HSLU Hochschule Luzern

Objektorientierte Programmierung (OOP)

Übung SW03: Kontrollstrukturen

Themen: Selektionen (O04) und Iterationen (O05)
Zeitbedarf: ca. 180min.

Roland Gisler, Version 1.6.4 (FS24)

1 Einfache Bedingungen mit `if`-Statements formulieren

1.1 Lernziele

- Selektionen mit **`if`**-Statements formulieren.
- Vor- und Nachteile von **`else-if`**-Statement erkennen.
- **`switch`**-Statement anwenden.

1.2 Grundlagen

Die Grundlagen für folgenden Übungen sind die Inhalte aus den Inputs O04_IP_Selektion.pdf und O05_IP_Iteration.pdf. Erstellen Sie in BlueJ am besten ein neues Projekt für die dritte Semesterwoche. Sofern kein spezieller Hinweis erfolgt, können Sie zur Vereinfachung alle verlangten Methoden in einer einzigen, grossen **`Demo`**-Klasse definieren. Für die Tests erzeugen Sie interaktiv ein Objekt dieser Klasse und können dann alle Methoden interaktiv aufrufen.

1.3 Aufgaben

a.) Implementieren Sie je eine Methode **`int max(int a, int b)`** und **`int min(int a, int b)`** welche jeweils den grösseren bzw. den kleineren **Wert** der beiden übergebenen aktuellen Parameter zurückliefern.

b.) Schreiben Sie eine Methode **`int max(int a, int b, int c)`** welche den maximalen Wert aller drei Parameter zurückliefert. Überlegen Sie sich dazu verschiedene Algorithmen bzw. Lösungen. Probieren Sie unterschiedliche Varianten aus (**`max1(…)`**, **`max2(…)`** etc.), welche nur mit **`if`**-Statements oder auch mit **`else-if`**-Statements arbeiten! Haben Sie daran gedacht, dass Sie für diese Aufgabe auch die Lösung von a) wiederverwenden könnten? Wenn nein, implementieren Sie auch noch diese Lösung! Welche halten sie für die Beste? Vergleichen Sie eventuell mit anderen Studierenden!

c.) Verwendet eine ihrer **`if`**-Lösungen von b) verschachtelte **`if`**-Statements und Sie kommen damit schon in die dritte (oder gar vierte) Einrückungsebene?

- Wenn ja: Legen Sie den Code einen Moment zu Seite und überlegen Sie rein algorithmisch. Wie würden Sie vorgehen, wenn es n Zahlen (also sehr viele) wären? Erkennen Sie die einfachste Lösung?
- Wenn nein: Sie haben offenbar sehr kompakten Code geschrieben. Hat ihre Lösung vielleicht dennoch einen Nachteil? Hinterfragen Sie kritisch die Laufzeiteigenschaften, wenn der erste, zweite oder dritte Parameter den Maximalwert trägt!

d.) Erinnern Sie sich an die Temperatur-Aufgabe von letzter Woche? Wir möchten die **`Temperatur`**-Klasse nun mit einer Methode erweitern, welche uns für ein übergebenes chemisches Element den jeweiligen Aggregatszustand (fest, flüssig, gasförmig), abhängig vom

gesetzten Temperaturwert zurückliefert. Als formalen Parameter verwenden Sie die chemische Elementbezeichnung (als **`String`**) und implementieren es für die drei Elemente **N** (Stickstoff), **Hg** (Quecksilber) und **Pb** (Blei). Gehen Sie bei der Implementation davon aus, dass später noch viele weitere Elemente ergänzt werden könnten.

Hinweis: Zur Vereinfachung soll der Returntyp ein **`String`** mit dem Inhalt "**`fest`**", "**`flüssig`**" etc.) sein, auch wenn wir das in wenigen Wochen noch viel eleganter[1] lösen werden.

e.) Vermutlich haben Sie die Teilaufgabe d) mit einem **`switch`**-Statement gelöst?
Wenn nein, schreiben Sie ihre Methode um (besser: kopieren Sie sie in eine Alternative) und vergleichen Sie die beiden Implementationen. Was sind die Vor- und Nachteile gegenüber einer Implementation mit **`if`**- oder **`else-if`**-Statements?

f.) Stellen Sie sich vor, Sie müssten die Funktion für alle Elemente des Periodensystems (das sind mehr als hundert) implementieren. Sie würde dadurch sehr gross und unübersichtlich werden. Wie könnten Sie das zumindest etwas vermindern? Setzen Sie ihre Idee um und hinterfragen Sie ihren Code auch hier immer wieder kritisch!

g.) Implementieren Sie eine Klasse **`Point`**, welche einen einfache, zweidimensionalen Punkt mit **`x-`** und **`y`**-Koordinaten (Datentyp **`int`** reicht völlig aus) repräsentiert.

h.) Ergänzen Sie die Klasse **`Point`** mit einer Methode **`int getQuadrant()`**, welche zurückliefert in welchem (geometrischen) Quadrant sich ein Punkt befindet. Testen Sie die Funktion gewissenhaft!

i.) Haben Sie daran gedacht, dass Punkte auch auf den Achsen oder sogar auf dem Nullpunkt liegen könnten?

[1] Wenn Sie schon wissen, was Enumerationen sind und wie man sie einsetzt: Nur zu!

2 Iterationen mit `while`-, `do-while`- und `for`-Schleifen

2.1 Lernziele

- Die verschiedenen Schleifenkonstrukte von Java verwenden.
- Vor- und Nachteile der einzelnen Konstrukte kennen.
- Adäquate Bedingungen formulieren.

2.2 Grundlagen

Sofern kein spezieller Hinweis erfolgt, können Sie zur Vereinfachung alle verlangten Methoden in einer einzigen, grossen **`Demo`**-Klasse definieren. Für die Tests erzeugen Sie ein Objekt dieser Klasse und können dann alle Methoden interaktiv aufrufen.

2.3 Aufgaben

a.) Implementieren Sie eine Methode, welche mit einer Iteration die Zahlen von **`0`** bis **`10`** Zeilenweise ausgibt. Die Formatierung können Sie vernachlässigen, verwenden Sie einfach **`System.out.println(…)`** für die Ausgabe. Welches Schleifenkonstrukt haben Sie dafür verwendet und warum?

b.) Implementieren Sie die Aufgabe a) zusätzlich noch mit den beiden verbleibenden Schleifenkonstrukten. Welches Konstrukt ist für diese Aufgabe am schlechtesten, und welches am besten geeignet und warum?

c.) Verwenden Sie eine **`while`**-Schleife um eine mit **`0.9f`** initialisierte **`float`**-Variable durch die iterative Addition von **`0.000025f`** zu erhöhen. Die Schleife soll beendet werden, wenn die **`float`**-Variable den Wert **`1.0f`** erreicht. Testen Sie die Methode.

d.) Wenn Sie bei c) einen (typischen) Fehler gemacht haben kann es sein, dass Sie gerade eine Endlos-Schleife erlebt haben. Versuchen Sie zu ergründen, warum das passiert ist. Wieviele Iterationen erwarten sie eigentlich? Prüfen Sie nach, indem Sie eine **`int`**-Variable ergänzen, welche die Anzahl der Iterationen zählt. Das Resultat sollte Sie überraschen!

Achtung: Korrigieren Sie ggf. noch ihre Abbruchbedingung, damit die Schleife nun auch sicher ein Ende findet!

e.) Implementieren Sie eine Alternative zu d) mit einem **`for`**-Loop welcher genau **`4000`** Iterationen macht. Was ist dann das Resultat der Summation? Spätestens jetzt sollten Ihnen klar sein, was seit Teilaufgabe c) alles schlimmes passiert ist?!

f.) Implementieren sie eine Methode

```
void printBox(final int width, final int height)
```

welche mit **`System.out.print[ln](…)`** und **`width`** Spalten breite und **`height`** Zeilen hohe Box, bestehend aus dem Zeichen «**`#`**» ausgibt!

Beispiel: Für den Aufruf **`printBox(10,4)`** soll folgende Ausgabe erfolgen:

```
##########
#        #
#        #
##########
```

Das ist übrigens eine (sehr bescheidene) Form von ASCII-Art: https://de.wikipedia.org/wiki/ASCII-Art :-)

g.) Wenn sie motiviert sind und noch mehr zu Funktionen, Selektionen und Iterationen üben wollen, probieren Sie doch mal ein (evt. grössenvariables) Schweizerkreuz auszugeben! ☺