Programmieren und Software-Engineering II
Übung 3

Name: _____ **Klasse:** _____ **Datum:** _____

Lernziele:

- Verwendung von printf() zur Erstellung formatierter Ausgaben

Aufgabe 1: Fortschrittsanzeige (Funktion `printProgressBar()` / beiliegendes Projekt)
Steht in einem Programm eine längere Operation an, so ist es oft vorteilhaft dem Benutzer eine Fortschrittsanzeige zu präsentieren. In der im beiliegenden Projekt Ex01_PrintfForFunAndProfit bereits enthaltenen Funktion `printProgressBar(double stepSize)` ist hierzu Code zu ergänzen, der den Fortschritt einer Operation von 0.00% bis 100.00% in Schritten zu je 3.70% zur Anzeige bringt:

```
Operation progress:    0.00%
    ⋮         ⋮          ⋮
Operation progress:  100.00%
```

Um dem Benutzer die Möglichkeit zu geben, die einzelnen Fortschritte entsprechend zu erfassen, ist zwischen diesen Ausgaben mittels der ebenfalls bereits enthaltenen Funktion sleep(); dabei jeweils eine Pause von `sleep(100);` Millisekunden einzulegen.

Achtung: 100 ist durch 3.7 nicht ohne Rest teilbar. Berücksichtigt man das nicht, lautet die letzte Ausgabe vermutlich auf "`Operation progress:  103.60%`". Versuche in deiner Implementierung sicherzustellen, dass der prozentuelle Fortschritt den Wert 100.00 nicht überschreitet!

Hinweis: damit die einzelnen Ausgaben nicht untereinander zu stehen kommen, sondern jede weitere "`Operation progress:`"-Ausgabe die (jeweils) vorige überschreibt, kann die Escape-Sequenz `\r` ("carriage return", der Wagenrücklauf) verwendet werden – vergleiche:

System.out.print("Operation progress: 0%%"); System.out.print("Operation progress: 0%%");

versus

System.out.print("\rOperation progress: 0%%"); System.out.print("\rOperation progress: 0%%");

Aufgabe 2: Bildschirmlineal (Funktion `printRuler()` / beiliegendes Projekt)
Um es Benutzern einer zeilenorientierten Konsolenanwendung zu ermöglichen, in etwa einschätzen zu können, wo innerhalb einer Zeile man sich zu jedem Zeitpunkt befindet, soll die beiliegende Funktion `printRuler(int consoleWidth, int tickSpacing)` um passenden Code ergänzt werden, sodass über die Bereite einer einzelnen Textzeile (mit `consoleWidth` Spalten)

- alle `tickSpacing` Spalten der – jeweils aktuelle – Spaltenindex gefolgt von einem Doppelpunkt `:` sowie
- alle `2*tickSpacing` Spalten den Spaltenindex gefolgt von einem „unterbrochenem Strich" `|` (=dem Zeichen, das in Java als Teil des Oder-Operators dient) ausgegeben wird.

Bei korrekter Umsetzung der Aufgabe sollte sich nach Ausführung der Funktion folgende Bildschirmausgabe ergeben:

Programmieren und Software-Engineering II
Übung 3

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
--------------------------------------------------------------------------------
   5:  10|  15:  20|  25:  30|  35:  40|  45:  50|  55:  60|  65:  70|  75:  80|
```

Zusatz: Die Funktion wird in main() standardmäßig mit CONSOLE_WIDTH = 80; und TICK_SPACING = 5; aufgerufen. Versuche, deine Funktion nach Möglichkeit schließlich so zu generalisieren, dass diese für beliebige Bildschirmbreiten und Markierungsabstände (größer gleich 4) korrekt funktioniert!

Aufgabe 3: Kassabon (Funktion `printReceipt()` / beiliegendes Projekt)

Mit einer Supermarktkette als Auftraggeber stehen wir vor der Aufgabe der Umsetzung einer Java-Funktion, mittels der Kassabons ausgedruckt werden können. An die im beiliegenden Projekt enthaltene Funktion `printReceipt(String[] product, int[] quantity, double[] itemPrize)` werden entsprechend einer mit dem Kunden bereits ausgearbeiteten Spezifikation insgesamt 3 Arrays übergeben, in denen sich folgende Informationen finden:

- ein String-Array mit den Bezeichnern der gekauften Produkte
- ein int-Array, in dem sich deren – jeweilige – Anzahl findet
- ein double-Array mit den zugehörigen Einzelstückpreisen

Diese Index-für-Index durchlaufend soll auf Basis des Inhalts dieser Arrays mittels printf() eine formatierte Ausgabe der einzelnen Rechnungszeilen versucht werden. Das folgende Beispiel zeigt dabei das gewünschte Ausgabeformat:

```
             SUPERMARKET
--------------------------------------
Datum: 09.10.2024

Clever Vanillejoghurt     15      0.45
Clever Bratwürstel dün   117      3.99
<——— 22 Zeichen ———>    (1)       (2)
--------------------------------------
SUMME                  EUR      473.58
======================================
<————————— 38 Zeichen —————————>
```

(1) `4-stellige Ausgabe der Anzahl` (rechtsbündig)
(2) `8-stellige Ausgabe, 2 Ziffern Dezimaltrennzeichen` (rechtsbündig)

Knobelaufgabe: In der ersten Version der Funktion kann die Überschrift (`"SUPERMARKET"`) zentriert werden, indem man dieser im Zuge des print()-Aufrufs eine passende Anzahl an Leerzeichen (händisch) voranstellt.

Wie könnte wohl aber ein printf() Aufruf aussehen, der die im Programm bereits deklarierte Konstante RECEIPT_WIDTH sowie ein Textliteral mit – zumindest – 38 Leerzeichen mittels des + Operators (des "string concatenation operators") so „zusammenbaut", dass die Ausgabe der in Form eines zusätzlichen Arguments übergebenen Zeichenkette `"SUPERMARKET"` durch die Funktion `System.out.printf()` – quasi – automatisch zentriert wird?