



Übung 04 – Schleifen, Strings und Arrays

Aufgabe 1 – Zeichenketten verkehrt herum ausgeben


Schreiben Sie ein Programm, welches eine Zeichenkette einliest und diese verkehrt herum wieder ausgibt. Der Programmablauf soll wie folgt aussehen:

Bitte Zeichenkette eingeben:	
MAGNETRESONANZTOMOGRAPHIE	 Eingabe
Ihre Eingabe in umgekehrter Reihenfolge:	
EIHPARGOMOTZNANOSERTENGAM	 Ausgabe

Legen Sie ein neues Java-Projekt in IntelliJ und ein Paket `de.htw.stringtools` im `src`-Ordner an. In diesem Paket können Sie eine Klasse `ReverseApplication` anlegen und darin eine `main`-Methode (siehe Beispiel).

```
package de.htw.stringtools;
import java.util.Scanner;

public class ReverseApplication {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // please add your code
    }
}
```

 Hier erweitern

Die Zeichenkette, welcher der Nutzer / die Nutzerin eingibt, können Sie mit der `Scanner`-Klasse und der `next`-Methode einlesen. Für die Ausgabe der umgekehrten Zeichenketten sollen Sie eine Schleife verwenden, die jedes einzelne Zeichen des eingelesenen Strings in der gewünschten Reihenfolge ausgibt.

Hinweis: Auf ein Zeichen in einer Zeichenkette (`String`) können Sie mit der `charAt`-Methode zugreifen (siehe Beispiel). Außerdem ist die `length`-Methode für die Bearbeitung der Aufgaben hilfreich.

Beispiel-Code:

```
String s = "hello";
System.out.print( s.charAt(1) ); // Gibt 'e' aus
System.out.print( s.charAt(4) ); // Gibt 'o' aus
System.out.print( s.length() ); // Gibt 5 aus
```

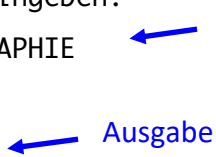
Aufgabe 2 – Häufigkeit von Vokalen zählen

Schreiben Sie ein Programm, welches eine Zeichenkette einliest und bestimmt, wie häufig ein Vokal in dieser Zeichenkette vorkommt. Als Vokale sollen hier die Buchstaben A,E,I,O,U behandelt werden, die Umlaute sollen in dieser Aufgabe zur Vereinfachung nicht betrachtet werden.

Legen Sie in dem Paket `de.htw.stringtools` eine Klasse `VowelApplication` mit einer `main`-Methode an. Prüfen Sie jeden Buchstaben in der Zeichenkette, ob einer der Vokale (A,E,I,O,U) vorliegt. Nutzen Sie die Methode `toUpperCase`-Methode bzw. die `toLowerCase`-Methode der Klasse `String`, um alle Buchstaben vor der Prüfung umzuwandeln, so dass für die Prüfung die Groß- und Kleinschreibung keine Rolle mehr spielt.

Der Programmablauf soll wie folgt aussehen:

```
Bitte Zeichenkette eingeben:  
MAGNETRESONANZTOMOGRAPHIE  
Anzahl der Vokale:  
10
```



Oder auch:

```
Bitte Zeichenkette eingeben:  
Magnetresonanztomographie  
Anzahl der Vokale:  
10
```

Erweiterung: Strukturieren Sie Ihr Programm um und überführen Sie die Überprüfungslogik in eine `countVowels`-Methode, die ebenfalls innerhalb der Klasse `VowelApplication` (aber außerhalb der `main`-Methode) angelegt wird. Die Methode soll folgenden Aufbau haben:

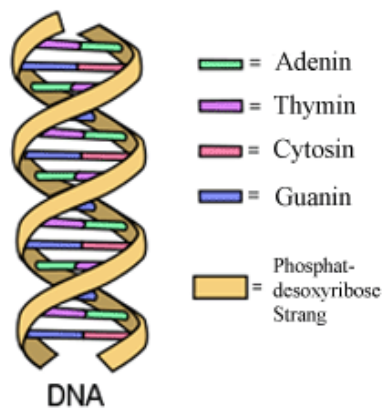
```
public static int countVowels(String s) {  
    // please add your code  
}
```

Die Methode soll die Anzahl der Vokale in dem `String s` zurückgeben. Rufen Sie nachdem Sie die Nutzereingabe aus der Konsole eingelesen haben von nun an die `countVowels`-Methode auf, um den `String` zu analysieren und verwenden das Ergebnis des Methodenaufrufs um es auf der Konsole auszugeben:

```
Anzahl der Vokale:  
10
```

Hinweis zu Randfällen: Falls der Parameter `s` in der `countVowels`-Methode den Wert `null` hat, soll das Ergebnis `0` sein.

Aufgabe 3 - DNA-Sequenz analysieren



Quelle: <https://www.biologie-schule.de/desoxyribonukleinsaeure-dna.php>

Die Erbinformation ist in der DNA gespeichert. DNA besteht aus 4 verschiedenen Nukleotiden, die mit A,C,G und T abgekürzt werden.

Schreiben Sie ein Programm, welches eine DNA-Sequenz von der Konsole einliest. In dem eingegebenen String soll die Häufigkeit der Buchstaben (Nukleotide) A, C, G und T gezählt werden und aus den Häufigkeiten deren prozentualer Anteil ausgerechnet werden.

Der Programmablauf soll wie folgt aussehen:

Bitte DNA-Sequenz eingeben:

```
AGCAGTTGAATTCTAAAGCAGTGCATCGATTGCGCATGTGCACAAACGTCGTCAGCTAGATGCCGGTACTGTACT
TGTACGTATCTAG
```

← Eingabe

Ergebnis:

A - Anzahl: 23, Prozent: 26,1

← Ausgabe

C - Anzahl: 19, Prozent: 21,6

G - Anzahl: 22, Prozent: 25,0

T - Anzahl: 24, Prozent: 27,3

Erweiterung: Falls sich ein anderes Zeichen als A;C,G,T in der Zeichenkette befindet, soll in dieses in als unbekanntes Zeichen gezählt werden. Der Programmablauf sieht nach der Erweiterung wie folgt aus:

Bitte DNA-Sequenz in die Konsole kopieren:

```
AGCAGTTGAATTCTAAAGCAGTGCATCGATTGCGCATGTGCACAAACGTCGTCAGCTAGATGCCGGTACTGTACT
TGTACGTATCTAGNN
```

Ergebnis:

A - Anzahl: 23, Prozent: 26,1

C - Anzahl: 19, Prozent: 21,6

G - Anzahl: 22, Prozent: 25,0

T - Anzahl: 24, Prozent: 27,3

Unbekannt - Anzahl: 2


Aufgabe 4 – Mittelwert berechnen

Erzeugen ein Paket `de.htw.average` und darin die Klasse `AverageApplication`, welche den Durchschnitt von Zahlen in einem Array berechnet. In dieser Aufgabe soll ein Array aus `double`-Werten als Argument an eine Methode `computeAverage` übergeben werden. Die Methode ist im Beispiel-Code schon angelegt und wird in der `main`-Methode bereits 3 Mal für unterschiedliche Arrays aufgerufen.

Bitte implementieren Sie Durchschnittsberechnung innerhalb der `computeAverage`-Methode und geben Sie das berechnete Ergebnis mit `return` zurück, so dass der Code funktionstüchtig wird und die untenstehenden Ausgaben erzeugt werden.

```
package de.htw.average;

public class AverageApplication {

    public static double computeAverage(double[] values) {
        // please add your code  Hier erweitern
    }

    public static void main(String[] args) {

        double[] values1 = {9.2, 10.9, 1.9, -1.1, 6.3, 4.8, 9.0, 10.1, 3.7, 5.5, 18.1, 12.5, 7.3, 15.0, 8.5};
        double[] values2 = {5.2, 19.9, 5.9, 20.1, -8.2, 3.1, 5.9, 12.5, 1.2, 17.8};
        double[] values3 = {-10.1, 21.3, 5.1, -3.1, 22.1, 10.1, 9.5, 0.5, 8.2, 6.3};

        double result1 = computeAverage(values1);
        System.out.printf("Arithmetisches Mittel: %.2f\n" , result1);
        double result2 = computeAverage(values2);
        System.out.printf("Arithmetisches Mittel: %.2f\n" , result2);
        double result3 = computeAverage(values3);
        System.out.printf("Arithmetisches Mittel: %.2f\n" , result3);
    }
}
```

Gewünschte Ausgabe:

```
Arithmetisches Mittel: 8,11
Arithmetisches Mittel: 8,34
Arithmetisches Mittel: 6,99
```