Prof. Dr. J. Giesl
J.-C. Kassing, M. Leven Rosarius, N. Lommen, E. Meyer

Allgemeine Hinweise:

- Die **Deadline** zur **Abgabe** der Hausaufgaben ist am **Donnerstag, den 30.10.2025, um 14 Uhr**.
- Der **Workflow** sieht wie folgt aus. Die Abgabe der Hausaufgaben erfolgt **im Moodle-Lernraum** und kann nur in **Zweiergruppen** stattfinden. Dabei müssen die Abgabepartner*innen **dasselbe Tutorium** besuchen. Nutzen Sie ggf. das entsprechende **Forum** im Moodle-Lernraum, um eine*n Abgabepartner*in zu finden. Es darf **nur ein*e** Abgabepartner*in die Abgabe hochladen. Diese*r muss sowohl die **Lösung** als auch den **Quellcode** der Programmieraufgaben hochladen. Die Bepunktung wird dann von uns für **beide** Abgabepartner*innen **separat** im Lernraum eingetragen. Die Feedbackdatei ist jedoch nur dort sichtbar, wo die Abgabe hochgeladen wurde und muss innerhalb des Abgabepaars **weitergeleitet** werden.
- Die **Lösung** muss als PDF-Datei hochgeladen werden. Damit die Punkte beiden Abgabepartner*innen zugeordnet werden können, müssen **oben** auf der **ersten Seite** Ihrer Lösung die **Namen**, die **Matrikelnummern** sowie die **Nummer des Tutoriums** von **beiden** Abgabepartner*innen angegeben sein.
- Der **Quellcode** der Programmieraufgaben muss als `.zip`-Datei hochgeladen werden und **zusätzlich** in der PDF-Datei mit Ihrer Lösung enthalten sein, sodass unsere Hiwis ihn mit Feedback versehen können.

  Auf diesem Blatt muss Ihre Codeabgabe Ihren vollständigen Java-Code in Form von `.java`-Dateien enthalten. Aus dem Lernraum heruntergeladene Klassen dürfen nicht mit abgegeben werden.

  Stellen Sie sicher, dass Ihr Programm von `javac` **in der Version 25 akzeptiert** wird. Ansonsten werden keine Punkte vergeben.

  Generell sollten alle Programme für alle Eingaben terminieren, solange in der Spezifikation (bzw. der Aufgabenstellung) nicht explizit etwas anderes verlangt wird!
- Einige Hausaufgaben müssen im Spiel Codescape gelöst werden. Klicken Sie dazu im Lernraum rechts in der ausklappbaren Seitenleiste im Block „Codescape" auf den angegebenen Link. Diese Aufgaben werden getrennt von den anderen Hausaufgaben gewertet.
- Aufgaben, die mit einem * markiert sind, sind Sonderaufgaben mit erhöhtem Schwierigkeitsgrad. Sie tragen nicht zur Summe der erreichbaren Punkte bei, die für die Klausurzulassung relevant ist, jedoch können diese Aufgaben ganz normal im Tutorium vorgerechnet werden und die in solchen Aufgaben erreichten Punkte werden Ihnen ganz normal gutgeschrieben.

Übungsaufgabe 1 (Überblickswissen):

a) Wann ist eine `for`-Schleife einer `while`-Schleife gegenüber vorzuziehen? Wann sollte man eher eine `while`-Schleife nutzen?

b) Was sind die Vor- und Nachteile der `switch`-Anweisung gegenüber einer `if ... else ...`-Anweisung?

Übungsaufgabe 2 (Casting):

Bestimmen Sie den Typ und das Ergebnis der folgenden Java-Ausdrücke und begründen Sie Ihre Antwort. Sollte der Ausdruck nicht typkorrekt sein, begründen Sie, worin der Fehler besteht.
Dabei seien die Variablen `x`, `y` und `z` wie folgt deklariert: `int x = 1; int y = 2; int z = 3;`

a) `10 / 3`

b) `10 / 3.`

c) `x == y ? x > y : y < z`

d) `(byte) (127 + 1)`

e) `'x' + y + z`

f) `1 || 0`

Hausaufgabe 3 (Casting): (16 Punkte)

Bestimmen Sie den Typ und das Ergebnis der folgenden Java-Ausdrücke und begründen Sie Ihre Antwort. Sollte der Ausdruck nicht typkorrekt sein, begründen Sie, worin der Fehler besteht.
Dabei seien die Variablen `x`, `y` und `z` wie folgt deklariert: `int x = 17; int y = 10; int z = 123;`

a) `'z' + z + "z"`

b) `"x" + y - z`

c) `"x" + y + z`

d) `x + y + "z"`

e) `true ? 2147483647+1 : 0L`

f) `2042 / 'x' == x ? 2 : 3.`

g) `5 / true ? 2 : 3.`

h) `(byte) 256 == (short) 2147483648L`

Übungsaufgabe 4 (Programmierung):

In dieser Aufgabe geht es um die Ein- und Ausgabe in Java. Dafür soll die von Java bereitgestellte Klasse `IO`[1] genutzt werden. Um einen String `str` auszugeben, nutzen Sie `IO.println(str)`. Benutzen Sie `IO.readln("Bitte geben Sie einen String ein: ")`, um einen Wert vom Typ `String` einzulesen. Sie können die von Java vordefinierte Methode `Integer.parseInt(str)` benutzen, um den `String str` in eine Zahl umzuwandeln. Mit `Integer.parseInt(IO.readln("Bitte geben Sie eine Zahl ein: "))` können Sie also einen Wert vom Typ `int` einlesen.

Schreiben Sie ein Java-Programm, welches einen einfachen Taschenrechner darstellt. Der Taschenrechner liest initial eine Zahl ein. Auf diese initiale Zahl (und eine später eingelesene weitere Zahl) kann nun eine der zwei folgenden Rechenoperationen angewandt werden:

- `ADD`: Addiere die aktuelle Zahl und eine weitere eingelesene Zahl.
- `SUB`: Subtrahiere eine weitere eingelesene Zahl von der aktuellen Zahl.

Nachdem eine dieser zwei Operationen ebenfalls eingelesen wurde, wird eine zweite Zahl eingelesen und das Ergebnis entsprechend der Operation berechnet. Danach wird das eben berechnete Ergebnis zwischengespeichert, eine weitere Operation und eine weitere Zahl werden eingelesen. Das ursprüngliche Zwischenergebnis wird nun benutzt, um ein neues Zwischenergebnis zu berechnen. Dieser Prozess wird solange wiederholt, bis die Eingabe `STOP` getätigt wurde. Danach wird das Endergebnis ausgegeben und das Programm wird beendet. Bei fehlerhaften Eingaben kann sich Ihr Programm beliebig verhalten.
Ein Ablauf des Programms könnte z.B. so aussehen:

[1] Siehe hierzu auch `https://docs.oracle.com/en/java/javase/25/docs/api/java.base/java/lang/IO.html`.

```
Bitte geben Sie eine Zahl ein: 12
Bitte geben Sie eine Rechenoperation (ADD oder SUB) oder STOP ein: ADD
Bitte geben Sie eine Zahl ein: 10
Aktuelles Ergebnis: 22
Bitte geben Sie eine Rechenoperation (ADD oder SUB) oder STOP ein: SUB
Bitte geben Sie eine Zahl ein: 5
Aktuelles Ergebnis: 17
Bitte geben Sie eine Rechenoperation (ADD oder SUB) oder STOP ein: STOP
Endergebnis: 17
```

Hinweise:

- Im Lernraum befindet sich ein Java-Template für Ihre Lösung.
- Verwenden Sie `str1.equals(str2)` um zu überprüfen, ob zwei `Strings str1` und `str2` die gleiche Sequenz von Zeichen beinhalten.
- Sie können `System.exit(1)` verwenden, um den `main`-Methodenaufruf im Falle von fehlerhaften Eingaben zu beenden.

Hausaufgabe 5 (Programmierung): (34 Punkte)

Implementieren Sie ein Programm, welches einen Bankautomaten simuliert. Um einen String `str` auszugeben, nutzen Sie `IO.println(str)`. Um einen Wert vom Typ `int` einzulesen, benutzen Sie `Integer.parseInt(IO.readln("Bitte eine Zahl eingeben: "))`. Analog können Sie mittels `IO.readln("Bitte einen String eingeben:")` einen String einlesen.
Jedes Konto hat initial immer 1000 EURO Guthaben. Verwenden Sie hierfür eine `int`-Variable `guthaben`.
Sie sollen nun verschiedene Operationen implementieren:

- `GUTHABEN`: Hier soll das aktuelle Guthaben der Kundin ausgegeben werden.
- `EINZAHLEN`: Die Kundin soll einen positiven EURO-Betrag eingeben können. Die Bank berechnet 5% Gebühren für das Einzahlen. Das Guthaben soll also um 95% (aufgerundet) des eingezahlten Betrags erhöht werden.
- `ABHEBEN`: Die Kundin soll einen positiven EURO-Betrag eingeben können. Falls die Kundin genug Guthaben zur Verfügung hat, dann soll das Geld ausgezahlt werden und das Guthaben entsprechend angepasst werden. Hat die Kundin nicht ausreichend Guthaben auf ihrem Konto, dann soll ein entsprechender Fehler ausgegeben werden.
- `ZINSESZINS`: Die Kundin soll eine Jahresanzahl `n` eingeben. Nun wird das Guthaben für `n` Jahre um 4% Zinsen pro Jahr erhöht (und jeweils auf eine ganze Zahl aufgerundet). Die Bank berechnet natürlich Kontoführungsgebühren von 2 EURO pro Jahr. Sollte zu einem Zeitpunkt das Guthaben negativ sein aufgrund der Kontoführungsgebühren, dann soll die Berechnung abgebrochen werden und das Guthaben auf 0 gesetzt werden.
- `STOP`: Hier soll ebenfalls das Guthaben ausgegeben werden. Jedoch soll danach das Programm beendet werden.

Die Operationen werden solange wiederholt, bis `STOP` eingegeben wurde. Quittieren Sie Ihre Berechnungen mit entsprechenden Ein- und Ausgaben. Orientieren Sie sich hier am folgenden Beispiellauf:

```
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
EINZAHLEN
Wie viel Geld möchten Sie einzahlen? Es wird eine Gebühr von 5% berechnet.
200
Ihr Guthaben beträgt 1190 EURO.
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
```

```
ABHEBEN
Wie viel Geld möchten Sie abheben?
-190
Betrag negativ.
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
ABHEBEN
Wie viel Geld möchten Sie abheben?
190
Ihr Guthaben beträgt 1000 EURO.
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
ZINSESZINS
Wie viele Jahre wollen Sie Ihr Guthaben verzinsen?
2
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
GUTHABEN
1078
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
ABHEBEN
Wie viel Geld möchten Sie abheben?
1077
Ihr Guthaben beträgt 1 EURO.
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
ZINSESZINS
Wie viele Jahre wollen Sie Ihr Guthaben verzinsen?
2
Kein Guthaben mehr verfügbar!
Bitte geben Sie eine Operation (GUTHABEN,EINZAHLEN,ABHEBEN,ZINSESZINS,STOP) ein:
STOP
Ihr Guthaben beträgt 0 EURO. Vielen Dank und bis zum nächsten Mal!
```

Hinweise:

- Verwenden Sie die Funktion `Math.ceil()`, um Zahlen aufzurunden. Zum Beispiel gibt `Math.ceil(0.11)` den `double`-Wert `1.0` zurück.

Hausaufgabe 6 (Programmierung): (32* Punkte)

In dieser Aufgabe erweitern wir die Sprache D korrekt geklammerter Ausdrücke aus der Hausaufgabe 5 des vorherigen Blatts um die Klammernpaare `[]` und `<>`. Das heißt wir betrachten die Sprache D_3, welche von der EBNF Grammatik

$$S = \{\texttt{"("}\, S\, \texttt{")"} \mid \texttt{"["}\, S\, \texttt{"]"} \mid \texttt{"<"}\, S\, \texttt{">"}\}$$

erzeugt wird.
Die folgenden Wörter sind beispielsweise in der Sprache D_3 enthalten:

$$\varepsilon \qquad \texttt{()<>} \qquad \texttt{(<[]>)[]} \qquad \texttt{<<>>[()]}$$

Hingegen sind die folgenden Wörter *nicht* in der Sprache D_3 enthalten:

$$\texttt{](} \qquad \texttt{<][} \qquad \texttt{([<><])} \qquad \texttt{[(])}$$

Schreiben Sie ein Java-Programm, welches untersucht, ob ein Ausdruck korrekt geklammert ist. Hierzu soll das Programm zunächst mittels `IO.readln("Bitte geben Sie einen String ein: ")` einen String von der Kommandozeile einlesen. Anschließend soll untersucht werden, ob es sich hierbei um einen korrekten Klammerausdruck handelt, d.h. ob der eingelesene String in der Sprache D_3 enthalten ist. Ist dies der Fall, so soll das Programm dieses durch Ausgabe des Strings `"Ja"` signalisieren und anschließend terminieren. Sie können hierzu die in Java vordefinierte Methode `IO.println` benutzen. Andernfalls soll das Programm den String `"Nein"` ausgeben und mit dem Fehlercode `1` terminieren.
Für einen *nicht* in D_3 enthaltenen String könnte ein Ablauf des Programs z.B. so aussehen:

```
Bitte geben Sie einen String ein: [(])
Nein
```

Für einen in D_3 enthaltenen String könnte sich folgender Ablauf ergeben:

```
Bitte geben Sie einen String ein: (<[]>)[]
Ja
```

Hinweise:

- Benutzen Sie den Aufruf `System.exit(1)`, um mithilfe des Fehlercodes `1` zu terminieren.
- Benutzen Sie einen Wert vom Typ `String`, um zu speichern, welche geöffneten Klammern als nächstes geschlossen werden müssen.
  - Für einen `String str` wertet `str.length()` zur Länge von `str` aus und `str.charAt(i)` berechnet für einen `int i` den `char`-Wert an Stelle `i` in `str`, sofern dieser existiert. Die erste Stelle hat hierbei den Index `0`. Es gilt also `"(([<".length() == 4` und `"(([<".charAt(0) == '('`.
  - Für einen `String str` mit `str.length() > 0` berechnet `str.substring(1)` den Teilstring von `str` ab dem Index `1`. Der Ausdruck `"(([<".substring(1)` wertet also zu dem `String`-Wert `"([<"` aus.

Hausaufgabe 7 (Deck 2): (Codescape)

Lösen Sie die Missionen von Deck 2 des Codescape Spiels. Ihre Lösung für die Codescape Missionen wird nur dann für die Zulassung gezählt, wenn Sie Ihre Lösung vor der einheitlichen Codescape Deadline am Freitag, den 30.01.2026, um 23:59 Uhr abschicken.