Aufgabe 7.1
Schreiben Sie eine Methode, die ein Array erzeugt, das bis zu einer einzugebenden Länge n die Folgeglieder $a_i = (-1)^i(2i+1)$ enthält.
Beispiel: Für den Eingabewert $n = 4$ soll das Array mit den Werten
$a_0 = 1, a_1 = -3, a_2 = 5, a_3 = -7$ belegt werden.

Überlegen Sie, wie Sie das alternierende Vorzeichen erzeugen können, ohne die Potenzfunktion `Math.pow` zu benutzen.

Aufgabe 7.2
Eine Menge von (n-vielen) Punkten $\{(x_i, y_i)|i = 1 \dots n\}$ in der euklidischen Ebene nennt man auch eine „Punktwolke“.

Nach der *Methode der kleinsten Quadrate (least squares method)* kann man durch eine Punktwolke eine Gerade $g(x) = ax + b$ bestimmen, die in gewissen Sinne den kleinsten Abstand zu allen Punkten hat.

Die Parameter a und b berechnen sich dabei nach den Formeln:

$$a = \frac{n\sum_{i=1}^{n}(x_i y_i) - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{n\sum_{i=1}^{n}(x_i^2) - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} x_i)}$$

$$b = \frac{\sum_{i=1}^{n}(y_i)\sum_{i=1}^{n}(x_i^2) - (\sum_{i=1}^{n} x_i)\sum_{i=1}^{n}(x_i y_i))}{n\sum_{i=1}^{n}(x_i^2) - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} x_i)}$$

Definieren Sie eine Klasse `Punkt` (wie die bereits aus Kap 3 bekannte Klasse `Punkt2D`, aber nun mit Koordinaten vom Typ `double`).

Schreiben Sie in einer Test-Klasse eine Methode `leastSquares()`, die zu einem **Array von Punkten** die beiden Parameter a und b der Geraden zurückliefert.
(Die Methode erzeugt also ein double-Array der Länge 2.)

Nutzen Sie diese Methode, um folgende Aufgabe zu lösen:

In den Jahren 2008 bis 2015 wurden in Leipzig folgende Durchschnittstemperaturen für den Monat Juli gemessen: (Quelle: www.langfristwetter.com, zuletzt abgerufen Mai 2024)

’08	’09	’10	’11	’12	’13	’14	’15
19.4	19.2	21.8	17.3	18.7	20.5	20.6	20.6

Bestimmen Sie nun mit der Methode der kleinsten Quadrate (leastSquares) die Parameter a und b einer Geraden $g(x) = ax + b$ durch diese Punktwolke.

Tipp:

- Betrachten Sie ein Paar (Jahreszahl, Temperatur) als einen Punkt. Sie sollten also auch die Jahreszahlen als `double` definieren.
- Erzeugen Sie durch entsprechende Hilfsmethoden aus einem Array von Punkten zunächst zwei Arrays (eines für die Jahreszahlen, eines für die Temperaturen). Im ersten Array sind die x-Koordinaten der Punkte abgelegt, im zweiten Array die y-Koordinaten.

Aufgabe 7.3
Ein Histogramm ist eine graphische Darstellung der Häufigkeit von Werten. Zum Beispiel wird die Häufigkeitsverteilung der Werte 0 bis 9 in dem Array
{6, 8, 1, 8, 5, 2, 2, 3, 7, 1, 5, 3, 1, 0, 1, 5, 6, 6, 9 } durch dieses Histogramm wiedergeben:

0: *
1: ****
2: **
3: **
4:
5: ***
6: ***
7: *
8: **
9: *

Implementieren Sie eine (statische) Methode, die das Histogramm eines übergebenen Arrays in der oben gezeigten Weise auf dem Bildschirm ausgibt.
Sie dürfen davon ausgehen, dass das Eingabe-Array nur Werte zwischen 0 (einschließlich) und 10 (ausschließlich) enthält.
Erzeugen Sie zunächst ein (Hilfs-)Array der Länge 10, das an jeder Position die Häufigkeit des betreffenden Index-Wertes erhält. Im Beispiel von oben müsste das Hilfsarray also den Inhalt
{1, 4, 2, 2, 0, 3, 3, 1, 2, 1} bekommen.

Achten Sie darauf, dass Sie *effizient* programmieren! Es gibt eine Lösung, bei der das Eingabe-Array nur **einmal** durchlaufen werden muss!

Aufgabe 7.4
Schreiben Sie eine (statische) Methode, die ein 2-dimensionales `double`-Array `A` (also eine Matrix), einen `double`-Wert $alpha$ und zwei int-Zahlen k und m als Eingabe erhält.
Die Methode soll einen Schritt des Gauss-Jordan-Verfahrens ausführen, indem sie das α-fache der k-ten Zeile zur m-ten Zeile der Matrix addiert.
(Beachten Sie, dass die *erste* Zeile der Matrix die Indexnummer 0 hat!)
Sie dürfen davon ausgehen, dass alle Zeilen der Matrix gleich lang sind und dass m und k zwischen 1 und der Zeilenzahl der Matrix liegen.

Beispiel:
Bei Aufruf von `add(m, 3, 2, -0.5)` wird die Matrix

$$m = \begin{pmatrix} 1.1 & 2.2 & -3.3 & 4.4 \\ 5.0 & -6.2 & 6.4 & -8.8 \\ 0.0 & 8.0 & 1.0 & 0.5 \end{pmatrix} \quad \text{zu} \quad m = \begin{pmatrix} 1.1 & 2.2 & -3.3 & 4.4 \\ 5.0 & -6.2 & 6.4 & -8.8 \\ -2.5 & 11.1 & -2.2 & 4.9 \end{pmatrix}$$

Aufgabe 7.5
Wir modellieren einen Versandhandel, bei dem Kunden mehrere Artikel in einen (virtuellen) „Warenkorb“ legen und anschließend kaufen können.

Es werden verschiedene `Artikel` angeboten; jeder Artikel hat eine Bezeichnung (String) und einen Preis (double) und (s.u.) eine Artikelnummer.

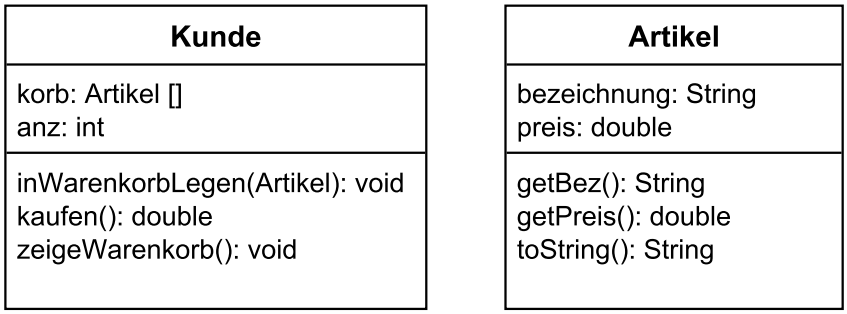
Ein `Kunde` hat einen Warenkorb (gerne auch einen Namen und (s.u.) eine Kundennummer).

Ein Warenkorb kann (zunächst) 5 Artikel aufnehmen; wenn ein Kunde mehr als 5 Artikel in seinen Warenkorb legen möchte, wird der Korb jeweils um 5 mögliche weitere Artikel vergrößert.

(In unserer Simulation kann man immer nur *ein* Exemplar eines Artikels in den Warenkorb legen, nicht mehrere Exemplare des gleichen Artikels auf einmal. Möchte ein Kunde zwei Exemplare eines Artikels kaufen, muss er den Artikel zweimal in seinen Warenkorb legen.)

Beim Kauf wird der Rechnungsbetrag ermittelt und der Warenkorb wieder auf seine ursprüngliche Größe reduziert und geleert.

Ausschnitte der UML-Diaramme sind unten angegeben.



Implementieren Sie die Klassen `Kunde` und `Artikel`.

- Das Attribut `anz` zählt mit, wieviele Artikel bereits im Warenkorb liegen.
- Das Attribut `korb` eines Kundenobjektes enthält zunächst Platz für 5 Artikelobjekte.
  Durch die Methode `inWarenkorbLegen()` wird der nächste freie Platz des Arrays mit dem übergebenen Artikel belegt. Falls das Array bereits komplett gefüllt ist, soll ein *neues* Array mit 5 weiteren Plätzen angelegt werden, die bisher eingetragenen Artikel sollen in das neue Array übertragen werden, der neu ausgewählte Artikel an den nun verfügbaren freien Platz eingetragen werden und das neue Array dem Attribut `korb` zugewiesen werden.
- Durch die Methode `kaufen` wird der Rechnungsbetrag ermittelt (Summe der Preise aller im Warenkorb enthaltenen Artikel) und zurückgegeben; außerdem wird der Warenkorb wieder geleert und auf seine ursprüngliche Größe (Platz für 5 Artikel) zurückgesetzt.
- Ein Löschen von Waren aus dem Warenkorb ist nicht möglich.
- Die Methode `zeigeWarenkorb` gibt die Artikel im Warenkorb auf dem Monitor aus.
- Fügen Sie (zusätzlich zu den in den UML-Diagrammen dargestellten Attributen und Methoden) ein:
  - erforderliche Klassen- und/oder Instanzattribute, um jedem Kunden eine eindeutige, automatisch generierte Kundennummer und jedem Artikel eine eindeutige, automatisch generierte Artikelnummer zu vergeben
  - entsprechende Konstruktoren

Schreiben Sie ein Testprogramm, in dem Artikel-Objekte erzeugt werden (zB Hose, T-Shirt, Sweater, . . . oder Tisch, Stuhl, Sofa, Bett, . . . oder was Ihnen sonst so einfällt).

Außerdem sollen mindestens zwei Kunden-Objekte erzeugt werden.
Der erste Kunde soll 7 Artikel und ein anderer Kunde 2 Artikel in seinen Warenkorb legen.
Anschließend legt der erste Kunde noch einmal einen Artikel in seinen Warenkorb.
Abschließend kaufen beide Kunden die Waren in ihrem Warenkorb. Lassen Sie die jeweiligen gekauften Artikel und den Gesamt-Rechnungsbetrag jedes Kunden ausgeben.