Programmieren in Java

http://proglang.informatik.uni-freiburg.de/teaching/java/2017/

int-to-string

Integer in String umwandeln

Woche 02 Aufgabe 1/7

Herausgabe: 2017-04-25
Abgabe: 2017-05-12

Achtung: beachten Sie unbedingt die allgemeinen Hinweise zur Abgabe auf der Homepage.

Project	`int-to-string`
Package	`inttostring`
Klassen	

Main
`public static void main(String[])`

Die Aufgabe besteht darin eine über `stdin` eingegebene Zahl vom Typ `int` einzulesen und diese in einen String umzuwandeln. Falls eine korrekte Integer Zahl n eingegeben wurde soll

`"Found int: "` n

auf `stdout` auszugeben werden, in jedem anderen Fall

`"No int"`

.

Beispieleingabe 01:

```
43
```

Erwartete Ausgabe 01:

```
Found int: 43
```

Beispieleingabe 02:

```
2.5
```

Erwartete Ausgabe 02:

```
No int
```

Hinweise

- Für diese Aufgabe werden „Conditionals" (`if`-Statements) benötigt. Wie die in Java aussehen steht z.B. hier: `http://docs.oracle.com/javase/tutorial/java/nutsandbolts/if.html`.
- Die API von `java.util.Scanner` ist hier hilfreich.

Programmieren in Java

`http://proglang.informatik.uni-freiburg.de/teaching/java/2017/`

integer-primitives

Speichern von ganzen Zahlen

Woche 02 Aufgabe 2/7

Herausgabe: 2017-04-25
Abgabe: 2017-05-12

Achtung: beachten Sie unbedingt die allgemeinen Hinweise zur Abgabe auf der Homepage.

Project	`integer-primitives`
Package	`integerprimitives`
Klassen	

Main
`public static void main(String[])`

Die Aufgabe besteht darin über `stdin` eingegebene Zahl einzulesen und auszugeben in welchem primitiven Ganzzahl-Datentypen diese Zahl gespeichert werden kann. Das Programm soll auch mitteilen, falls die die Zahl in keinem der Ganzzahl-Typen dargestellt werden kann.

Die primitiven Datentypen für ganze Zahlen sind `byte, short, int, long`. Die Beispiele unten zeigen das gewünschte Ausgabeformat.

Sie können davon ausgehen, dass genau eine Zahl eingegeben wird.

Beispieleingabe 01:

```
-150
```

Erwartete Ausgabe 01:

```
-150 fits in:
* short
* int
* long
```

Beispieleingabe 02:

```
235235235235235325325
```

Erwartete Ausgabe 02:

```
"235235235235235325325" doesn't fit anywhere.
```

Hinweise: Die APIs der folgenden Klassen könnten nützlich sein:

- `java.lang.Integer`, `java.lang.Byte`, `java.lang.Short`, `java.lang.Long`
- `java.lang.Math`

Programmieren in Java

`http://proglang.informatik.uni-freiburg.de/teaching/java/2017/`

weird-numbers

Erkennen von merkwürdigen Zahlen

Woche 02 Aufgabe 3/7

Herausgabe: 2017-04-25
Abgabe: 2017-05-12

Achtung: beachten Sie unbedingt die allgemeinen Hinweise zur Abgabe auf der Homepage.

Project `weird-numbers`
Package `weirdnumbers`
Klassen

Main
`public static void main(String[])`

Die Aufgabe besteht darin über `stdin` eingegebene Zahl vom Typ `int` einzulesen und auszugeben ob diese Zahl merkwürdig (`Weird`) ist oder nicht (`Not Weird`). Ob eine Zahl merkwürdig ist lässt sich wie folgt bestimmen:

- Die Zahl ist ungerade → merkwürdig
- Die Zahl ist gerade und liegt zwischen einschließlich 2 und 5 → nicht merkwürdig
- Die Zahl ist gerade und liegt zwischen einschließlich 6 und 20 → merkwürdig
- Die Zahl ist gerade und größer als 20 → nicht merkwürdig
- In allen anderen Fällen ist eine Zahl genau dann merkwürdig, wenn Sie negativ ist.

Sie können davon ausgehen, dass nur Zahlen die in den Typ `int` passen eingegeben werden.

Beispieleingabe 01:

```
20
```

Erwartete Ausgabe 01:

```
Weird
```

Beispieleingabe 02:

```
22
```

Erwartete Ausgabe 02:

```
Not Weird
```

Programmieren in Java

`http://proglang.informatik.uni-freiburg.de/teaching/java/2017/`

time-to-words

Die Zeit in Worten

Woche 02 Aufgabe 4/7

Herausgabe: 2017-04-25
Abgabe: 2017-05-12

Achtung: beachten Sie unbedingt die allgemeinen Hinweise zur Abgabe auf der Homepage.

Project `time-to-words`
Package `timetowords`
Klassen

`Main`
`public static void main(String[])`

Die Aufgabe besteht darin über `stdin` ein als Stunden und Minuten eingegebene Uhrzeit einzulesen und nach folgendem Muster in Worten auszugeben:

- 5:00 → five o'clock
- 5:01 → one minute past five
- 5:10 → ten minutes past five
- 5:30 → half past five
- 5:40 → twenty minutes to six
- 5:45 → quarter to six
- 5:47 → thirteen minutes to six
- 5:28 → twenty eight minutes past five

Die gültige Uhrzeit wird über zwei `int`-Zahlen eingegeben. Die erste Zahle H gibt die Stunden an, die zweite Zahl, M, die Minuten. Es gilt ferner: $1 \leq H \leq 12$ und $0 \leq M < 60$.

Falls Ihr Programm eine ungültige Eingabe bekommt, soll es die Meldung

`Wrong Input`

ausgeben.

Beispieleingabe 01:

```
5 51
```

Erwartete Ausgabe 01:

```
nine minutes to six
```

Beispieleingabe 02:

```
0 45
```

Erwartete Ausgabe 02:

```
Wrong Input
```

Hinweise: Bei dieser Aufgaben können Funktionen (bzw. `static` Methoden) nützlich sein. Auf der folgenden Internetseite finden Sie einige Erklärungen und Beispiele zu Funktionen in Java: `http://introcs.cs.princeton.edu/java/21function/`.

Programmieren in Java

http://proglang.informatik.uni-freiburg.de/teaching/java/2017/

simple-functions

Einfache Funktionen

Woche 02 Aufgabe 5/7

Herausgabe: 2017-04-25
Abgabe: 2017-05-12

Achtung: beachten Sie unbedingt die allgemeinen Hinweise zur Abgabe auf der Homepage.

Project `simple-functions`
Package `simplefunctions`
Klassen

Main

`public static double distanceOrigin(double x, double y)`
`public static double cubeVolume(double x)`
`public static double cubeSurface(double x)`
`public static String stringInsert(String str, int pos)`
`public static double capacitance(double epsilonR, double A, double d)`

Folgende Funktionen sollen implementiert werden:

- `public static double distanceOrigin(double x, double y)`
  Berechnet den Abstand von Punkt (x,y) zum Ursprung.
- `public static double cubeVolume(double x)`
  Berechnet das Volumen eines Würfels mit Seitenlänge x.
- `public static double cubeSurface(double x)`
  Berechnet die Oberfläche eines Würfels mit Seitenlänge x.
- `public static String stringInsert(String str, int pos)`
  Gibt einen neuen String aus, an dem in gegebener Position ein `"_"` eingefügt wurde.
- `public static double capacitance(double epsilonR, double A, double d)`
  Berechnet die Kapazität eines Kondensators aus der Dielektrizitätskonstante, der Fläche der Kondensatorplatten und der Distanz der Kondensatorplatten.

Programmieren in Java

`http://proglang.informatik.uni-freiburg.de/teaching/java/2017/`

convert-units

Einheiten konvertieren

Woche 02 Aufgabe 6/7

Herausgabe: 2017-04-25
Abgabe: 2017-05-12

Achtung: beachten Sie unbedingt die allgemeinen Hinweise zur Abgabe auf der Homepage.

Project `convert-units`
Package `convertunits`
Klassen

Conversions
`public static double km2miles(double x)`
`public static double miles2km(double x)`
`public static double celsius2fahrenheit(double x)`
`public static double fahrenheit2celsius(double x)`

Die Aufgabe besteht darin, Funktionen zur Einheitenumrechnung zu schreiben. Es soll möglich sein eine gegebene Distanz zwischen Kilometern und Meilen zu konvertieren. Außerdem soll eine Temperatur zwischen Celsius und Fahrenheit konvertiert werden können. Benennen sie die Funktionen wie folgt:

- `public static double km2miles(double x)`
- `public static double miles2km(double x)`
- `public static double celsius2fahrenheit(double x)`
- `public static double fahrenheit2celsius(double x)`

Die benötigten Umrechnungsformeln müssen Sie selbst recherchieren. Es bleibt auch Ihnen überlassen, entsprechende Testfälle zu erstellen (d.h. es gibt keinen `testcases` Ordner oder Beispieleingaben).

Hinweise:

- Um die Testfälle auf dem Build-Server zu bestehen, müssen die Funktionen unbedingt wie oben spezifiziert benannt werden.
- Um Ihre Methoden zu testen, rufen Sie sie mit Beispielwerten auf und prüfen Sie das Ergebnis (z.B. indem Sie es in einer main-Methode ausgeben).
- Auf der folgenden Internetseite finden Sie einige Erklärungen und Beispiele zu Funktionen in Java: `http://introcs.cs.princeton.edu/java/21function/`.

Programmieren in Java

http://proglang.informatik.uni-freiburg.de/teaching/java/2017/

conditional-functions

Bedingungen

Woche 02 Aufgabe 7/7

Herausgabe: 2017-04-25
Abgabe: 2017-05-12

Achtung: beachten Sie unbedingt die allgemeinen Hinweise zur Abgabe auf der Homepage.

Project `conditional-functions`
Package `conditionalfunctions`
Klassen

Functions
`public static String turbineControl(double f)`

Die folgende Funktion soll implementiert werden:

```
public static String turbineControl(double f)
```

Abhänging von der gegebenen Frequenz `f` werden Kommandos für eine Turbine als String ausgegeben.

- `"DISCONNECT"` falls $f \leq 49$ oder $f \geq 51$,
- `"MORE_WATER"` falls $f \lesssim 50$,
- `"LESS_WATER"` falls $f \gtrsim 50$ und
- `"STEADY"` falls $f \approx 50$.

Zwei Zahlen sind annähernd gleich ($\approx$), falls ihr Unterschied weniger als 0.001 beträgt. Entsprechend sind auch $\lesssim$ („signifikant kleiner") und $\gtrsim$ („signifikant größer") definiert.

Beispielaufruf 01: `Functions.turbineControl(49.9)` ergibt `MORE_WATER` (als `String`)