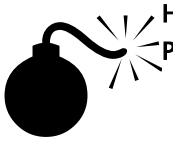

Halten Sie beim Programmieren rigoros ALLE aufgestellten Programmierregeln ein. Ihre Programme werden in Hinsicht auf die Einhaltung dieser Regeln kontrolliert.

1. Welche *Ausgabe* liefert folgendes Progrogrammfragment:

```
String titel = "Erlkönig";
String vers1 = "Wer reitet so spät durch Nacht und Wind";
int erscheinungsjahr = 1782;
vers1 = vers1.substring(0,11) + vers1.substring(14);
System.out.println(vers1);
titel = titel.substring(0,3) + "en" + titel.substring(3);
System.out.println(titel);
System.out.println(titel.compareTo(vers1) < 0);
System.out.println(titel + (int)(erscheinungsjahr + 2));
System.out.println(Character.toUpperCase(titel.charAt(titel.length() - 1)));
System.out.println(vers1.indexOf("reitet"));
System.out.println(vers1.substring(0,vers1.indexOf("Nacht")) +
   vers1.substring(vers1.indexOf("Nacht") + "Nacht".length() + 1));
```

2. Welche *Ausgabe* liefert folgendes Programmfragment:

```
String titel = "Erlkönig";
String vers1 = "Wer reitet so spät durch Nacht und Wind";
String vers2 = "Es ist der Vater mit seinem Kind.";
String suchString = "Nacht";
System.out.println(vers1.length() + " " + vers2.length());
System.out.println(suchString + " finden wir bei " + vers1.indexOf(suchString));
suchString = vers2.substring(11,16);
System.out.println(suchString);
vers2 = vers2.substring(0,vers2.indexOf(suchString)) +
   vers1.substring(vers1.length() - 4) +
   vers2.substring(vers2.indexOf(suchString) + suchString.length());
System.out.println(vers2);
```

3. Wie viele nicht mehr benötigte Strings werden von der Virtual Machine während der Programmdurchführung zerstört?

```
String titel = "Erlkönig";
for (int i = 0; i < titel.length(); i = i + 2)
   titel = titel + i;
```

4. Schreiben Sie ein Programm mit dem Namen ISOLatin1Zeichensatz, welches Ihnen – vom 33. bis zum 256. Zeichen – in der angegebenen Form den *ISO-Latin1-Zeichnsatz* ausgibt.

Dabei kann es sein, dass Ihr System das einige Zeichen nicht ausgeben kann. In diesem Fall wird ein *Fragezeichen* (?) angezeigt. Es sollen die einzelnen Zeichen direkt untereinander platziert werden.

```
32    33 !  34 "  35 #  36 $  37 %  38 &  39 '
40 (  41 )  42 *  43 +  44 ,  45 -  46 .  47 /
48 0  49 1  50 2  51 3  52 4  53 5  54 6  55 7
56 8  57 9  58 :  59 ;  60 <  61 =  62 >  63 ?
64 @  65 A  66 B  67 C  68 D  69 E  70 F  71 G
72 H  73 I  74 J  75 K  76 L  77 M  78 N  79 O
80 P  81 Q  82 R  83 S  84 T  85 U  86 V  87 W
88 X  89 Y  90 Z  91 [  92 \  93 ]  94 ^  95 _
96 `  97 a  98 b  99 c 100 d 101 e 102 f 103 g
...
240 ð 241 ñ 242 ò 243 ó 244 ô 245 õ 246 ö 247 ÷
248 ø 249 ù 250 ú 251 û 252 ü 253 ý 254 þ 255 ÿ
```

5. Schreiben Sie ein Programm mit dem Namen ZeichensatzAnalyse, welches – vom 33. bis zum 256. Zeichen – die *Buchstaben* (isLetter), *Zahlen* (isDigit), *Wortzwischenräume* (isWhitespace), *Groß-* und

```
isLetter
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z a b c d
e f g h i j k l m n o p q r s t u v w x y z ª µ º À Á Â Ã Ä
Å Æ Ç È É Ê Ë Ì Í Î Ï Ð Ñ Ò Ó Ô Õ Ö Ø Ù Ú Û Ü Ý Þ ß à á â ã
ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ø ù ú û ü ý þ ÿ
isDigit
0 1 2 3 4 5 6 7 8 9
isWhitespace
32
isLowerCase
a b c d e f g h i j k l m n o p q r s t u v w x y z ª µ º ß
à á â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ø ù ú û ü ý þ
ÿ
isUpperCase
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z À Á Â Ã
Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ð Ñ Ò Ó Ô Õ Ö Ø Ù Ú Û Ü Ý Þ
```

Kleinbuchstaben (`isLowerCase` und `isUpperCase`) in der angegebenen Form ausgibt. Dabei sollen von den Zwischenräumen der entsprechende Zahlencode ausgegeben werden (beispielsweise soll anstelle des Leerzeichens der Zahlencode des Leerzeichens angezeigt werden). Es sollen in einer Zeile jeweils 30 gefundene Zeichen ausgegeben werden.

6. Schreiben Sie ein Programm mit dem Namen `TextStatistik`, welches von einem eingegebenen Text die Anzahl der *Selbstlaute*, *Buchstaben*, *Leerzeichen* und *Zeichen* zählt. Verwenden Sie zur Eingabe des Textes die mitgelieferte Methode `readString` welche sich im Programm `TestScannerErweitert.java` befindet.

   Realisieren Sie die nebenstehende Benutzerschnittstelle.

```
Textstatistik
=============
Text:
Text muss mindestens ein Zeichen enthalten
Text: Rotkäppchen ging in den Wald

Anzahl Selbstlaute: 6
Anzahl Buchstaben: 24
Anzahl Leerzeichen: 4
Anzahl Zeichen: 28
```

7. Schreiben Sie ein Programm mit dem Namen `VerschlüsselungDrehen`, welches einen eingegebenen Text verschlüsselt, indem zuerst das *erste und letzte* Zeichen, dann das *zweite und vorletzte*, danach das *dritte und vorvorletzte* usw. ausgegeben wird, bis alle Zeichen ausgegeben wurden.

   Es soll dabei möglich sein, auch mehrere Texte hintereinander zu verschlüsseln. Verwenden Sie dazu die mitgelieferte Methode `readChar` die sich ebenfalls im Programm `TestScannerErweitert.java` befindet. Achten Sie beim Programmieren darauf, dass Sie die abgebildete Benutzerschnittstelle realisieren.

```
Verschlüsselung durch Drehen
============================
Text:
Text muss mindestens ein Zeichen enthalten
Text: Roberta
Verschlüsselt: Raotbre
Nochmal (j/n)? j
Text: Roberta ging in den Wald.
Verschlüsselt: R.odblearWt an egdi nngi
Nochmal (j/n)? n
```

8. Schreiben Sie ein Programm mit dem Namen `VerschluesselungCaesar`, welches durch die *„Cäsar-Verschlüsselung“* einen Text *ver- und entschlüsseln* kann.

   Informieren Sie sich selbständig darüber, wie der Algorithmus zur Cäsar-Verschlüsselung funktioniert.

   Das Programm soll die nebenstehende Benutzerschnittstelle haben und auf *Falscheingaben* reagieren können.

```
Verschlüsselung nach Cäsar
==========================
V>erschlüsseln, E>ntschlüsseln, A>bbruch: v
Text: veni vidi vinci
Verschiebung: 10
Verschlüsselt: FOXSFSNSFSXMS
V>erschlüsseln, E>ntschlüsseln, A>bbruch: e
Text: FOXSFSNSFSXMS
Verschiebung: 10
Entschlüsselt: VENIVIDIVINCI
V>erschlüsseln, E>ntschlüsseln, A>bbruch: a
```

9. Schreiben Sie ein Programm mit dem Namen `WortRatespiel` bei welchem ein Wort eingegeben werden kann, welches der Benutzer dann erraten soll, indem er Buchstaben dieses Wortes eingibt. Ob und an welcher Stelle die eingegebenen Buchstaben im Wort zu finden sind, wird dem Benutzer angezeigt.

   Das Programm soll zählen, in *wie vielen Schritten* das Wort erraten werden konnte. Weiters soll das Programm abfragen, ob ein *weiteres Spiel* begonnen werden soll.

```
Wortratespiel
=============
Gesuchtes Wort: scheibenwischer
Ihr Wort:       ...............
Buchstabe/Wort: e
Ihr Wort:       ...E..E......E.
Buchstabe/Wort: i
Ihr Wort:       ...EI.E..I...E.
Buchstabe/Wort: h
Ihr Wort:       ..HEI.E..I..HE.
Buchstabe/Wort: scheibenwischer
Sie haben in 4 Schritten das Wort erraten!
Nochmal (j/n)? n
```

Das Programm soll auf mögliche *Falscheingaben* reagieren können und zur Bearbeitung der Wörter ausschließlich Strings verwenden.

10. Schreiben Sie ein Programm mit dem Namen `Textverarbeitung`, welches schrittweise *vier* eingegebene Textzeilen zuerst

    - auf eine *fixe Länge kürzt*,
    - dann *unnötige Leerzeichen* am Beginn und am Ende aber auch zwischen den Wörtern entfernt und zum Schluss
    - auf die einzelnen Zeilen einen *Blocksatz* in vorgegebener Breite einstellt.

    Verwenden Sie beim Programmieren die bereitgestellte *Rohdatei* mit dem Namen `Textverarbeitung.java`. In dieser sollen Sie zuerst die Methoden `abschneiden`, `leerzeichenEntfernen` und `blocksatz` programmieren. Wie die Methoden arbeiten sollen, wird Ihnen in den *Kommentaren* beschrieben.

```
Textverarbeitung
================
1. Zeile:   Sein  oder  nicht sein,
2. Zeile: das  ist hier  die Frage.
3. Zeile:  Wer hat   an der Uhr gedreht? Zuviel
4. Zeile: ist es wirklich  schon spät?
>>Abschneiden
  Sein  oder  nicht sein,
das  ist hier  die Frage.
 Wer hat   an der Uhr gedreht?
ist es wirklich  schon spät?
>>Leerzeichen entfernen
Sein oder nicht sein,
das ist hier die Frage.
Wer hat an der Uhr gedreht?
ist es wirklich schon spät?
>>Blocksatz
Sein    oder    nicht    sein,
das   ist   hier   die   Frage.
Wer  hat  an  der Uhr gedreht?
ist  es  wirklich  schon spät?
```

    Die *Methodenköpfe* dürfen von Ihnen *nicht verändert* werden.