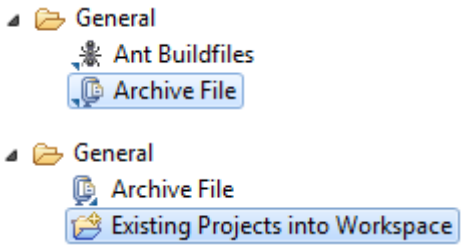
Aufgaben zur Abgabe als Prüfungsleistung für Prog I

- Allgemeine Regeln
  - Abgabetermin: Siehe OLAT
    - Bitte den Abgabetermin einhalten
    - Falls technische Probleme in OLAT auftreten, dann per e-mail direkt an Dozent senden
  - Verwenden Sie keine Umlaute ä ö ü und kein ß in Bezeichnern, Klassen etc.
    - Bitte legen Sie Ihre Klassen alle in einem Paket namens „`mypack`“ an, dann lässt es sich einheitlich testen; d.h. die erste Zeile jeder Klasse sollte lauten:
      `package mypack;`
  - Zielumgebung: Java 1.8 mit Eclipse 4.x
  - Falls zusätzliche Dokumente gefordert sind, dann bitte in einem der Formate `pdf` , `jpg` oder `gif` abgeben
  - Plagiate (d.h. Kopien oder veränderte Kopien anderer Lösungen) werden mit 0 Punkten sowohl für das „Original“ als auch die „Kopie“ bewertet
  - Falls Sie Quellen (Internet etc.) verwenden, müssen diese angegeben werden.
    - Informationen recherchieren ist in Ordnung, aber eine fertige Lösung aus dem Internet verwenden bzw. leicht abändern wird als Plagiat gewertet
  - Falls Sie nicht die komplette Aufgabe erfüllen, geben Sie Teile ab und ergänzen einen eigenen Text: Was geht bzw. was geht (gar) nicht

Aufgaben zur Abgabe als Prüfungsleistung für Prog I

- Abgabeformat:
  - Ihre Lösung bitte mittels OLAT bereitstellen
    - Ihr Projekt muss Ihren Namen haben, z.B. „`MuellerAufg1`“
    - OLAT lässt die Veränderung der abgegebenen Dateien zu, solange die Abgabe noch nicht abschlossen wurde (Button „Endgültige Abgabe“)
      - Empfehlung: Vermeiden Sie die „Endgültige Abgabe“; falls Sie dies doch gedrückt haben, können Sie zur Not eine neue Version direkt an den Dozenten senden f.mehler@th-bingen.de
    - Bitte Ihr Projekt <Name> so exportieren, dass es komplett alles enthält und problemlos wieder importiert werden kann und lauffähig ist
    - Export: Eclipse Projekt: Rechte Maustaste: Export: General: *Archive File* (dann: Save in zip Format, Finish)
    - Import: Auf ein beliebiges Eclipse-Projekt gehen, rechte Maustaste: Import: General: *Existing Projects into Workspace*, Select Archive File (Browse `MuellerAufg1.zip`), Finish
    - Führen Sie selbst vor Abgabe einen Probelauf durch, nicht lauffähiger Import gibt Punktabzug



1. Aufgabe: Abschreibungsverläufe

http://machopan.com/rauchzeichen/?p=247

- Abschreibung ist ein Begriff aus dem Rechnungswesen, der Wertminderungen von Vermögensgegenständen beschreibt
- Beispiele
  - Abnutzung durch Gebrauch, Verschleiß etc. von Fahrzeugen, Maschinen, Gebäuden usw.
  - Zahlenbeispiel: Eine Maschine kostet 21.000,-EUR und wird 7 Jahre lang genutzt, danach ist die Maschine 0,- EUR wert, d.h. jedes Jahr verliert die Maschine bei linearer Abschreibung 3.000,- EUR an Wert
- Ziele
  - Abschreibungen vermindern auf der Aktivseite der Bilanz den Wert des Vermögens und dienen somit der Ermittlung des aktuellen Vermögensstands
  - Die Anschaffungskosten werden in Form von Abschreibungen auf die voraussichtliche Nutzungsdauer verteilt und können kalkulatorisch in die Angebotserstellung bzw. Kostenrechnung einfließen

Vorgaben

- Erstellen Sie eine Klasse `Abschreibung`, die folgende drei Abschreibungsverfahren berechnet und ausgibt:
  - Lineare Abschreibung: Die Anschaffungskosten werden gleichmäßig auf die Jahre der Nutzungsdauer aufgeteilt, jedes Jahr wird der gleiche Betrag abgeschrieben
  - Geometrisch-degressive Abschreibung: Die Abschreibungsbeträge werden von dem aktuellen Buchwert mit einem konstanten Multiplikationsfaktor errechnet, z.B. jedes Jahr 20% des aktuellen Buchwerts
  - Methodenwechsel: „Um zu gewährleisten, dass das Wirtschaftsgut nach Ablauf der Nutzungsdauer vollständig abgeschrieben ist, ist es in einigen Ländern erlaubt, von der geometrisch-degressiven Abschreibung zur linearen Abschreibung zu wechseln. Im Jahr des Wechsels wird der Restbuchwert durch die Zahl der verbleibenden Abschreibungsjahre dividiert, sodass sich ab dem Wechsel gleich bleibende, also lineare Abschreibungsbeträge ergeben, die alle größer sind als die, die sich bei fortgeführter degressiver Abschreibung ergeben hätten“ (Quelle: https://de.wikipedia.org/wiki/Abschreibung)

Vorgaben

- Programmausgabe für:
  - Anschaffungskosten 21.000,-
  - Nutzungsdauer 7 Jahre
  - Prozentsatz für geometr.-degr. Abschreibung: 20%
- Beispiel
  - Im 4. Jahr beträgt die geometr.-degr. Abschreibung 20% von 10.752,- d.h. 2.150,- EUR
  - Die lineare Abschreibung ist ab dem 4. Jahr Restbuchwert / Restjahre = 10.752,- / 4 = 2.688,- und somit wird gewechselt, da größer als 2.150,- EUR

```
Lineare Abschreibung:
Jahr     Abschreibung    Restbuchwert
 0            0,00 EUR   21.000,00 EUR
 1        3.000,00 EUR   18.000,00 EUR
 2        3.000,00 EUR   15.000,00 EUR
 3        3.000,00 EUR   12.000,00 EUR
 4        3.000,00 EUR    9.000,00 EUR
 5        3.000,00 EUR    6.000,00 EUR
 6        3.000,00 EUR    3.000,00 EUR
 7        3.000,00 EUR        0,00 EUR
Geometrisch-degressive Abschreibung:
Jahr     Abschreibung    Restbuchwert
 0            0,00 EUR   21.000,00 EUR
 1        4.200,00 EUR   16.800,00 EUR
 2        3.360,00 EUR   13.440,00 EUR
 3        2.688,00 EUR   10.752,00 EUR
 4        2.150,40 EUR    8.601,60 EUR
 5        1.720,32 EUR    6.881,28 EUR
 6        1.376,26 EUR    5.505,02 EUR
 7        1.101,00 EUR    4.404,02 EUR
Methodenwechsel Abschreibung:
Jahr     Abschreibung    Restbuchwert
 0            0,00 EUR   21.000,00 EUR
 1        4.200,00 EUR   16.800,00 EUR
 2        3.360,00 EUR   13.440,00 EUR
 3        2.688,00 EUR   10.752,00 EUR
 4        2.688,00 EUR    8.064,00 EUR Wechsel linear
 5        2.688,00 EUR    5.376,00 EUR Wechsel linear
 6        2.688,00 EUR    2.688,00 EUR Wechsel linear
 7        2.688,00 EUR        0,00 EUR Wechsel linear
```

Vorgaben

- Beachten Sie folgende Vorgaben
  - Verwenden Sie Konstanten für feste Werte; die Eingabedaten sind im Quellcode hart codiert, können aber zentral an einer Stelle für alle drei Verfahren verändert werden. Dadurch kann ein Test auch mit anderen Werten wie z.B. mit 10 Jahren Nutzungsdauer durchgeführt werden
  - Angezeigt werden die jährliche Daten exakt wie auf der vorherigen Seite angegeben (inkl. Formatierung, Nachkommastellen, Ausrichtung, führende Leerzeichen etc.)
  - Lagern Sie wiederverwendbare oder zur Strukturierung sinnvolle Funktionalitäten in statische Funktionen aus
  - Trennen Sie die Berechnungen von den Ausgaben auf der Konsole

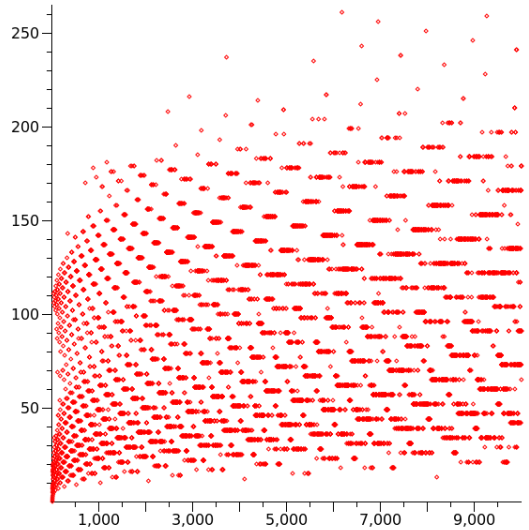
Hinweise

- Zur Formatierung können folgende Hilfsmittel hilfreich sein, Sie können aber auch eigene Formatierungshilfen recherchieren und verwenden
  - `"\t"` erzeugt einen Vorschub zum nächsten Tabulator, z.B. `System.out.println("Jahr \tAbschreibung");`
  - Die Klasse `DecimalFormat` dient zur Formatierung von Zahlen mit Vor- und Nachkommastellen, siehe Internet-Recherche
    - Beispiel:

```
import java.text.*;
double d = 1234.5678;
// mindestens 1 Vorkommastelle, genau 2 Nachkommastellen
DecimalFormat myFormat = new DecimalFormat("#0.00");
System.out.println(myFormat.format(d));
```

2. Aufgabe: Collatz-Problem

- Eine Collatz-Folge ist eine Zahlenfolge, die ausgehend von einer natürlichen Zahl n wie folgt berechnet wird
  - Falls n gerade ist, dann wird n zu $n/2$
  - Falls n ungerade ist, dann wird n zu $3n + 1$
- Beispiel: Mit der Zahl $n = 13$ ergibt sich die Collatz-Folge:
  - `13 -> 40 -> 20 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1`
  - Diese Folge hat die Kettenlänge (= Pfadlänge) 10, d.h. besteht aus 10 Zahlen
- Das Collatz-Problem wurde 1937 von Lothar Collatz gestellt
  - Die Vermutung ist, dass jeder beliebige Startwert mit dem Zyklus `4 -> 2 -> 1` endet, diese Vermutung ist bis heute weder bewiesen noch widerlegt
  - Alle bislang untersuchten Startwerte enden mit der Zahl 1



Die Kettenlängen in Abhängigkeit von den Startzahlen 1 bis 10.000
https://de.wikipedia.org/wiki/Collatz-Problem

Vorgaben

- Schreiben Sie ein Programm, das die Collatz-Folge für den konkreten Startwert 13 berechnet und auf der Konsole exakt wie folgt formatiert ausgibt:

```
13 -> 40 -> 20 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1
```

- Berechnen Sie für die Startwerte von $n = 1, \ldots, 50.000$ folgende vier Kennzahlen:
  1. Welcher Startwert erzeugt die längste Kollatz-Folge (= maximale Kettenlänge)?
  2. Die Kollatz-Folge erzeugt Elemente, die größer als der Startwert sein können (z.B. die 40 in obiger Folge). Welches maximale Element wird über alle Startwerte n hinweg erzeugt?
  3. Verhältnis "Maximales Element zu Startelement": Mit größeren Startzahlen ergeben sich in der Regel auch größere Elemente der Folge. Interessant ist deshalb das Verhältnis des größten Elements zum Startelement
     - Beispiel: Die Startzahl $c_0 = 27$ bringt es mit $c_{77} = 9232$ immerhin auf fast das 342-fache
     - http://www.rzbt.haw-hamburg.de/dankert/spezmath/html/collatzproblem.html
  4. Welche Kettenlänge kommt am häufigsten vor?

Vorgaben

- Beispiel Zwischenrechnung nur für $n = 1, \ldots, 10$ (soll nicht ausgegeben werden, da im Programm bis 50.000, nur zur Erläuterung):

```
n = 1   Laenge: 1       max Element: 1  rel. Groesse: 1.0
n = 2   Laenge: 2       max Element: 2  rel. Groesse: 1.0
n = 3   Laenge: 8       max Element: 16 rel. Groesse: 5.33
n = 4   Laenge: 3       max Element: 4  rel. Groesse: 1.0
n = 5   Laenge: 6       max Element: 16 rel. Groesse: 3.2
n = 6   Laenge: 9       max Element: 16 rel. Groesse: 2.67
n = 7   Laenge: 17      max Element: 52 rel. Groesse: 7.43
n = 8   Laenge: 4       max Element: 8  rel. Groesse: 1.0
n = 9   Laenge: 20      max Element: 52 rel. Groesse: 5.78
n = 10  Laenge: 7       max Element: 16 rel. Groesse: 1.6
```

- Ergebnis (muss wie folgt ausgegeben werden):
  - Wenn Kettenlängen gleich oft vorkommen, dann ein beliebiges davon

```
Maximale Laenge: 19 bei n = 9
Maximales Element: 52 bei n = 7
Maximale relative Groesse: 7.43 bei n = 7
Am haeufigsten kommt die Kettenlaenge 1 vor, und zwar 1 mal
```

Vorgaben

- Die Klasse heißt `Collatz.java`
- Alle Vorgaben der Aufgabe 1 gelten auch hier (zentral änderbare Werte, statische Funktionen usw.)
- Versuchen Sie, den Algorithmus nur einmal zu implementieren (Vermeidung von Redundanz)
  - Der Algorithmus darf mehrfach aufgerufen werden
- Die Kennzahl Verhältnis "Maximales Element zu Startelement“ soll auf zwei Nachkommastellen kaufmännisch gerundet werden
- Verwenden Sie keine Arrays (Felder)
- Mit Punktabzug können Sie auch Teile weglassen, z.B. ist die 4. Kennzahl nur relativ aufwändig zu ermitteln