

Übungen zu Java

In den Aufgaben bezeichnen wir mit `Klasse#methode` Methoden, die auf Objekten der jeweiligen Klasse aufgerufen werden müssen, um die Aufgabe zu lösen.

Beispiel: `Scanner#nextLine` kann so verwendet werden:

```
Scanner scanner = new Scanner(System.in);  
String input = scanner.nextLine();
```

Intervalle:

Für Integer a und b definieren wir $[a, b]$ als das Intervall aller Integer von a inklusiv bis b inklusiv bzw. $[a, b)$ als das Intervall aller Integer von a inklusiv bis b exklusiv. Gleiches gilt für Floats.

Beispiele:

$[4, 7] = \{4, 5, 6, 7\}$.

$[3, 6) = \{3, 4, 5\}$.

$[0.0, 1.0] = \{\infty\text{-viele (bzw. sehr viele) Zahlen von 0 bis 1 inklusiv, z.B. 0.25}\}$



Echo-Höhle

Schreibe ein Programm, das unendlich oft Eingaben einliest und diese jeweils direkt ausgibt.

- Benötigte Syntax:
 - `Scanner#nextLine`, `System.out.println`
 - `while`-Schleife



Verzögertes Echo

Schreibe ein Programm, das unendlich oft Eingaben einliest und nach jeder Eingabe die vorherige Eingabe ausgibt.

(Beim ersten Mal soll ein leerer String mit println ausgegeben werden)

- Benötigte Syntax:
 - Scanner#nextLine, System.out.println
 - String-Variable
 - while-Schleife

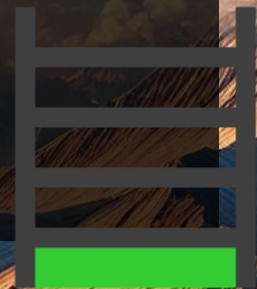


Hello, Worlds!

Schreibe ein kurzes Programm, das 100 Mal "Hello, World!" ausgibt.

- Benötigte Syntax:
 - `System.out.println`
 - while-Schleife oder for-Schleife

```
Hello, World!  
Hello, World!  
Hello, World!  
Hello, World!  
Hello, World!  
Hello, World!  
Hello, World!  
Hello, World!
```

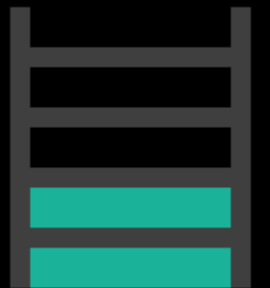
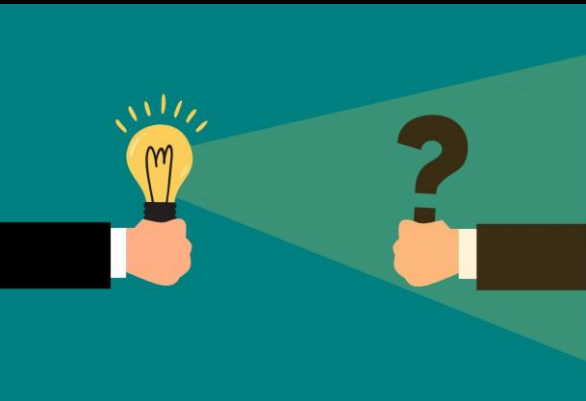


Frage-Antwort-Roboter



Schreibe ein Programm, das eine Frage aus der Konsole einliest und durch vorgefertigte Antworten darauf reagiert. Die Fragen und Antworten darfst du dir selbst ausdenken.

- Benötigte Syntax:
 - `Scanner#nextLine`, `System.out.println`
 - if-else oder Ternary



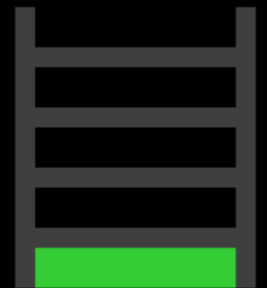
Fläche eines Kreises



Schreibe ein Programm, das einen Radius einliest und die Fläche eines Kreises mit diesem Radius ausgibt.

Formel: Fläche = π * radius * radius

- Benötigte Syntax:
 - Scanner#nextDouble, System.out.println
 - double-Variable
 - Math.PI



Male einen Kreis

Schreibe ein Programm, das eine Kreisfläche als ASCII-Art ausgibt.
Der Radius soll 8 sein.

- Benötigte Syntax:
 - while-Schleife (oder for-Schleife)
 - System.out.print und println
 - if-else oder Ternary

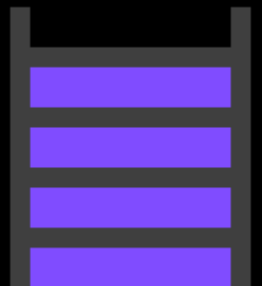
Iteriere über alle x-Koordinaten im Intervall [-10, 10]
und alle y-Koordinaten im Intervall [-10, 10].

Verwende den Satz des Pythagoras, um zu bestimmen,
ob an einer Stelle ein x (" x ") oder space (" ") mit print
ausgegeben werden soll.

Formel: $x * x + y * y < 64 // 8^2 = 64$

Verwende nach jeder Zeile einmal println, um in die
nächste Zeile zu gelangen.

```
      x x x x x x x
    x x x x x x x x x x x
  x x x x x x x x x x x x
x x x x x x x x x x x x x
x x x x x x x x x x x x x
x x x x x x x x x x x x x
x x x x x x x x x x x x x
x x x x x x x x x x x x x
x x x x x x x x x x x x x
  x x x x x x x x x x x
    x x x x x x x x x x x
      x x x x x x x
```



Gerade oder ungerade?

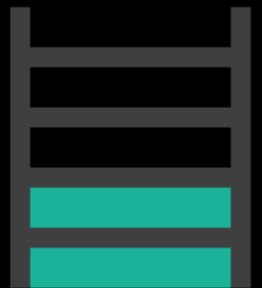


Schreibe eine Methode mit einem Parameter `int n`, die als `boolean` zurückgibt, ob `n` gerade ist.

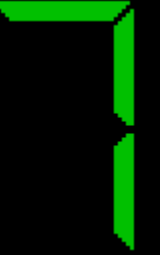
- Benötigte Syntax:
 - Aufbau einer Methode
 - Modulo-Operator

Der Modulo-Operator `%` bestimmt den Divisionsrest zweier Zahlen `a` und `b`, also das, was beim Teilen von `a` durch `b` übrig bleibt. Beispiele:

- `8 % 2 == 0` (da `8 / 2 = 4` Rest 0)
- `8 % 3 == 2` (da `8 / 3 = 2` Rest 2)
- `9 % 2 == 1` (da `9 / 2 = 4` Rest 1)
- `9 % 3 == 0` (da `9 / 3 = 3` Rest 0)



Schaltjahr?

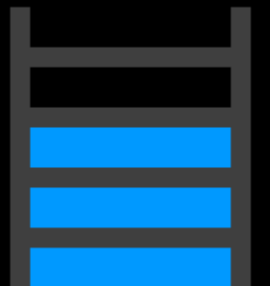


Schreibe eine Methode mit einem Parameter `int jahr`, die als `boolean` zurückgibt, ob das Jahr ein Schaltjahr ist.

- Benötigte Syntax:
 - Aufbau einer Methode
 - if-else oder Ternary
 - Modulo-Operator, und (`&&`) und oder (`||`) Operator

Jahre sind nicht durch Tage teilbar (das Ergebnis ist nicht genau 365). Die Schaltjahr-Regel versucht, diesen Effekt auszugleichen: Ein Jahr heißt Schaltjahr, wenn seine Jahreszahl

- durch 400 teilbar ist oder
- durch 4, aber nicht durch 100 teilbar ist.



Taschenrechner

Schreibe eine Methode "float rechne(float a, char operator, float b)", die das Ergebnis von $a <\text{operator}> b$ berechnet. Die Methode soll mindestens für +, - und * korrekt funktionieren.

Test-Beispiele:

```
System.out.println(rechne(2, '+', 3)); // Soll 5.0 ausgeben
```

```
System.out.println(rechne(-3, '*', 2)); // Soll -6.0 ausgeben
```

- Benötigte Syntax:
 - Aufbau einer Methode
 - if-else oder Ternary, um zwischen Operatoren zu unterscheiden

Vergleichs-Methode



Schreibe eine Methode "String vergleiche(float a, float b)", die "kleiner" (für $a < b$), "gleich" (für $a == b$) bzw. "größer" (für $a > b$) zurückgibt.

- Test-Beispiele:

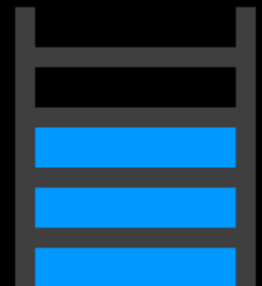
```
System.out.println(vergleiche(-3f, -3f)); // Soll gleich ausgeben
```

```
System.out.println(vergleiche(2.7182818f, 3f)); // Soll kleiner ausgeben
```

```
System.out.println(vergleiche(3.1415926f, 3f)); // Soll größer ausgeben
```

- Benötigte Syntax:

- Aufbau einer Methode
- if-else oder Ternary

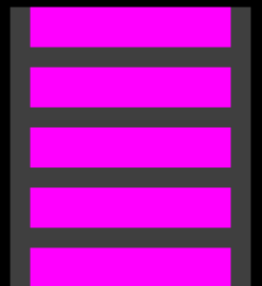


Zahl erraten

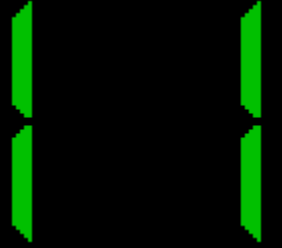


Schreibe ein Programm, das einen zufälligen Integer aus dem Intervall $[0, 100)$ in einer Variable speichert und so lange Integer einliest und ausgibt, ob diese kleiner, gleich oder größer sind, bis die richtige Zahl erraten wurde.

- Benötigte Syntax:
 - `Scanner#nextInt`, `System.out.println`
 - `int`-Variablen, `Random#nextInt`
 - `(do-)while`-Schleife und [Vergleichs-Methode](#)



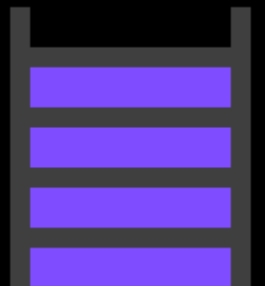
Fibonacci-Folge



In der Fibonacci-Folge ist jede Zahl die Summe der beiden vorherigen Zahlen. Sie beginnt mit 0 und 1. Berechne alle Fibonacci-Zahlen bis 1000 und gib sie aus.

- Benötigte Syntax:
 - `System.out.println`
 - `int`-Variablen
 - `while`-Schleife

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987



Palindrom

12

Schreibe eine Methode, die überprüft, ob ein String ein Palindrom ist.

- Test-Beispiele:

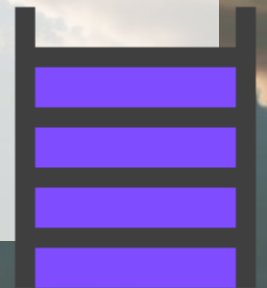
```
System.out.println(palindrom("Hallo")); // Soll false ausgeben
```

```
System.out.println(palindrom("lagerregal")); // Soll true ausgeben
```

```
System.out.println(palindrom("reifer")); // Soll false ausgeben
```

- Benötigte Syntax:

- Aufbau einer Methode
- String#length, String#charAt
- for-Schleife

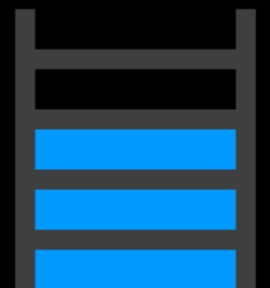


Fakultät

13

Schreibe eine Methode, die für einen Parameter `int n` den Wert $n!$, also $1 * 2 * 3 * 4 * \dots * n$ berechnet.

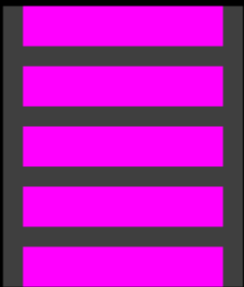
- Benötigte Syntax:
 - Aufbau einer Methode
 - `int`-Variable
 - `for`-Schleife



Primzahlen

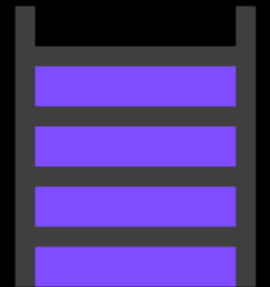
14

- Schreibe ein Programm, das alle Primzahlen unter 100 berechnet und ausgibt. Teste dafür für jede Zahl, ob sie durch eine kleinere Zahl teilbar ist.
- Benötigte Syntax:
 - `System.out.println`
 - for-Schleife, ggf. `break`



2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

← Bonusaufgabe: Verwende einen `int[]` für die Berechnung.

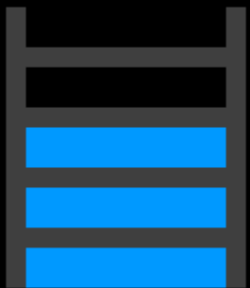


Zufallszahl in einem Intervall

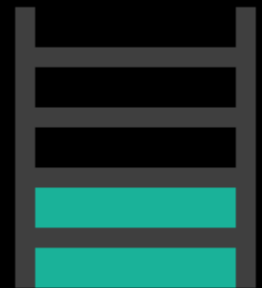
15

Gib einen zufälligen float-Wert zwischen 10 (inklusive) und 20 (exklusiv) aus.

- Benötigte Syntax:
 - `System.out.println`
 - `Random#nextFloat`



← Bonusaufgabe: Lies das Intervall $[x, y)$ durch 2x `Scanner#nextFloat` ein.



Lustige Sätze

15

Erstelle String-Arrays mit

1. Artikeln + Subjekten (Nominativ Singular)
2. Prädikaten (3. Person Singular)
3. Adverbien
4. Adjektiven (Akkusativ)
5. Artikeln (Akkusativ) + _ + Objekte (Akkusativ Plural)
6. Interpunktion (., !, ?)

- Benötigte Syntax:

- Random#nextInt, String[] length zum Auswählen zufälliger Wörter
- String#replace zum Ersetzen des Unterstrichs

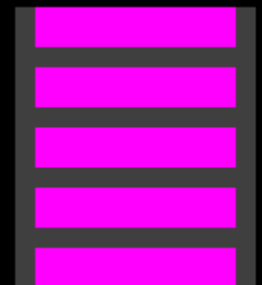
Generiere einen zufälligen Satz aus diesen Wörtern.

Muster: <1> <2> <3> (<5>, ersetze _ durch <4>) <6>.

<x> steht dabei jeweils für ein zufälliges Element aus dem x-ten Array.

Beispiel: Der Rabe berechnet niemals laute Felsen.

Tipp: Schreibe eine statische Methode, um ein zufälligen String aus einem String[] auszuwählen.

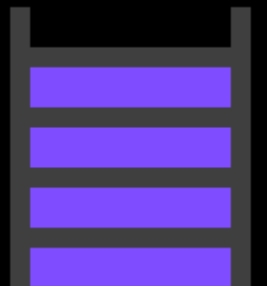


Glücksspiel



Schreibe ein Textspiel, bei dem man mit 100 Punkten beginnt und solange man mehr als 0 Punkte hat, einen beliebigen Betrag einsetzen kann, den man zu 50% gewinnt und sonst verliert. Verwende `continue`, wenn `!scanner.hasNextInt()` oder der eingelesene Integer nicht im Intervall `[1, punkte]` liegt.

- Benötigte Syntax:
 - `System.out.println`, `Scanner#hasNextInt`, `Scanner#nextInt`
 - `Random#nextInt` oder `Random#nextBoolean`
 - `while`-Schleife, `int`-Variablen, ggf. `continue`;



Summe



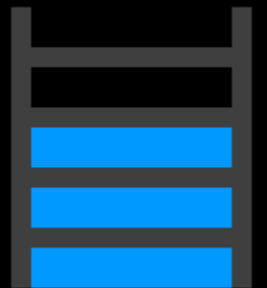
Schreibe eine Methode, die alle Elemente eines float[] aufsummiert.

- Test-Beispiel:

```
float[] array = {1.5f, 2.5f, 3.5f, 4.5f};  
System.out.println(summe(array)); // Soll 12.0 ausgeben.
```

- Benötigte Syntax:

- Aufbau einer Methode
- float-Variable
- foreach-Schleife

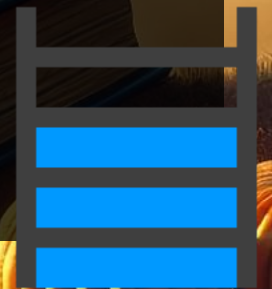


Größtes Element

19

Schreibe eine Methode, die das größte Element eines `float[]` bestimmt.
Tipp: Starte mit $-\infty$. (`Float.NEGATIVE_INFINITY`)

- Test-Beispiel:
`float[] werte = {-22.3f, 2.2f, 1.5f, -4.2f, -9.87f};`
`System.out.println(max(werte));`
- Benötigte Syntax:
 - Aufbau einer Methode
 - float-Variable
 - foreach-Schleife, if



Array sortieren



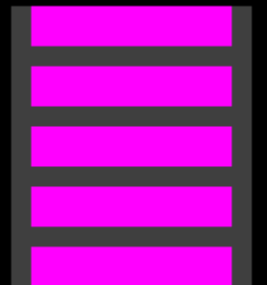
Schreibe eine Methode, die einen `int[]` sortiert. Überlege dir zuerst anhand eines Beispiels auf einem Blatt Papier, wie sie dies tun kann.

- Test-Beispiel:

```
int[] array = {1, 3, 4, 2};  
sort(array);  
System.out.println(Arrays.toString(array)); // Soll 1 2 3 4 ausgeben.
```

- Benötigte Syntax:

- Aufbau einer Methode
- `int[]` length
- for-Schleife



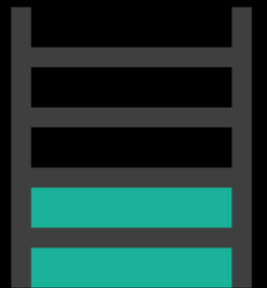
Timer



Schreibe ein Programm, das zweimal Input einliest und die Zeit dazwischen misst und ausgibt.

Berechne die Zeitdifferenz durch den Minus-Operator (-).

- Benötigte Syntax:
 - `Scanner#nextLine`, `System.out.println`
 - `System.currentTimeMillis`



Finde das Wort



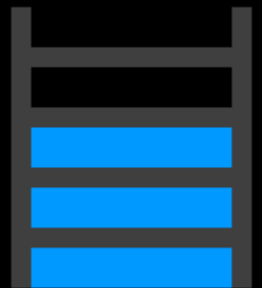
Schreibe eine Methode, die einen vorgegebenen String in einem String[] sucht und den Index (bzw. -1, wenn er nicht im Array enthalten ist) zurückgibt.

- Test-Beispiel:

```
String[] strings = {"Baum", "Haus", "Dorf"};  
System.out.println(indexVon(strings, "Dorf")); // Soll 2 ausgeben.
```

- Benötigte Syntax:

- Aufbau einer Methode
- Arrays, for-Schleife
- String#equals

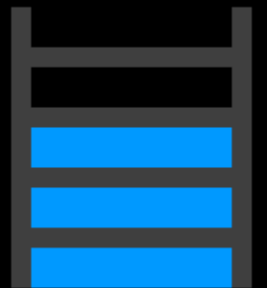


Keine Vokale erlaubt.



Lies einen String durch den Scanner ein. Gib alle Zeichen aus, bis du auf einen Vokal triffst.

- Benötigte Syntax:
 - `System.out.print`
 - `String#length`, `String#charAt`
 - `for`-Schleife, `break`;



Schwache Zahlen

Gib die Integer im Intervall $[1, 1000]$ aus, überspringe dabei aber alle Zahlen, für die es eine kleinere natürliche Zahl gibt, die mehr Teiler hat.

Beispiel: 9 ist eine schwache Zahl, da 9 nur 3 Teiler (1, 3, 9), die kleinere Zahl 6 aber 4 Teiler (1, 2, 3, 6) hat. Also muss 9 übersprungen werden.

Tipp: Merke dir immer nur die bisher größte Anzahl an Teilern. Zahlen mit gleichen Teileranzahlen sollen nicht übersprungen werden.

- Benötigte Syntax:

- `System.out.println`
- `int`-Variablen
- `for`-Schleife, `if`, `Modulo`, ggf. `continue`;

1, 2, 3, 4, 6, 8, 10, 12, 18, 20, 24, 30, 36, 48, 60, 72, 84, 90, 96,
108, 120, 168, 180, 240, 336, 360, 420, 480, 504, 540, 600, 630,
660, 672, 720, 840

