Proseminar Programmiermethodik

Universität Innsbruck - Institut für Informatik

A. Blaas, M. Mayerl, B. Murauer, M. Tschuggnall

Blatt 1: Java Basics

Eclipse

Erzeugen Sie zunächst ein neues Projekt in Eclipse. Importieren Sie in dieses neue Projekt die gegebene Java-Datei `MyFirstProgram.java` (OLAT). Bearbeiten Sie anschließend die untenstehenden Aufgaben.
Zögern Sie nicht, bei Unklarheiten über die Aufgabenstellungen nachzufragen. Fragen Sie in der Diskussionsrunde und benutzen Sie das Forum auf OLAT, um bei schwierigen Fragestellungen Hilfe zu erhalten. Wenn Ihnen nach dem Lösen der Aufgaben manche Inhalte immer noch unklar sind, schreiben Sie dies jederzeit gerne in eine Textdatei und fügen Sie diese Ihrer Lösung hinzu.

Diskussionsteil

a) [★] Führen Sie das Programm aus, indem Sie im Menü Run As > Java Application klicken. Damit startet Eclipse automatisch die `main`-Methode und führt den darin enthaltenen Code aus. Im Fall des oben angeführten Beispiels wird "Hello, World!" ausgegeben und eine Schleife durchlaufen.

b) [★★] Schreiben Sie eine Methode, die die ersten 10 Fibonacci-Zahlen ausgibt. Legen Sie dafür eine neue Klasse `Fibonacci`[1] an, in der Sie eine `main`-Methode erstellen (oder kopieren Sie die Methode aus `MyFirstProgram`). Verwenden Sie für die Berechnung der Fibonacci-Zahlen einen **iterativen** Ansatz und eine `while`-Schleife.

c) [★] Zum Debuggen von Code können sogenannte Breakpoints gesetzt werden. Klicken Sie dazu mit der rechten Maustaste auf den linken Rand des Fensters, das den Code beinhaltet (grauer Bereich direkt links vom Code; neben den Zeilennummern[2]) und wählen Sie "Toggle breakpoint"; alternativ können Sie auch das Tastaturkürzel Shift + Ctrl + B verwenden). Anschließend wählen Sie im Menü Run > Debug as > Java Application. Eclipse schaltet nun automatisch in die Debug-Ansicht, in der Sie den Code, die Variablen (und deren Belegung) und Breakpoints sehen. Sie können nun durch den Code "steppen", indem Sie die Kürzel F5, F6 oder F8 verwenden. Sehen Sie sich die Datei `Debug.java` (OLAT) an und veruchen Sie festzustellen, welchen Wert die Variable x beim dritten Durchgang der Schleife besitzt.

d) [★★] Versuchen Sie, folgende Fragen für sich selbst zu beantworten:

- Was ist der Unterschied zwischen einer Klasse und einem Objekt?
- Was ist der Unterschied zwischen einer Funktion (wie z.B. in C) und einer Methode?

---

[1] https://en.wikipedia.org/wiki/Fibonacci_number

[2] Zeilennummern sind in Eclipse standardmäßig ausgeblendet. Sie können diese aktivieren, indem Sie mit der rechten Maustaste auf den grauen Bereich links neben dem Code klicken und "Show Line Numbers" aktivieren. Alternativ können Sie im Menü Window > Preferences > General > Editors > Text Editors die Checkbox "Show line numbers" setzen und mit "Apply" oder "OK" die Änderung bestätigen.

Übungsteil (selbstständig zu lösen)

Aufgabe 1 (Interne Operationen) [3 Punkte]

Sehen Sie sich folgende Ausdrücke und Operationen an und geben Sie das Ergebnis aus (Wert und Datentyp). Es empfiehlt sich, eine kleine `main`-Methode zu schreiben, um diese Ausdrücke zu testen. Sie können das Ergebnis dieser Operationen entweder mit dem Debugger oder mit Hilfe von `System.out.println()` ausgeben. Erklären Sie, wie welches Ergebnis zustande kommt. Falls eine Operation so nicht ausführbar ist, geben Sie den Grund dafür an. Geben Sie Ihre Antworten per Textdatei ab.

Hinweis ⚠

Geben Sie ausschließlich **UTF-8**-formattierte Textdateien ab. Verwenden Sie einen modernen Texteditor, um dies zu bewerkstelligen (oder bearbeiten Sie die Dateien direkt mit Eclipse). Geeignete/empfohlene Editoren sind VIM[a], Emacs[b], Sublime[c], Atom[d] oder auch Notepad++[e].

[a] http://www.vim.org
[b] https://www.gnu.org/software/emacs/
[c] https://www.sublimetext.com/
[d] https://atom.io/
[e] https://notepad-plus-plus.org/

Ausdruck	Ergebnis
`6 * 4 / 3`	
`1 << 8 % 3`	
`(short) Integer.MAX_VALUE`	
`(double) 23 / 11`	
`(double) (23/11)`	
`30f`	
`4e4D`	
`12 * 1.2 != 18`	
`"Luck=Coffee+" + 'chocolate' + 2`	
`"Luck=Coffee+" + "chocolate" + 2`	
`1 == 18 % 4 && 7 > 4 || true`	
`1 == 18 % 4 ? 7 : 4`	

Aufgabe 2 (for, if, switch, Arrays) [3 Punkte]

Das gegebene Programm `ShoppingCart.java` (OLAT) simuliert einen Einkaufswagen, der Artikelpreise für hinzugefügte Artikel enthält. Für diesen Einkaufswagen sollen nun Rabatte berechnet werden: einerseits gibt es Rabatt aufgrund des Gesamtpreises der enthaltenen Artikel im Einkaufswagen, und andererseits aufgrund der Anzahl der enthaltenen Artikel.

Hinweis ⚠

Sie müssen den bereits vorgegebenen Code nicht zur Gänze verstehen, manche Elemente werden erst während des Semesters erarbeitet (z.B. was bedeutet `static`, `private`, etc.).

a) *2 Punkte* Sehen Sie sich den Code der Klasse an und lesen Sie die Kommentare genau durch. Folgen Sie den Anweisungen in den TODO-Kommentaren und vervollständigen so den Code

zu einem lauffähigen Programm. Testen Sie Ihre Implementierung, indem sie die bereitgestellte `main`-Methode ausführen. Sie müssen dabei nur die Berechnung selbst erstellen und die Ausgabe schreiben. Bitte ändern Sie nur die Teiles des Codes, die mit `//TODO` markiert sind. Verwenden Sie ausschließlich die Grundkonstrukte von Java und noch keine fortgeschrittenen Tools (wie z.B. ArrayLists). Dokumentieren Sie ihr Vorgehen in den Kommentaren auf Englisch.

b) *1 Punkt* Sie sehen am Anfang der Klasse 4 verschiedene Einkaufswägen (jeweils 3 sind stets auskommentiert). Kommentieren Sie nun abwechselnd genau einen Einkaufswagen ein und führen Sie Ihr Programm aus. Ist dies für alle Einkaufswägen möglich? Was sind die Unterschiede bzw. welcher Code wird jeweils ausgeführt?

Geben Sie die vervollständigte Java-Klasse und die Ausgabe der Konsole für die einzelnen Einkaufswägen als eigene Dateien ab.

Aufgabe 3 (Objekte und Klassen) [4 Punkte]

a) *1 Punkt* Erstellen Sie eine neue Klasse mit dem Namen `Person`. Diese Klasse soll zwei Felder enthalten: `firstname (String)` und `lastname (String)`. Erstellen Sie für jedes dieser Felder eine getter- und setter-Methode[3].

b) *0.5 Punkte* Erstellen Sie einen Konstruktor, der beide Felder der Klasse setzt[4]. Zusätzlich soll eine Methode erstellt werden, die den vollständigen Namen der Person auf der Konsole ausgibt.

c) *0.5 Punkte* Schreiben Sie dann eine `main`-Methode, in der Sie folgendes ausführen:

- Erstellen Sie zwei Personen mit beliebigen Namen.
- Rufen Sie die Methoden auf, die die Namen der Personen auf der Konsole ausgeben.

d) *2 Punkte* Erweitern Sie die Klasse `Person` nun um eine Methode `greet`, die als Parameter ein Objekt der selben Klasse entgegennimmt. In dieser Methode soll die Person, für die die Methode aufgerufen wurde, die übergebene Person grüßen. Testen Sie ihre Implementierung, indem Sie den unten angeführten Code in Ihre `main`-Methode einbauen. Ein Beispiel für einen Aufruf dieser Methode könnte so aussehen:

```
1  Person p1 = new Person("Max", "Mustermann");
2  Person p2 = new Person("Donald", "Duck");
3  p1.greet(p2);
4  // Auf der Konsole sollte an dieser Stelle ein Gruß erscheinen, z.B.
5  // Max Mustermann sagt "Hallo" zu Donald Duck.
```

Wichtig: Laden Sie bitte Ihre Lösung (.txt, .java oder .pdf) in OLAT hoch und geben Sie mittels der Ankreuzliste auch unbedingt an, welche Aufgaben Sie gelöst haben. Die Deadline dafür läuft am Vortag des Proseminars um 23:59 (Mitternacht) ab.

[3] Diese können mit Eclipse automatisch generiert werden, sobald die Felder angelegt wurden. Das kann entweder im Menü unter Source > Generate Getter and Setters oder per Tastenkürzel Alt + Shift + S > R erfolgen

[4] Auch das ist in Eclipse automatisch möglich: Source > Generate Constructor using Fields oder Alt + Shift + S > C