

Übungen 01.12.-05.12.2025
Blatt 7

Abgabe der Hausaufgaben bis spätestens **Freitag, 05.12.2024 um 23:59 Uhr** via git.
Besprechung der Hausaufgaben in der nächsten Woche.

Hinweise:

- **Alle Klassen einer Aufgabe gehören ins gleiche Package wie die main-Klasse.** Sollen Sie in Hausaufgabe 42 also eine Klasse **DieKlasse** erstellen, und die **main-Klasse H42_main** befindet sich in **h42**, dann muss auch **DieKlasse** sich in **h42** befinden.
- Sie müssen jede Klasse, die Sie erstellen, in einer **eigenen** Datei speichern.
- **Attribute in Klassen sind grundsätzlich als `private` zu deklarieren.** Abweichungen davon werden wir explizit in der Aufgabestellung notieren.
- **Wenn eine Methode ein Objekt übergeben bekommt oder zurückgibt,** dann meinen wir damit eigentlich die **Referenz auf das Objekt.**
- **Nicht-statische Methoden werden, genau wie Attribute, über die Referenz und mittels des Punktoperators aufgerufen.** Die Syntax ist identisch zu statischen Methoden (Mehr dazu erfahren Sie in VL08).
- Hinweis zu *genau dann*, *wenn* und *wennn*:
 - **wenn** beschreibt nur eine Bedingung, unter der etwas wahr ist. Es sagt nichts darüber aus, was passiert, **wenn** die Bedingung nicht erfüllt ist. Beispiel: pos ist true, **wenn** zahl eine positive Zahl ist.
→ Was passiert, wenn zahl keine positive Zahl ist, bleibt offen.
 - **genau dann, wenn** beschreibt eine genaue logische Beziehung. Es legt fest, dass etwas wahr ist, wenn und nur wenn eine bestimmte Bedingung erfüllt ist. Beispiel: pos ist true, **genau dann, wenn** zahl eine positive Zahl ist.
→ Das bedeutet auch, dass pos false ist, wenn zahl keine positive Zahl ist.



Before methods
and objects

Methods
and objects

Präsenzaufgabe 1 [multFrac (Arbeiten mit der Klasse Bruch)]

Erstellen Sie im Repository ein Package **p1** und fügen Sie diesem Package die Klasse **P1_main** hinzu, welche die **main**-Methode enthält.

(a) Schreiben Sie in die Hauptklasse die folgende Methode:

- Modifikatoren: **static** und **public**
- Rückgabotyp: **void**
- Name: **multFrac**
- Übergabeparameter: **int z1, int n1, int z2, int n2**
Dabei steht **z1** für den Zähler und **n1** für den Nenner der ersten Zahl. Analoges gilt für **z2** und **n2**, für die zweite Zahl.

Die Methode **multFrac** soll die Werte so multiplizieren, als würden **z1** und **n1** einen Bruch bilden, und **z2** und **n2** ebenfalls. Das Ergebnis soll auf die Konsole ausgegeben werden.

- Beispiel: **multFrac(5,2,3,4)** entspricht der Multiplikation von $\frac{5}{2}$ und $\frac{3}{4}$. Das Ergebnis ist $\frac{15}{8}$. Also soll **15:8** auf die Konsole ausgegeben werden.
- (b) Erzeugen Sie die öffentliche Klasse **Bruch** mit den öffentlichen **int**-Attributen **z** und **n**. Erstellen Sie einen Konstruktor für die Klasse **Bruch**, der zwei **int**-Variablen **zaehler** und **nenner** übergeben bekommt und diese **z** und **n** zuweist.
- (c) Überladen Sie die Methode **multFrac** mit einer weiteren Methode, welche zwei **Bruch**-Objekte **a** und **b** übergeben bekommt. Das Ergebnis der Multiplikation der beiden Brüche **a** und **b** soll auf der Konsole ausgegeben werden.
- (d) Erzeugen Sie in der Klasse **Bruch** die öffentliche Methode **toString**, welche sinnvoll die in Bruch gespeicherten Inhalte wiedergibt. Dabei liefert **toString** einen **String** zurück und hat keinen Übergabeparameter.
- Beispiel: Ruft man für den Ergebnisbruch aus (a) dann im Rahmen einer **print**-Anweisung die **toString**-Methode auf, wird **15:8** auf die Konsole ausgegeben.

Was passiert eigentlich, wenn Sie statt der gerade beschriebenen **print**-Anweisung einfach nur **...println(b)** als Anweisung angeben, wenn **b** einer der oben genannten Brüche Bruch ist?

Präsenzaufgabe 2 [Franz]

Erstellen Sie im Repository ein Package **p2** und fügen Sie diesem Package die Klasse **P2_main** hinzu, welche die **main**-Methode enthält.

- Erstellen Sie eine öffentliche Klasse namens **LetterGame**. Versetzen Sie diese Klasse mit den öffentlichen Attributen **keyString** und **word** (Typ: **String**). Den **String keyString** belegen Sie mit dem Satz „FRANZ JAGT IM KOMPLETT VERWAHRLOSTEN TAXI QUER DURCH BAYERN!“. Das Attribut **word** wird mit dem leeren String ("") initialisiert.
- Versetzen Sie **LetterGame** weiterhin mit der öffentlichen Methode **setWord**, welche ein **int**-Array namens **a** als Übergabeparameter erhält. Der Rückgabetyp ist **void**. Dieses **int**-Array hat der Reihe nach den Index gespeichert, welches die Zelle im **String keyString** angibt, aus dem das nächste Zeichen gelesen werden soll. Die ausgelesenen Zeichen fügen Sie zu einem String zusammen und weisen es dem Attribut **word** zu.
 - Beispiel: Das Array **a** ist belegt mit **0, 1, 2, 43**. Dann muss **word** den Wert FRAU erhalten, denn F ist bei **keyString** der Buchstabe mit Index 0, R ist bei **keyString** der Buchstabe mit Index 1, A ist bei **keyString** der Buchstabe mit Index 2, und U ist bei **keyString** der Buchstabe mit Index 43.
- Testen Sie Ihre Lösung mit weiteren **int**-Arrays, zum Beispiel {6, 7, 23, 7} oder {14, 2, 0, 0, 19, 19}.
- * Gelingt es Ihnen, diese Aufgabe auch „rückwärts“ zu lösen? Nun soll **LetterGame** ein weiteres Attribut erhalten, ein **int**-Array namens **a**. Schreiben Sie eine Methode namens **findCode**, die als Übergabeparameter einen **String** erhält, sagen wir **s**. Diesen **String** soll **findCode** so „übersetzen“, dass **setWord** genau das Wort **s** wieder herstellen kann.
 - Beispiel: Wird das Wort **UNI** an **findCode** übergeben, dann soll **a** wie folgt belegt werden: {43, 3, 11}.
- **Hinweis:** Wir erinnern an die Methoden **charAt** und **indexOf** aus der Vorlesung. Weiterhin können Sie mit dem Plus-Operator einen String und ein **char** miteinander konkatenieren, sodass das Ergebnis dieses Ausdrucks wieder ein **String** ist.

Hausaufgabe 1 [Zahl-Klasse]

Gehen Sie in das Package **h1**. Die **main**-Methode finden Sie in der Klasse **H1_main**. Sie dürfen die Attribute zu Beginn mit beliebigen Werten initialisieren.

- Erstellen Sie die öffentliche Klasse **Zahl**. Diese Klasse hat die öffentlichen und **nicht** statischen Attribute **even**, **small** und **positive** (Typ: **boolean**) und **num** (Typ: **int**).
- **Zahl** hat die folgenden öffentlichen und **nicht** statischen Methoden: **setEven**, **setSmall** und **setPositive**. Diese haben alle keinen Übergabeparameter und der Rückgabetyt ist **void**.
- Wird **setEven** aufgerufen, erhält **even** genau dann den Wert **true**, wenn in **num** ein gerader Wert gespeichert ist. Zahlen gelten hier als gerade, wenn ihr Betrag durch 2 teilbar ist. Somit sind also z.B. -16, 0 und 80 gerade.
 - Zur Erinnerung (siehe Hinweise oben): Wenn **setEven** aufgerufen wird und **num** einen **ungeraden** Wert speichert, **muss** durch den Aufruf von **setEven** der Wert von **even** auf **false** gesetzt werden.
- Wird **setPositive** aufgerufen, erhält **positive** genau dann den Wert **true**, wenn der Wert in **num** echt größer 0 ist.
- Wird **setSmall** aufgerufen, erhält **small** genau dann den Wert **true**, falls der Wert in **num** echt kleiner als 100 ist.

Hausaufgabe 2 [Arrays verwenden]

Gehen Sie in das Package **h2**. Die **main**-Methode finden Sie in der Klasse **H2_main**. Die hier beschriebene Methode schreiben Sie direkt in die Klasse **H2_main**.

Schreiben Sie die folgende Methode:

- Modifikatoren: **static** und **public**
- Rückgabetyt: **int[]**
- Name: **change**
- Übergabeparameter: **int[] a, int[] b, int start, int end**

Die Methode arbeitet folgendermaßen:

- Ist **end** größer als **start** und der Inhalt von **a** nicht identisch mit dem Inhalt von **b**, so wird ein neues Array erzeugt, das die gleichen Inhalte hat wie **a**. Dieses Array wird zurück gegeben.

- Ist **end** größer als **start** und ist der Inhalt von **a** und der Inhalt von **b** identisch, so wird **a** aufsteigend sortiert. In einem neu erzeugten Array werden anschließend nur noch die Werte gespeichert, die nach der Sortierung von **a** in den Zellen **start** bis **end** enthalten sind, allerdings exklusive **end**. Dieses neu erzeugte Array wird zurück gegeben.
- Ist **end** höchstens so groß wie **start**, dann wird ein Array zurückgegeben, das zwar erzeugt wurde, aber keine Zellen hat (also ein leeres Array, nicht **null**).
- Verwenden Sie die Methoden **compare**, **sort** und **copyOfRange** aus dem Package **Arrays**, um diese Aufgabe zu lösen.

Hinweise und Beispiele:

- Sie dürfen von Folgendem ausgehen: Der Wert von **start** ist nie größer als die Länge von **a** und nie kleiner als 0. Das gleiche gilt für **end**.
- Den Inhalt von zwei **int**-Arrays bezeichnen wir als identisch, wenn in beiden Arrays die gleichen Werte in der gleichen Reihenfolge und Vielfachheit enthalten sind.
- Beispiele
 - Rückgabe von **change([2,7,1,9], [5,6,7], 2, 4)** ist **[2,7,1,9]**.
 - Rückgabe von **change([2,7,1,9], [2,7,1,9], 2, 4)** ist **[7,9]**.
 - Rückgabe von **change([2,7,1,9], [2,7,1,9], 3, 2)** ist **[]**.

Hausaufgabe 3 [Mensch]

Gehen Sie in das Package **h3**. Die **main**-Methode finden Sie in der Klasse **H3_main**. Sie dürfen die Attribute zu Beginn mit beliebigen Werten initialisieren.

- Schreiben Sie eine öffentliche Klasse namens **Mensch**. Die Klasse hat die **nicht** statischen, aber öffentlichen Attribute **name** (Typ: **String**), **gebJahr** und **alter** (Typ: **int**). Die Klasse hat weiterhin die nicht statischen, aber öffentlichen Methoden **setName**, **setGebJahr** und **setAlter**, sowie **getName**, **getGebJahr** und **getAlter**.
- Die Methode **setName** erhält einen **String** als Übergabeparameter und hat keinen Rückgabewert. Der übergebene **String**-Wert wird dem Attribut **name** zugewiesen.
- Die Methode **setGebJahr** erhält einen **int**-Wert als Übergabeparameter und hat keinen Rückgabewert. Der übergebene Wert wird dem Attribut **gebJahr** zugewiesen. Es handelt sich hierbei um eine vierstellige Jahreszahl.
- Die Methode **setAlter** hat keinen Übergabeparameter und **void** als Rückgabebetyp und berechnet das aktuelle Alter (nur unter Berücksichtigung des vierstelligen Geburtsjahres) und weist das Ergebnis dem Attribut **alter** zu.
- Die Methode **getName** hat keinen Übergabeparameter und als Rückgabewert einen **String**, und liefert den Wert von **name** zurück.

- Die Methoden **getGebJahr** und **getAlter** haben keinen Übergabeparameter und liefern als Rückgabebetyp ein **int** und den Wert der Attribute **gebJahr** bzw. **alter**.
 - Beispiel: Ein Objekt vom Typ **Mensch** kann den Namen **Elsa** erhalten, das Geburtsjahr **2010** und damit (weil wir im Jahr 2025 sind) das Alter **15**.

Hinweis:

- Sie können davon ausgehen, dass **gebJahr** ≤ 2025 ist.