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

Inhaltsverzeichnis

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.

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.

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.
```

Tipp: siehe Klassenmethoden **`isDigit`** und **`isLetter`** der Klasse **`Character`** in der Java Dokumentation.

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.

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.

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.

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

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.

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 !
```

2 Arbeiten mit der Standardklasse StringTokenizer

2.1. Informieren Sie sich an Hand der API-Dokumentation über die Arbeitsweise der Klasse `StringTokenizer`.

2.2. Von der Tastatur soll ein Satz eingelesen werden.

a. Ermitteln Sie mit Hilfe der Klasse `StringTokenizer` die Anzahl der einzelnen Wörter des Satzes (erkennbar an den Leerzeichen).
b. Geben Sie die Wörter einzeln in jeweils einer neuen Zeile am Bildschirm aus.
c. Setzen Sie die einzelnen Wörter des Satzes in einem neuen `StringBuilder`-Objekt wieder zusammen, indem Sie die einzelnen Wörter durchnummerieren, also z.B. „`1.: Hallo 2.: Welt …`" und geben danach das Ganze am Bildschirm aus.
d. Lesen Sie von Tastatur ein Zeichen ein, das als Trennzeichen betrachtet werden soll. Trennen Sie den eingelesenen Satz an diesem Trennzeichen und zeigen Sie die so enstandenen Teile („Tokens") am Bildschirm an.

2.3. Wie Aufgabe 1.4, nur dass zum Zählen der Wörter ein Objekt der Klasse `StringTokenizer` verwendet wird.

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

3 Arbeiten mit den Standardklassen LocalDate und LocalTime

3.1. Gegeben ist die Main-Methode in der Klasse `A0301KalenderTest_v2` (siehe Moodle)
In dieser Main-Methode werden verschiedene Klassenmethoden aus der Klasse `A0301KalenderKlasse_v2` aufgerufen.

Aufgabe:
Erstellen Sie die Klasse `A0301KalenderKlasse_v2` mit den Klassenmethoden, die in der Tabelle unten aufgelistet sind.
Zum Testen gehen Sie am besten so vor, dass Sie in der gegebenen Klasse in Main zunächst alles auskommentieren und dann versuchen Anweisung für Anweisung zum Laufen zu bringen, indem Sie die notwendigen Methoden implementieren.

`String getDatum(int iJahr, int iMonat, int iTag, int iFormat)`	liefert das übergebene Datum im Format `iFormat`:
	`0: "Freitag, 18.Dezember 2009"`
	`1: "18/12/09"`
	`2: "18.Dezember 2009"`
	`3: "18.Dezember.09"`
	`4: "18.12.09"`
	`5: "18/12/2009"`
	`6: "2009-12-18"`
	`7: "Dezember 09"`
	`8: "18-Dezember-09"`
	`9: "Dez.09"`

`String getDatum(int iJahr, int iMonat, int iTag)`	liefert das übergebene Datum im langen Format (FormatNr. 0)
`String getAktuellesDatum(int iFormat)` → ruft die Methode `getDatum(…)` für das aktuelle Datum auf	liefert das aktuelle Datum im Format `iFormat` → benutzt die Methode `getDatum(…)`
`String getAktuellesDatum()` → ruft die Methode `getAktuellesDatum(0)` auf	liefert das aktuelle Datum im langen Format (FormatNr. 0)
`String getAktWochentag()`	Gibt den Namen des aktuellen Wochentags als String zurück
`String getAktuelleUhrzeit()`	Liefert die aktuelle Uhrzeit im Format `"hh:mm:ss"`

`int getAktStunde()`
`int getAktMinute()`
`int getAktSekunde()`	

`void druckeKalenderMonat(int iJahr, int iMonat)`	`z.B. 12/2000:`
	`Dezember 2000`
	`Mo Di Mi Do Fr Sa So`
	`01 02 03`
	`04 05 06 07 08 09 10`
	`11 12 13 14 15 16 17`
	`18 19 20 21 22 23 24`
	`25 26 27 28 29 30 31`

`void druckeAktKalenderMonat()`	

4 Arbeiten mit der Standardklasse Random

4.1. Erstellen Sie ein Programm zum Testen der Klasse Random nach folgenden Vorgaben:

- Der Benutzer wird aufgefordert einen Initialwert („seed“) für die Folge der Pseudo-Zufallszahlen einzugeben. Gibt der Benutzer einen Wert ungleich -1 ein, dann wird das Random-Objekt mit dem Initialwertwert erzeugt, sonst ohne.
- Der Benutzer hat dann noch die Möglichkeit einen oberen Grenzwert für die Zufallszahlen vorzugeben.
- Das Programm erzeugt dann eine Folge von 100 ganzzahligen Zufallszahlen und gibt diese am Bildschirm aus; jeweils 10 Zahlen in einer Zeile.

  Testen Sie das Programm mit verschiedenen Eingaben. Was passiert bei Eingabe des selben Initialwertes und der selben oberen Grenze?

4.2. Erstellen Sie eine Klasse `Zufall` mit folgenden **Klassen**methoden:

`double getZufallDouble()`	Liefert einen `double`-Zufallswert im Bereich zwischen 0.0 <= z < 1.0 Das Random Objekt wird ohne Initialwert erzeugt.
`double getZufallDouble(double dVon, double dBis)`	Liefert einen `double`-Zufallswert im Bereich zwischen `dVon` <= z < `dBis`
`int getZufallInt ()`	Liefert einen `int`-Zufallswert aus dem Zahlenbereich aller möglichen `int`-Zahlen
`int getZufallInt (int iVon, int iBis)`	Liefert einen `int`-Zufallswert im Bereich zwischen `iVon` <= z <= `iBis`

4.3. Testen Sie die Methoden in eine Main Methode.

5 Arbeiten mit den Standardklassen Formatter und DecimalFormat und Wrapper Klassen

5.1. Erzeugen Sie einen "Zahlenpfeil" aus Zufallszahlen. z.B.

```
9
82
155
6225
96585
768535
44145
4077
675
33
3
```

Die Grundidee dabei ist, dass aus beliebig vielen Zufallszahlen zuerst eine einstellige, dann eine zweistellige usw. bis 6-stellig, dann rückwärts ermittelt und ausgegeben werden. Zur Ermittlung der Stellenzahl und zur Ausgabe auf dem Bildschirm wird die Zufallszahl in einen String umgewandelt.

5.2. Es sollen bis zu 10 double-Zufallszahlen ermittelt werden. Die genaue Anzahl wird zufällig ermittelt. Diese sollen in einem String verkettet werden, jeweils durch ; getrennt. Anschließend soll dieser String wieder in einzelne doubles getrennt werden, die dann mit 3 Vor- und 5 Nachkommastellen ausgegeben werden.
Beispiel:
Kette: `72.13726205311184 ;413.98616499178144 ;8.1345711326591 ;`

```
Nr. 1 072,13726
Nr. 2 413,98616
Nr. 3 008,13457
```

Hinweise:
Zur Erzeugung der Zufallszahen können die Methoden aus Aufgabe 4.2 aufgerufen werden.
Um die Zahlenkette wieder zu zerlegen ein StringTokenizer Objekt verwenden.
Um die Zahl in einer Stringvariablen wieder in einen int Umzuwandeln kann man die Klassenmethode Double.parseDouble() verwenden..
Die Ausgabe von Fließkommazahlen mit führenden Nullen funktioniert nicht mit printf(), deshalb eine DecimalFormat Objekt verwenden.