Hochschule Bonn-Rhein-Sieg
Fachbereich Informatik
M.Sc. Andre Kless
Dipl. Inf. Sigrid Weil

Übung 6

Einführung in die
Programmierung
SoSe 2023

Themen:
Stoff der Vorlesung bis einschließlich Kap06 (Iteration).
Außerdem wird die Klasse `Wuerfel` benötigt. (Siehe Code-Beispiel in LEA)

Aufgabe 6.1
Kater Tom und Maus Jerry spielen (auf einem Spielfeld aus hintereinanderliegenden Feldern) Nachlaufen. Wie weit sie jeweils laufen, wird ausgewürfelt:
Wird eine 1, eine 2 oder eine 4 gewürfelt, läuft Jerry entsprechend viele Felder vorwärts.
Bei einer 5 muss Jerry allerdings 5 Felder zurück.
Tom springt immer, wenn eine 3 oder eine 6 gewürfelt wird, um 3 Felder nach vorne.

Tom beginnt an Position 0, Jerry erhält einen Vorsprung. Am Ende des Spielfelds liegt das rettende Mauseloch.

Das Spiel endet, wenn entweder Jerry das Mauseloch erreicht hat, oder wenn Tom Jerry eingeholt hat.

Implementieren Sie eine Klasse `Jagd`, die ein Spielfeld mit einer anzugebenden Länge erzeugt (die Länge ist also die Position des Mauselochs). Außerdem wird dem Jagd-Objekt der Vorsprung mitgegeben, den Jerry erhält. Tom beginnt immer bei Position 0.

Definieren Sie eine (private) Methode, die die aktuelle Position von Tom und Jerry auf dem Monitor anzeigt.

Eine Methode `los()` simuliert den Spielverlauf:

- Zunächst wird die (Start-)Position der beiden Spieler angezeigt.
- Dann wird (mit einem Objekt der Klasse Wuerfel) gewürfelt und die entsprechende Figur entsprechend bewegt.
- Anschließend wird wieder die aktuelle Position der beiden angezeigt.
- Bei Spielende wird der Text „gerettet!“ bzw „gefangen!“ ausgegeben.

Ein (zufälliger) Spielverlauf mit den Werten `mauseloch = 20` und `vorsprung = 5` könnte so aussehen:

```
Tom: 0     Jerry: 5
Tom: 0     Jerry: 9
Tom: 3     Jerry: 9
Tom: 6     Jerry: 9
Tom: 6     Jerry: 10
Tom: 6     Jerry: 12
Tom: 9     Jerry: 12
Tom: 9     Jerry: 16
Tom: 9     Jerry: 17
Tom: 9     Jerry: 21
gerettet!
```

Hinweis:
Auch wenn hier von „Feldern“ gesprochen wird, benötigt man für diese Aufgabe **keine** Arrays!

Aufgabe 6.2
Erstellen Sie eine eine Klasse `Kredit`, die Kreditverträge modelliert.

Ein Kredit wird über eine zu übergebende Kreditsumme abgeschlossen.
Der jährliche Zinssatz (in Prozent) ist für alle Kredite gleich (zunächst zum Beispiel $p = 2.5$), könnte sich aber durch eine Methode `setZinssatz()` ändern.
Ein Kredit wir mit einer gleichbleibenden jährlichen Ratenzahlung abgezahlt. Diese Rate sollte mindestens so groß sein, wie der Zins des ersten Jahres. Der überschießende Betrag wird von der Kreditsumme abgezogen. So verringert sich von Jahr zu Jahr die „Restschuld".

- Implementieren Sie eine Methode `tilgungsPlan()`, die einen Tilgungsplan für einen Kredit (auf dem Monitor) ausgibt.
  Der Tilgungsplan gibt für jedes Jahr den gezahlten Zins und die verbleibende Restschuld aus.
- Zu Beginn ist die Restschuld gleich der Kreditsumme.
  Nach dem Verlauf eines Jahres berechnen sich die zu zahlenden Zinsen durch die Formel
  $zins = rest \cdot \frac{p}{100}$
  Die „neue" Restschuld ist dann $rest = rest + zins - rate$
- Die Methode `tilgungsPlan()` soll zeilenweise für soviele Jahre jeweils Zinsen und verbleibende Restschuld ausgeben, bis die Restschuld nur noch höchstens so groß wie die Jahresrate ist.

Hinweis:
Überlegen Sie genau, ob bzw. welche der Variablen

`kSumme`	Kreditsumme
`p`	Zinssatz
`rate`	jährlich gezahlte Rate
`zins`	Zinsen für das abgelaufene Jahr
`rest`	Restschuld

- statische oder nichtstatische Attribute eines Kredites sind
- dem Konstruktor übergeben werden sollten
- der Methode `tilgungsPlan()` als Eingabe-Parameter übergeben werden sollten
- innerhalb der Methode als lokale Variable definiert werden sollten.

Beispielrechnung:
Für einen Kredit über 10000,- Euro und einer jährlichen Rate von 1000,- Euro sollten die ersten Zeilen des Tilgungsplans so aussehen (Anzahl der Nachkommastellen, Tabulatoren etc. sind nebensächlich):

	Zins	Restschuld
1. Jahr:	250.0	9250.0
2. Jahr:	231.25	8481.25
3. Jahr:	212.03125	7693.28125
	⋮	