgStdClass;

import input.Eingabe;

/**
 * @author stk
 * Kurzbeschreibung: Test die Passwort Qualität
 */
public class A0103PasswortTest
{
    /**
     * @param args
     */
    public static void main(String[] args)
    {
        final int MINLAENGE = 10; // Minimale Passwortlänge
        int i=0;                  // Schleifenzähler
        int iLaenge = 0;
        int iZiffern = 0;         // Anzahl der Ziffern
        int iBuchst  = 0;         // Anzahl der Buchstaben
        String iPw;               // Verweisvariable auf Stringobjekt

        System.out.println("Programm zum Testen der Güte eines Passwortes\n");

        System.out.printf("Wie lautet das zu prüfende Passwort ? ");

        iPw = Eingabe.getString();

        iLaenge = iPw.length();

        for (i = 0; i < iLaenge; i++)
        {
            if (Character.isDigit(iPw.charAt(i))) iZiffern++;
            else
                if (Character.isLetter(iPw.charAt(i))) iBuchst++;
        }

        if (iLaenge >= MINLAENGE && iZiffern >= 1 && iBuchst >= 1
            && (iLaenge - iBuchst - iZiffern) >=1)
            System.out.printf("\n\nIhr Passwort \"%s\" ist o.k.\n", iPw);
        else
        {
            System.out.printf("\n\nIhr Passwort ist unsicher!\n" +
                "Es sollte mindestens 10 Zeichen lang sein\n" +
                "und mindestens 1 Ziffer, 1 Buchstabe\n" +
                "und 1 Sonderzeichen enthalten!\n");
            System.out.printf("\"%s\" enthält %d Zeichen, %d Ziffern,\n %d Buchstaben" +
                "und %d Sonderzeichen",
                iPw, iLaenge, iZiffern, iBuchst, iLaenge - iBuchst - iZiffern);
        }
    }
}
```

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, in dem 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 {
15*      * @param args
18      public static void main(String[] args)
19      {
20          String sText;
21
22          sText = Eingabe.getString("Geben Sie einen Text ein damit die Wörter "
23                  + "gezählt werden können");
24
25          System.out.printf("Es waren %d Wörter im Text", zaehleWoerter(sText));
26      }
27
28      public static int zaehleWoerter(String sText)
29      {
30          int iWoerter = 0;
31          Boolean bWarleer = true;
32
33          for (int i = 0; i < sText.length(); i++)
34          {
35              if (Character.isWhitespace(sText.charAt(i)))
36              {
37                  bWarleer = true;  //lehrzeichen ist da
38              }
39              else
40              {
41                  if (bWarleer == true)
42                  {
43                      bWarleer = false;
44                      iWoerter++;  // zählen bei Wortbeginn
45                  }
46              }
47          }
48          return iWoerter;
49      }
50 }
```

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 Konkatenation (Operator `'+'`).

Begründen Sie, warum der Datentyp String für dieses Programm aus Perfomanzgründen 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 {
16*     * @param args
19      public static void main(String[] args)
20      {
21          String s1;
22          String s2 = "";
23          int i;
24
25          s1 = Eingabe.getString("Text = ");
26
27          for (i = 0; i < s1.length(); i++)
28          {
29              s2 = s1.charAt(i) + s2;
30          }
31
32          System.out.println(s2);
33      }
34 }
```

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 Anmeldenamen besteht immer aus acht Zeichen und enthält nur standard ASCII Zeichen, d.h. keine deutschen Umlaute oder ‘ß‘. Der Anmeldenamen enthält nur Kleinbuchstaben.
- Die ersten sechs Zeichen des Anmeldenamen 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: „klotzxot“

Tipp: verwenden Sie die String-Methoden: `replace, toLowerCase, indexOf, substring, length` und die StringBuilder-Methode `append`.
Testen Sie die Methode, in dem Sie in Main einen Namen einlesen und den erzeugten Anmeldenamen auf dem Bildschirm ausgeben.

```java
/*
 * A0106AnmeldeName.java
 */
package oopProgStdClass;
import input.Eingabe;
/**
 * @author stk
 *    Kurzbeschreibung: Erzeugung eines Benutzernames aus
 *                      Vor- und Nachname
 */
public class A0106AnmeldeNameOhneSplit
{
    /**
     * @param args
     */
    public static void main(String[] args)
    {
        String sIn, sOut;

        System.out.println("Start des Programms zum Konvertieren "
                           + "von Namen in Anmeldenamen\n");
        System.out.printf("Geben Sie Ihren Vor- und Nachnamen ein: ");

        sIn = Eingabe.getString(); // Eine Zeile einlesen

        sOut = A0106AnmeldeNameOhneSplit.generiereAnmeldeName(sIn);

        System.out.printf("Der Anmeldename von %s lautet: %s\n", sIn, sOut);
    }

    public static String generiereAnmeldeName(String sIn)
    {
        StringBuilder sbOut;
        String sVorname, sNachname;
        int iPosition;
        int i;

        sIn = sIn.toLowerCase();

        sIn = sIn.replace("ä", "ae");
        sIn = sIn.replace("ö", "oe");
        sIn = sIn.replace("ü", "ue");
        sIn = sIn.replace("ß", "ss");

        iPosition = sIn.indexOf(' '); // Position des Trennzeichens
        sVorname = sIn.substring(0, iPosition);
        sNachname = sIn.substring(iPosition + 1, sIn.length());

        if (sNachname.length() > 6)
        {
            sbOut = new StringBuilder(sNachname.substring(0, 6));
        }
        else
        {
            sbOut = new StringBuilder(sNachname);
            for (i = 0; i < 6 - sNachname.length(); i++)
            {
                sbOut.append("x");
            }
        }

        sbOut.append(sVorname.substring(0, 2));

        return sbOut.toString();
    }

}
```

1.7. Wie Aufgabe 1.5, nur dass für das Umkehren des Textinhalts die Methode `reverse` der Klasse StringBuilder verwendet wird.

```
 2 * A0105StringReverse.java
 5 package oopProgStdClass;
 6
 7 import input.Eingabe;
 8
10 * @author stk
15 public class A0105StringBuilderReverse
16 {
17
19      * @param args
22     public static void main(String[] args)
23     {
24         String s1;
25         StringBuilder sb;
26
27         s1 = Eingabe.getString("Text = ");
28
29         sb = new StringBuilder(s1);
30         System.out.println(sb.reverse());
31     }
32 }
```

1.8. Ein Programm gibt eine Reihe von Wörtern auf dem Bildschirm aus und frägt dann den Benutzer, welches der Wörter nicht wirklich in die Reihe gehört. Bsp. für Wortreihen:
`"Klasse Objekt Methode Weihnachtsbaum"`
`"bytes ints shorts bermudas doubles"`
`"i12 Java Freizeit ADV"`
Es wird dann zunächst geprüft, ob das vom Benutzer vorgeschlagene Wort überhaupt im Text vorkommt (String-Objektmethode `indexOf()`). Dann wird geprüft, ob das vorgeschlagene Wort das „falsche“ Wort ist. Wenn ja, wird es aus der Reihe entfernt und der korrigierte String angezeigt. In den anderen Fällen kommt jeweils eine passende Fehlermeldung.

```java
 * A0108WortReihe.java
package oopProgStdClass;
import input.Eingabe;
 * @author stk
public class A0108WortReihe
{
    /**
     * @param args
     * Kurzbeschreibung: Welcher Begriff gehört nicht in die jeweilige Reihe ?
     * Löschen Sie diesen Begriff .
     * "Klasse Objekt Methode Weihnachtsbaum"
     * "bytes ints shorts bermudas doubles"
     * "I12 Java Freizeit ADV"
     */
    public static void main(String[] args)
    {
        String sBegriff;
        StringBuilder sb1 = new StringBuilder("Klasse Objekt Methode Weihnachtsbaum");
        int iIndex;

        sBegriff = Eingabe.getString("Welcher Begriff gehört nicht in die Reihe?\n"+
                                     sb1 + " ");
        iIndex = sb1.indexOf(sBegriff);
        if (iIndex != -1)
        {
            if (sBegriff.equals("Weihnachtsbaum"))
            {
                sb1.delete(iIndex, iIndex+sBegriff.length());
                System.out.println("Richtig.");
                System.out.println(sb1.toString());
            }
            else
            {
                System.out.println("Falsch.");
            }
        }
        else
        {
           System.out.println(sBegriff+" nicht gefunden");
        }
    }

}
```

1.9. Bringen Sie diese wahren Aussagen jeweils in die richtige Reihenfolge, indem Sie ein StringBuilder-Objekt mit dem jeweiligen Text abbauen und ein neues StringBuilder-Objekt entsprechend aufbauen. Lassen Sie die Wörter jeweils von Tastatur eingeben:
`"Programmieren das Java mit Spaß macht"`
`"Weihnachtsferien den in Tag jeden ich programmiere"`

`Beispieldialog:`

```
Bringen Sie diese Begriffe jeweils in die richtige Reihenfolge:
Programmieren das Java mit Spaß macht
Bitte Sortieren :
1. Wort: das
Programmieren  Java mit Spaß macht
das
2. Wort: Programmieren
  Java mit Spaß macht
das Programmieren

...

5. Wort: macht
    Spaß
das Programmieren mit Java macht
6. Wort: Spaß

das Programmieren mit Java macht Spaß
Gut !
```

```java
/**
 * @author stk
 *         Kurzbeschreibung:Bringen Sie diese Begriffe jeweils in die richtige
 *         Reihenfolge: "Programmieren das Java mit Spaß macht"
 */
public class A0109Reihenfolge
{
     * @param args
    public static void main(String[] args)
    {
        StringBuilder sbOrg = new StringBuilder("Programmieren das Java mit Spaß macht");
        StringBuilder sbNeu = new StringBuilder();
        String sWort = "";
        int iAnzWörter = 0;
        int iIndexVon = -1;

        System.out.println("Bringen Sie diese Begriffe jeweils in die richtige Reihenfolge:");
        System.out.println(sbOrg.toString());

        // Anzahl Wörter bestimmen
        iAnzWörter = A0104WoerterZaehlen.zaehleWoerter(sbOrg.toString());

        System.out.println("Bitte Sortieren :");
        for (int iAnz = 0; iAnz < iAnzWörter; iAnz++)
        {
             // Wort suchen...

            do // Wiederhole, falls Benutzer falsches Wort eingibt
            {
                sWort = Eingabe.getString(iAnz + 1 + ". Wort: ");
                if (sWort != null)
                    iIndexVon = sbOrg.indexOf(sWort); // Wort im Text suchen
            } while (iIndexVon == -1 || sWort == null);

            // aus Original löschen und an Neuen StringBuiler anhängen...
            sbOrg.delete(iIndexVon, iIndexVon + sWort.length());
            System.out.println(sbOrg);
            sbNeu.append(sWort + " ");
            System.out.println(sbNeu);
        }
        if (sbNeu.toString().equals
                                ("das Programmieren mit Java macht Spaß "))
        {
            System.out.println("Gut !");
        }
        else
        {
            System.out.println("Nicht gut");
        }
    }
}
```