Methoden – Übungsaufgabe 1

- Programmieren Sie eine Methode zur Umrechnung von Zentimetern in Zoll. Der Methode bekommt eine (Komma-) Zahl übergeben. Der berechnete Zollwerte ist der Rückgabewert der Methode. Vor der Rückgabe wird der Zollwert auf zwei Nachkommastelle gerundet. Entwickeln Sie in der Main-Methode mehrere Methodenaufrufe der entwickelten Umrechnungsmethode.
- Hinweis: 1 Zoll = 2,54 cm
- Hinweis: Verwenden Sie Math.round zum Runden

- Zeitvorgabe = 10 Minuten

Methoden – Übungsaufgabe 2

- Analysieren Sie den folgenden Programmcode. Wie lautet die erzeugte Ausgabe? Begründen Sie ihre Antwort. Zeitvorgabe = 5 Minuten

```
public class MethodeAusgabe {

    public static int zahl1 = 7;
    public static int zahl2 = 35;

    public static void main (String[] args) {

        int zahl1 = 10;
        int zahl2 = 27;
        summieren(zahl1,zahl2);

    }

    public static void summieren(int zahl1, int zahl2) {

        System.out.println(zahl1 + zahl2);
        System.out.println(MethodeAusgabe.zahl1 + MethodeAusgabe.zahl2);

    }

}
```

Methoden – Übungsaufgabe 3

- Erstellen Sie ein Programm, dass die Berechnung des verzinsten Kapitals am Jahresende vornimmt. Das Programm beinhaltet mehrere Methoden.
  - *1. Methode: Berechnung des verzinsten Kapitals. Übergabeparameter = Kapital und Zinssatz (z.B. 2.5)*
  - *2. Methode: Ausgabe des berechneten Kapitals, gerundet auf 2 Nachkommastellen*
  - *3. Methode: Main-Methode zur Steuerung der ersten und zweiten Methode. Programmieren Sie die Berechnung für 3 Jahre mit einer Ausgabe des verzinsten Kapitals am Ende jeden Jahres*
- Testen Sie das Programm mit einem Kapital von 50.000€ und einem Zinssatz von 1.3%.
- Hinweis: Verwenden Sie Math.round zum Runden

- Zeitvorgabe = 15 Minuten

Typkonvertierung - Übungsaufgabe 1

Erstellen Sie ein Programm, dass vier Integer Variablen zahl1, zahl2, zahl3 und zahl4 mit den Werten 71, 85, 84 und 33 initialisiert. Das Programm muss alle Zahlen in Charakter-Werte konvertieren und zu einem String zusammenbauen, der anschließend ausgeben wird.

Zeitvorgabe = 10 Minuten

Typkonvertierung - Übungsaufgabe 2

Erstellen Sie ein Programm, dass eine Benutzereingabe in Form einer beliebigen Ganzzahl entgegennimmt. Das Programm muss den kleinsten Werteberich ermitteln, in den die Ganzzahl passt, eine entsprechende Typkonvertierung und eine Ausgabe über den kleinsten Wertebereich (-> Namen) vornehmen.

Hinweis:

- Verwenden Sie für die Benutzereingabe die Klasse Scanner
  - *Scanner input = new Scanner (System.in); // Initialisieren der Benutzereingabe*
  - *Variable = input.nextLong(); // Abfragen der Benutzereingabe*
  - *Input.close(); // zum beenden der Benutzereingabe*
- Überlegen Sie, wie Sie die Wertebereichgrenzen prüfen
- Zeitvorgabe = 15 Minuten

Arrays – Übungsaufgabe 1

- Programmieren Sie eine Methode, die alle Elemente eines Arrays ausgibt. Die Methode bekommt ein String-Array (ohne definierte Größe) als Parameter übergeben. Die Ausgabe erfolgt auf der Konsole. Die Methode hat keinen Rückgabewert. Initialisieren Sie in der Main-Methode ein String-Array, dass Sie an die entwickelte Methode übergeben.
- Zeitvorgabe = 10 Minuten

Arrays – Übungsaufgabe 2

- Entwickeln Sie ein Programm, das prüft, ob in einem String-Array ein gesuchter String enthalten ist. Neben der Main-Methode ist eine Methode zu entwickeln, die die besagte Prüfung vornimmt und je nach Ergebnis true oder false zurückgibt. Rufen Sie diese Methode aus der Main-Methode auf.
- Zeitvorgabe = 15 Minuten

Übungsaufgabe: Finde die Fehler

- In dem nachfolgenden Programmcode sind 9 Fehler enthalten. Finde und korrigiere die Fehler.
- Zeitvorgabe = 10 Minuten

```
3  public FehlerFinden {
4      public static main (String[] args) {
5          long fibo=0; fibo1 = 1, fibo2 = 0;
6          int n = 5,
7          if (n==0) ( fibo = 0; }
8          else {
9              if (n = 1) { fibo = 1; }
10             else {
11                 for (a = 1; a < n; ++a ) {
12                     fibo =fibo1 + fibo2;
13                     System.out.println("aktuelles Ergebnis + fibo);
14                     fibo2 = fibo1;
15                     fibo1 = fibo;
16                 }
17             }
18         system.out.println("Fibaonaci Gesamtergebnis:" + fibo);
19         }
20     }
21 }
```

Übungsaufgabe: Schreibtischtest

- Nehmen Sie für den folgenden Programmcode einen Schreibtischtest vor.
- Zeitvorgabe = 15 Minuten

```
public static void main (String[] args) {

    int laenge;
    int breite;
    int hoehe;
    boolean formWuerfel;

    laenge = hoehe = 20;
    breite = 18;
    breite++;
    ++breite;
    formWuerfel = (laenge == breite && laenge == hoehe);
    System.out.println("Ist die Form ein Würfel " + formWuerfel);

    laenge = 10;
    hoehe = 10;
    breite = (breite * laenge);
    formWuerfel = (laenge == breite && laenge == hoehe);
    System.out.println("Ist die Form ein Würfel: "  + formWuerfel);

    laenge = breite;
    hoehe *= 20;
    formWuerfel = (laenge == breite && laenge == hoehe);
    System.out.println("Ist die Form ein Würfel: "  + formWuerfel);

}
```

Übungsaufgabe: Code-Analyse

- Analysieren Sie den Code. Wie oft findet die Ausgabe statt?
- Zeitvorgabe = 15 Minuten

```
public static void main (String[] args) {

    int counter = 1;
    int laufendeNr = 1;

    while (++counter < 10) {

        System.out.println (laufendeNr + ": Ausgabe!");
        laufendeNr = laufendeNr+1;

    }

}
```

Übungsaufgabe: Klassen und Objekte

- Entwickeln Sie ein Programm für eine Verkaufsverwaltung. Dazu ist die Klasse Kunde zu entwickeln. Jeder Kunde hat die Attribute Name, Vorname, Nachname, Wohnort, PLZ und ein ganzzahlige ID. Der Konstruktor der Klasse „Kunde“ erhält die ID, Name, Wohnort und PLZ. Zusätzlich benötigt die Klasse Kunde ein Methode, die prüft, ob der übergebene Name ein Leerzeichen hat. Falls ja, teilt die Methode den Name in Vor- und Nachname auf. Falls nein, ist auf der Konsole eine entsprechende Meldung auszugeben.
- Zusätzlich ist die Klasse Verkaufsverwaltung zu entwickeln. Über diese Klasse wird das Array Kundenverwaltung mit 1000 möglichen Kunden angelegt. Zusätzlich werden 3 Kunden angelegt, die dem Array hinzuzufügen sind. Anschließend ist die Kundenverwaltung mit einer Schleife zu durchlaufen und für jeden enthaltenen Kunde ist die Methode zur Aufteilung des Namens auszuführen. Anschließend sind die Vor und Nachnamen aller Kunden auf der Konsole auszugeben.
- Hinweis: Verwenden Sie die Methode substring, um einen String aufzuteilen
- Hinweis: Verwenden Sie die Methode indexOf um die Position eines Zeichens in einem String zu ermitteln
- Hinweis: Finden Sie einen Weg, um zu prüfen, ob ein Index in einem Array belegt ist
- Zeitvorgabe = 25 Minuten

Übungsaufgabe: Passwort-Validierung

- Entwickeln Sie ein Programm, dass eine Methode enthält, die ein übergebenes Passwort (= String) auf verschiedene Regeln prüft:
  - *Das Passwort hat ein Länge von mindestens 10 Zeichen*
  - *Das Passwort enthält mindestens einen Großbuchstaben*
  - *Das Passwort enthält mindestens einen Kleinbuchstaben*
  - *Das Passwort enthält mindestens eine Ziffer*
  - *Das Passwort enthält mindestens eines der Sonderzeichen _ % / **
- Falls alle Regeln positiv getestet sind, gibt die Methode ein True zurück. Andernfalls ein False.
- Zusätzlich ist ein Main-Methode zu entwickeln, die die Methode zur Passwort-Validierung aufruft und das Ergebnis auf der Konsole ausgibt.
- Schreiben Sie Source-Code oder Pseudo-Code
- Zeitvorgabe = 20 Minuten

Übungsaufgabe: Halde

- Zeigen Sie den Zustand der Halde in Form einer Tabellenvisualisierung nachdem die Main-Methode ausgeführt worden ist und bevor die Garbage Collection die Speicherbereinigung vornimmt. Zeigen Sie alle temporären Variablen auf der Halde.
- Zeitvorgabe = 25 Minuten

```
public class SupermarktVerwaltung {

    public static void main (String[]args) {

        Supermarkt einkaufsland = new Supermarkt ("Einkaufsland",2);
        Artikel cola = new Artikel ("Cola", 1.59);
        Artikel brot = new Artikel ("Bauernbrot", 2.99);
        Artikel wasser = new Artikel ("Mineralwasser", 0.99);

        einkaufsland.inSortimentAufnehmen(cola);
        einkaufsland.inSortimentAufnehmen(brot);
        einkaufsland.artikel[0] = brot;
        einkaufsland.artikel[1] = wasser;

        einkaufsland.preiseerhöhen(10);

    }
}
```

```
public class Artikel {

    String name;
    double preis;

    public Artikel (String name, double preis) {

        this.name = name;
        this.preis = preis;

    }
}
```

```
public class Supermarkt {

    String supermarktName;
    Artikel[] artikel;
    int artikelIndex;

    public Supermarkt (String name, int artikelAnzahl) {

        artikelIndex = 0;
        supermarktName = name;
        int supermartkNummer = 10;
        artikel = new Artikel[artikelAnzahl];

    }

    public void inSortimentAufnehmen (Artikel a) {

        artikel[artikelIndex] = a;
        artikelIndex++;

    }

    public void preiseerhöhen (int preisErhoehung) {

        int preisErhoehung1 = preisErhoehung+1;;

        for (int i = 0; i < artikel.length; i++) {

            artikel[i].preis += preisErhoehung1;

        }

    }

    public void preiseAusgeben () {

        for (int i = 0; i < artikel.length; i++) {

            System.out.println(artikel[i].name + ": " + artikel[i].preis);

        }

    }
}
```

Übungsaufgabe: Klassendiagramm für eine Hochschulverwaltung – Teil 1

Die Software zur Hochschulverwaltung stellt für jede Hochschule exakt eine Personalverwaltung zur Verfügung. Beim ersten Start der Personalverwaltung sind für die Hochschule die Stammdaten Name und Adresse anzugeben. Die Personalverwaltung enthält je eine Personalkartei für Professoren, Dozenten und eingeschriebene Studenten an der Hochschule. Alle verwalteten Personen haben Vor- und Nachname und je eine ID passend zur der Personenart. Es sind keine Begrenzungen für die Anzahl verwalteter Personen bekannt.

Professoren haben in der Hochschulverwaltung die Möglichkeit Studenten einen Nachricht zu senden. Dozenten haben in der Hochschulverwaltung die Möglichkeit ihre Honorarabrechnung abzuschicken. Für das Abschicken sind über das User-Interface der Name der Vorlesung und die Anzahl der Unterrichtseinheiten einzugeben. Studenten haben die Möglichkeit Fragen an Ihre Dozenten zu schicken. Dazu müssen die Studenten den Nachnamen des Dozenten im User-Interface angeben. Da die Hochschulverwaltung noch nicht abschließend entwickelt ist, erhalten weder Professoren, Dozenten noch Studenten eine visuelle Rückmeldung bei der Durchführung der genannten Funktionen.

Zusätzlich erfüllt die Personalverwaltung zwei weitere Aufgabe. Anhand eines Parametern in Form eines Worts (Dozenten, Professoren oder Studenten) zeigt die Personalverwaltung die Anzahl der aktuell verwalteten Personen. Zusätzlich kann die Personalverwaltung über zwei Funktionen für die Professoren die Gehaltsabrechnungen und für die Dozenten die Honorarabrechnungen erstellen. Dazu ist die ID des Dozenten einzugeben. Nach Ausführen der Funktionen wird lediglich angezeigt, ob die Ausführung erfolgreich gewesen ist oder nicht. Bei der Entwicklung der Hochschulverwaltung wurde sehr viel Wert darauf gelegt, möglichst wenig Source-Code zu schreiben. Aus diesem Grund wurde an sinnvollen Stellen Vererbung verwendet.

- Erstellen Sie ein Klassendiagramm anhand der vorliegenden Informationen. Treffen Sie Annahmen für fehlende Informationen. Verzichten Sie auf Konstruktoren und Getter/Setter-Methoden
- Zeitvorgabe = 25 Minuten

Übungsaufgabe: Klassendiagramm für eine Hochschulverwaltung – Teil 2

In einem 2. Release wurden für die Hochschulverwaltung weitere Funktionen ausgeliefert. Diese sind nachfolgend beschrieben:

Die Hochschulverwaltung kann beliebig viele Studiengänge verwalten. Je Studiengang können zusätzlich bis zu 100 verschiedene Vorlesungen verwaltet werden. Sollte die Hochschule im Verwaltungsprogramm gelöscht werden, werden automatisch auch alle verwalteten Studiengänge gelöscht. Gleiches gilt für Vorlesungen bei Löschen eines Studiengangs. Ein Studiengang hat einen Namen, eine Studiengangs-ID und den beabsichtigten Studienabschluss. Eine Vorlesung hat einen Namen, eine Vorlesungs-ID, eine maximale Teilnehmeranzahl und die Nummer des Semesters, in dem die Vorlesung gehalten wird.

Zusätzlich übernimmt de Hochschulverwaltung die Zuordnung von Studiengängen zu leitenden Professoren. Ein Professor kann maximal 2 Studiengänge leiten. Das Anlegen eines Studiengangs kann nicht ohne die Angabe des leitenden Professors erfolgen. Darüber hinaus ordnet die Hochschulverwaltung den Vorlesungen Dozenten zu. Eine Vorlesung kann von maximal 2 Dozenten gehalten werden. Im Rahmen der Suche nach Dozenten kann eine Vorlesung temporär ohne zugeordneten Dozenten existieren. Ein Dozent kann bis zu 5 Vorlesungen halten.

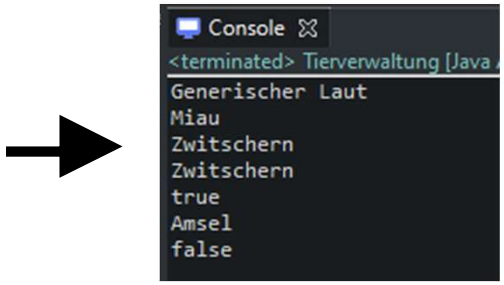
Die letzte neue Funktion ermöglicht es Studenten sich in Vorlesungen einzuschreiben. Dazu muss der Student in der Hochschulverwaltung eine Funktion ausführen, in der die Vorlesungs-ID anzugeben ist. Die Funktion zeigt eine Erfolgsmeldung sofern noch mindestens 1 freier Platz in der Vorlesung vorhanden ist. Ein Student muss mindestens 5 Vorlesungen je Semester besuchen.

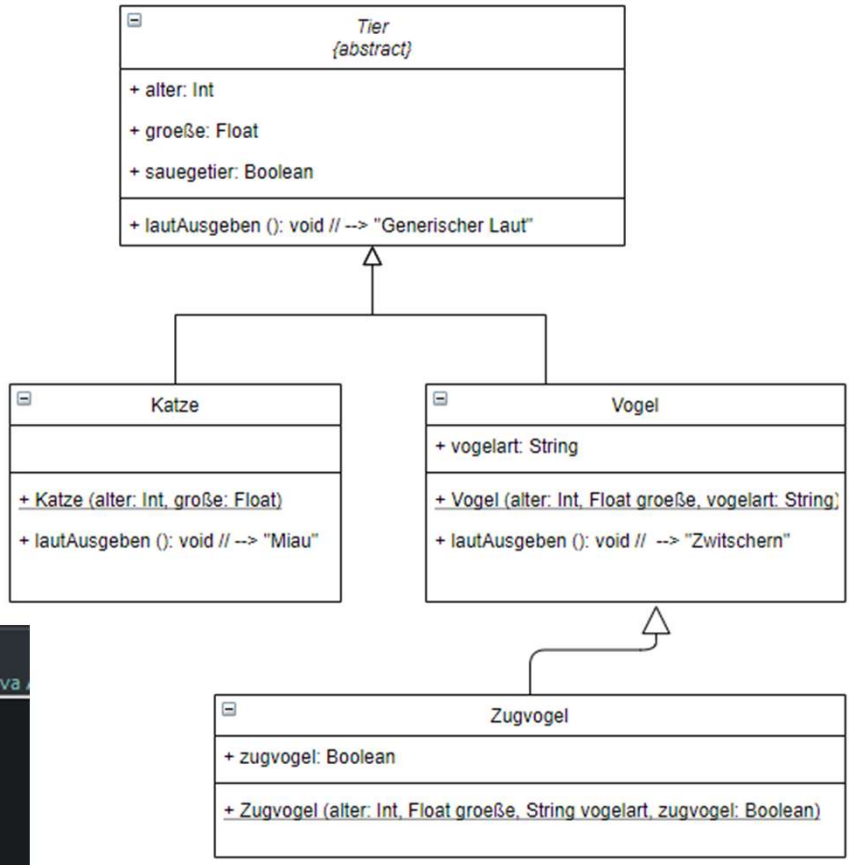
- Erweitern Sie das Klassendiagramm anhand der vorliegenden Informationen. Treffen Sie Annahmen für fehlende Informationen. Verzichten Sie auf Konstruktoren und Getter/Setter-Methoden
- Zeitvorgabe = 25 Minuten

Übungsaufgabe: Vererbung

- Gegeben ist ein Klassendiagramm
- Gegeben ist der Source-Code einer Tierverwaltung (-> Main Methode)
- Schreiben Sie den Source-Code / Pseudo-Code für das Klassendiagramm, der zusammen mit dem gegebenen Source-Code der Tierverwaltung die gezeigte Konsolenausgabe erzeugt
- Zeitvorgabe = 20 Minuten

```
public class Tierverwaltung {

    public static void main (String[] args) {

        Katze maylo = new Katze (5, 25.0f);
        Vogel mia = new Vogel (2, 10.5f, "Amsel");
        Zugvogel josi = new Zugvogel (3, 80, "Storch", true);

        maylo.lautAusgeben();
        mia.lautAusgeben();
        josi.lautAusgeben();

        System.out.println (maylo.saeugetier);
        System.out.println (mia.vogelart);
        System.out.println (josi.saeugetier);

    }
}
```





Übungsaufgabe: Sichtbarkeit und statische Methoden

- Entwickeln Sie eine Fuhrparkverwaltung mit 3 Klassen (Fuhrpark, Fuhrparkverwaltung, Auto).
- Ein Auto hat einen Namen, Kilometerstand und ein Passwort zum Ändern des Kilometerstands. Alle Variablen sind privat. Verwenden Sie Getter/Setter-Methoden für den Zugriff auf die privaten Variablen. Vor der Änderung des Kilometerstands ist im Auto das Passwort abzufragen.
- Der Fuhrpark hat eine Variable für die Anzahl der enthaltenen Fahrzeuge. Die Variable ist privat. Verwenden Sie Getter/Setter Methoden. Die Methode zur Änderung der Fahrzeuganzahl muss sowohl mit einer Erhöhung als auch Verringerung der Fahrzeuganzahl umgehen können. Zusätzlich ist zu berücksichtigen, dass von der Klasse Fuhrpark keine Instanz erzeugt wird (!!).
- Bei der Aufnahme eines neuen Fahrzeugs in den Fuhrpark ist ein Auto-Objekt zu erzeugen. Während der Erzeugung ist die Fahrzeuganzahl im Fuhrpark um 1 zu erhöhen.
- Setzen Sie beim Start der Fuhrparkverwaltung die Fahrzeuganzahl auf 5. Nehmen Sie anschließend zwei weitere Autos auf. Ändern Sie bei einem der Fahrzeuge den Kilometerstand. Fragen Sie anschließend die Fahrzeuganzahl des Fuhrparks ab.
- Zeitvorgabe = 20 Minuten