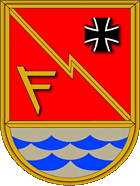
Schule Informationstechnik

der Bundeswehr



**Sprachausbildung Java**



**Übungen**

**„Wiederholungsaufgaben LN 1“**

# StringUtil

Definieren sie eine Klasse StringUtil von welcher keine Objekte erzeugt werden können. Schreiben Sie die Methoden toUpperCase und toLowerCase, welche einen Parameter vom Typ char oder String entgegennehmen und denselben Typ zurück liefern.

Die Methode toUpperString wandelt kleine Buchstaben in Großbuchstaben um. Alle anderen Zeichen werden nicht verändert.

Die Methode toLowerString wandelt große Buchstaben in kleine Buchstaben um. Alle anderen Zeichen werden nicht verändert.

Sie dürfen keine Methoden der Standardbibliothek verwenden, welche diese Funktionalität abbildet.

# PasswortChecker

Gute Passwörter sollten so aufgebaut sein, dass nicht leicht durch massives Ausprobieren aufgrund von Wörterbüchern und Zahlenfolgen erraten werden können.

Ein gutes Passwort sollte zum Beispiel mindestens

a) acht Zeichen,

b) zwei Buchstaben,

c) zwei Ziffern,

d) ein Gross- und ein Kleinbuchstabe,

e) ein Sonderzeichen enthalten und

f) mindestens eine Ziffer oder Sonderzeichen, soll sich innerhalb des Passwortes befinden. Also nicht direkt am Anfang oder am Ende stehen.

"AxB1&a8c" erfüllt zum Beispiel diese Kriterien, "1passw0rt" aber nicht (kein Sonderzeichen und kein Großbuchstabe).

Implementieren Sie eine Java-Klasse PasswortChecker mit einer Java-Methode, die überprüft, ob ein Passwort obige Kriterien erfüllt oder nicht. Das Passwort soll dabei als char-Feld gegeben sein.

Ein Sonderzeichen soll einfach ein Zeichen sein, dass keine Ziffer und kein Unicode-Buchstabe ist.

# EURO-Seriennummer

Euro-Banknoten haben eine eindeutige Seriennummer, die aus einem führenden Buchstaben, 10 Ziffern und einer Prüfziffer bestehen.

Beispiel: Z 6016220022 6 (Leerzeichen zur Übersicht hinzugefügt).

Der führende Buchstabe codiert die nationale Zentralbank (NZB), die den Geldschein in Umlauf gebracht hat. Sie wird NZB-Nummer genannt.

Die Prüfziffer berechnet sich wie folgt:

* Der Buchstabe wird durch seine Position im lateinischen Alphabet ersetzt (bei Z also 26)
* Es wird die Quersumme der Positionszahl und der 10 Ziffern berechnet (im Beispiel 2+6 + 6+0+1+6+2+2+0+0+2+2 = 29)
* Die Zahl wird mit Rest durch 9 geteilt (2)
* Der Rest wird von 8 subtrahiert. Das Resultat ist die Prüfziffer (6). Es sei denn es kam 0 dabei heraus, dann ist die Prüfziffer 9

Implementieren Sie eine Java-Methode, die für eine Seriennummer (gegeben als char-Array seriennummer) die Prüfziffer berechnet und zurückgibt.

Der Buchstabe ist immer groß. Fehlerfälle brauchen nicht berücksichtigt werden.

# int / boolean Arrays

Lesen Sie in der main – Methode eine Feldlänge ein. Definieren Sie dann zwei Felder mit dieser Länge, eines für int – Werte, eines für Werte vom Typ boolean.

Schreiben Sie eine Methode, die folgendes leistet: An den Positionen, an denen das boolean – Array den Wert false hat, wird im int – Array das Vorzeichen des Wertes gewechselt.

Beispiel:

int – Array: { 5, 6, 9, -3, 9, 0 }

boolean – Array: { true, false, true, true, false, true}

Ergebnis: { 5, -6, 9, -3, -9, 0}

# Analyse 1

Analysieren Sie folgenden Quellcode. Beschreiben Sie den Ablauf textuell und grafisch sowie die Ausgabe.

1. public class Aufgabe5 {
2. public static void main( String[] args ) {
3. int[] arr1 = new int[ 4 ];
4. arr1[ 0 ] = 3;
5. for ( int i = 1; i < arr1.length; i++ ) {
6. arr1[ i ] = arr1[ i - 1 ] << 1;
7. if ( arr1[ i ] % 6 == 0 ) {
8. arr1[ i ] = arr1[ i ] \* -1;
9. }
10. }
11. for ( int i = 0; i < arr1.length; i++ ) {
12. System.*out*.println( arr1[ i ] + " " );
13. }
14. }
15. }

# Analyse 2

Analysieren Sie folgenden Quellcode. Beschreiben Sie den Ablauf textuell und grafisch sowie die Ausgabe.

1. class Aufgabe6 {
2. public static void main( String[] args ) {
3. int[][] A3x1 = { { 1, 2 }, { 2, 3 }, { 3, 4 } };
4. for ( int[] a3 : A3x1 )
5. System.*out*.println( a3 );
7. int[][] A3x2 = new int[][]{ { 1, 2 }, { 2, 3 }, { 3, 4 } };
8. System.*out*.println( A3x2[ 0 ][ 1 ] );
10. int[][] A3x3 = new int[][]{ new int[]{ 1, 2 },
11. new int[]{ 2, 3 },
12. new int[]{ 3, 4 } };
14. for ( int i = 0; i < 3; i++ ) {
15. for ( int j = 0; j < 2; j++ ) {
16. System.*out*.print( A3x3[ I ][ j ] + " " );
17. }
18. System.*out*.println();
19. }
20. }
21. }

# McCarthy

Schreiben Sie die folgende mathematische Funktion (McCarthy 91) als Java-Methode:

Das umgebende Hauptprogramm soll die ganzzahligen Ergebnisse der Funktion für die Eingangswerte n im Wertebereich 90 bis 120 ausgeben.

# EKONS-Verfahren

Zur Systemsicherung werden verschiedene Verfahren zur Ermittlung von sogenannten Prüfziffern verwendet. Ein Verfahren ist das EKONS-Verfahren (**E**inheitlicher **Ko**nto**n**ummern**s**chlüssel).

Das Verfahren bildet die Summe aus den Ziffern einer Nummer, die mit einer bestimmten Gewichtung multipliziert werden. Die Gewichtung beträgt beginnend mit der ersten Ziffer abwechselnd 2 und 1. Von den so gebildeten Summanden wird dann lediglich die Quersumme zur Summenbildung verwendet. Die Prüfziffer wird dann durch modulare Teilung der Summe durch 10 ermittelt.

Beispiel: 34991815

(2\*3 + 1\*4 + 2\*9 + 1\*9 + 2\*1 + 1\*8 + 2\*1 + 1\*5 ) mod 10 =

( 6 + 4 + 18 + 9 + 2 + 8 + 2 + 5) mod 10 =

( 6 + 4 + 9 + 9 + 2 + 8 + 2 + 5) mod 10 =

45 mod 10 = 5

Schreiben Sie eine Methode, die die Prüfziffer zu einer beliebigen 4- bis 8-stelligen Nummer ermittelt. Ist das Format der Nummer falsch (falsche Zeichen, falsche Länge) ist -1 zurück zu geben.

# Vervollständigen

Implementieren Sie alle notwendigen Konstruktoren / Methoden um das Hauptprogramm lauffähig zu machen. Die Form der Beispielausgaben ist VERBINDLICH.

1. public class Main {
2. public static void main( String[] args ) {
3. final int MIN\_ZEILE = 1;
4. final int MAX\_ZEILE = 8;
5. final char MIN\_SPALTE = 'A';
6. final char MAX\_SPALTE = 'H';
7. Brett brett = new Brett( MAX\_ZEILE, MAX\_SPALTE );
8. Bauer b1 = new Bauer( "Fred", "weiss" );
9. brett.add( b1, new Koordinate( 2, 'A' ) ); // Positioniert Figur an der Koordinate auf dem Brett
10. Turm t1 = new Turm( "Tower", "schwarz" );
11. if ( !brett.add( t1, new Koordinate( 8, 'H' ) ) ) {
12. System.*err*.println( "Auf dem Feld steht bereits eine Figur!" );
13. }
14. Springer pferd = new Springer( "Pferd", "schwarz" );
15. if ( !brett.add( pferd ) ) { // Positioniert selbstständig auf die Ausgangsposition.
16. System.*err*.println(
17. "Alle Positionen für die Aufstellung eines schwarzen Springers sind bereits belegt." );
18. }
19. Laeufer l1 = new Laeufer( "Bolt", "weiss" );
20. brett.add( l1, new Koordinate( 1, 'C' ) );
21. Koenig king = new Koenig( "Gustav", "schwarz" );
22. brett.add( king, new Koordinate( 8, 'E' ) );
23. Dame dame = new Dame( "weiss" );
24. brett.add( dame, new Koordinate( 1, 'D' ) );
25. // Ausgabe (die Ausgabe aller anderen Figuren analog zum Bauern):
26. // Falls der Bauer auf Brett gesetzt wurde:
27. // Fred ( Bauer, weiss, Pos: 2, A )
28. // Falls der Bauer nicht auf dem Brett gesetzt wurde:
29. // Fred ( Bauer, weiss )
30. System.*out*.println( b1 );
31. // Ausgabe (Die Figuren werden zeilen- und spaltenweise aufsteigend ausgegeben, siehe Beispiel):
32. // Brett ( Anzahl Zeilen: 8, Anzahl Spalten: H )
33. // Bolt ( Laeufer, weiss, Pos: 1, C )
34. // Dame ( Dame, weiss, Pos: 1, D )
35. // Fred ( Bauer, weiss, Pos: 2, A )
36. // Pferd ( Springer, schwarz, Pos: 8, B )
37. // Gustav ( Koenig, schwarz, Pos: 8, E )
38. // Tower ( Turm, schwarz, Pos: 8, H )
39. System.*out*.println( brett );
40. }
41. }
42. public class Brett {
43. /\*\*
44. \* Array für die Schachfiguren auf dem Brett
45. \* 1. Dimension: Zeilen 1..\*
46. \* 2. Dimension: Spalten 'A'..\* (1..\*)
47. \*/
48. private Figur[][] brett;
49. }
50. public class Koordinate {
51. private char spalte;
52. private int reihe;
53. }
54. public abstract class Figur {
55. private String farbe;
56. private String name;
57. private Koordinate pos;
58. }

# Zeichenzähler

Schreiben Sie ein Java-Programm, welches nach Eingabe einer Zeichenkette die Häufigkeiten der einzelnen Zeichen zählt. Anschließend sind die Zeichen mit ihren Häufigkeiten und das Zeichen bzw. die Zeichen mit der größten Auftrittshäufigkeit untereinander auszugeben.

Legen Sie dazu eine geeignete Datenstruktur zur Speicherung der Häufigkeit an (Liste).

Bitte eine Zeichenkette (max. 80 Zeichen) eingeben:

*123213123eee$%&%&$333333333333333333333333333*

Folgende Zeichen wurden gezaehlt:

$ = 2

% = 2

& = 2

1 = 3

2 = 3

3 = 30

e = 3

Maximalhaeufigkeit von 30 hat Buchstabe

3

Ausgabe ist VERBINDLICH!