Grundlagen der Programmierung

Objektorientierte Programmierung

Beschreibung:

Es ist Freitag und heute wiederholen wir alles, was diese Woche dran war. Dafür gründen wir unseren eigenen kleinen Streichelzoo, mit lauter spannenden Tieren und auch lauter spannenden Aufgaben, die es zu erledigen gilt.

Hinweise zur Bearbeitung:

Achte auf einen sauberen Quellcode, insbesondere Einrückungen sind wichtig!

Wichtige Materialien für heute:

Handbuch: Objektorientierte Programmierung (OOP) → Verstehen→ Kapselung

Handbuch: Objektorientierte Programmierung (OOP) → Implementieren → Klasse + Eigenschaften + Methoden implementieren, instanziieren und nutzen

Aufgabe 1 - Wir gründen unseren kleinen Streichelzoo

Als erstes erstellst du ein neues Projekt, in dem du arbeiten wirst. Benenne es passend. In dem Projekt erstellst du dann im kotlin Ordner eine “Main.kt” Datei (File).

Anleitung:

- ➔ Rechtsklick auf den kotlin Ordner in eurem Projekt und dann auf "New".
- ➔ Nun "Kotlin Class/File" auswählen und einen passenden Namen finden.
- ➔ Als Letztes darunter **"File"** auswählen und mit "Enter"-Taste bestätigen.

Schreibe darin eine main()-Funktion. Diese main()-Funktion verwenden wir in den Aufgaben, um unseren Code zu testen.

Aufgabe 2 - Unsere Gäste

Erstelle jetzt eine Kotlin-Klasse mit dem Namen *"Besucher"*.

Anleitung:

- ➔ Rechtsklick auf den kotlin Ordner in eurem Projekt und dann auf "New".
- ➔ Nun "Kotlin Class/File" auswählen und einen passenden Namen finden.
- ➔ Als Letztes darunter **"Class"** auswählen und mit "Enter"-Taste bestätigen.

Die Besucher-Klasse soll die Eigenschaften **name** (String) und **alter** (Int) haben.

Die Besucher-Klasse hat eine Methode, in der sich der Besucher mit seinem Namen vorstellt. Das wird in der Konsole ausgegeben.

Test:

Teste deine Klasse, indem du in der Main.kt in der main()-Funktion einen Besucher mit dem Namen “Franz” und dem Alter 35 erstellst. Lass den Besucher grüßen, wird sein Name in der Konsole ausgegeben?

Aufgabe 3 - Tiere unseres Streichelzoos

Jetzt brauchen wir eine Basisklasse mit dem Namen Tier. Erstelle dazu die Klasse Tier. Die Klasse hat die Eigenschaften **name** (String), **gewicht** (Double), **alter** (Double) und **geschlecht** (String). Die Klasse ist vererbbar.

Test:

Erstelle (in der Main.kt in der main()-Funktion) ein Tier mit dem Namen “Helga”, dem Gewicht 35.3, dem Alter 12 und dem Geschlecht “Weiblich”. Gib dann alle Eigenschaften des Tieres in der Konsole aus. Sind alle Eigenschaften richtig?

Aufgabe 4 - Fähigkeiten unserer Tiere

Unsere Tiere können ein paar grundlegende Fähigkeiten. Erweitere die Klasse Tier mit Methoden.

1. Ein Tier kann sich bewegen. Dabei wird in der Konsole ausgegeben, dass sich das Tier bewegt hat.

2. Ein Tier kann Geräusche machen. Dabei wird in der Konsole ausgegeben, dass das Tier Geräusche gemacht hat. Die Methode hierfür ist überschreibbar.

3. Ein Tier kann gestreichelt werden. Die Methode hierfür erhält als Parameter einen Besucher. In der Konsole wird dann ausgegeben, welches Tier von welchem Besucher gestreichelt wurde.

4. Ein Tier kann von einem Besucher gefüttert werden. Die Methode hierfür erhält als Parameter einen Besucher. In der Konsole wird dann ausgegeben, welches Tier von welchem Besucher gefüttert wurde. Zusätzlich erhöht sich das Gewicht des Tieres um 2%.

Test:

Erstelle (in der Main.kt, in der main()-Funktion) ein Tier mit dem Namen “Helga”, dem Gewicht 35.3, dem Alter 12 und dem Geschlecht “Weiblich”. Lass das Tier sich dann bewegen, gibt es eine Ausgabe in der Konsole? Lass das Tier Geräusche machen, gibt es

ebenfalls eine Ausgabe in der Konsole? Erstelle einen Besucher mit dem Namen “Franz” und Alter 35. Lass Franz das Tier streicheln und füttern. Wird jeweils eine Ausgabe in der Konsole gemacht und hat sich das Gewicht des Tier erhöht? Es sollte jetzt in etwa ein Gewicht von 36.0 haben.

Aufgabe 5 - Schaf

Jetzt kreieren wir unser erstes, richtiges Tier. Erstelle eine Klasse **Schaf**, die von der Klasse **Tier** erbt. Das Schaf überschreibt die geerbte Methode zum Geräusche machen. Die Methode gibt jetzt “määäh” in der Konsole aus.

Test:

Erstelle ein Schaf mit dem Namen “Shawn”, dem Gewicht 23.4, dem Alter 5 und dem Geschlecht “Weiblich”. Lass das Schaf Geräusche machen.

Aufgabe 6 - Huhn

Wir erweitern das Sortiment. Erstelle eine Klasse **Huhn**, die von der Klasse **Tier** erbt. Das Huhn hat eine zusätzliche Eigenschaft, **eierProTag** (Int). Das Huhn überschreibt die geerbte Methode zum Geräusche machen. Die Methode gibt jetzt "ga-gack" in der Konsole aus.

Test:

Erstelle ein Huhn mit dem Namen “Kikeri”, dem Gewicht 2.5, dem Alter 2, dem Geschlecht “Männlich” und das 2 Eier pro Tag legt. Lass das Huhn Geräusche machen und die Anzahl der Eier pro Tag ausgeben.

Aufgabe 7 - Kuh

Erstelle eine Klasse **Kuh**, die von der Klasse **Tier** erbt. Die Kuh hat eine zusätzliche Eigenschaft, **gibtMilch** (Boolean). Die Eigenschaft ist wahr, wenn die Kuh mindestens 3 Jahre alt und weiblich ist. Die Eigenschaft wird in einem sekundären Konstruktor festgelegt. Ebenso überschreibt die Kuh die geerbte Methode zum Geräusche machen. Die Methode gibt jetzt “Muuuh” in der Konsole aus. Die Kuh hat zusätzlich noch die Methode melken() mit dem Parameter Besucher. Die Methode gibt in der Konsole aus, ob die Kuh Milch gibt oder nicht.

Test:

Erstelle eine Kuh mit dem Namen “Berta”, dem Gewicht 525.0, dem Alter 7 und dem Geschlecht “Weiblich”. Erstelle einen Besucher mit dem Namen “Franz” und Alter 35. Melke dann die Kuh mit dem Besucher Franz. Gibt Berta Milch?.

Aufgabe 8 - Pony

Erstelle eine Klasse **Pony**, die von der Klasse **Tier** erbt. Das Pony hat eine zusätzliche Eigenschaft, **geschwindigkeit** (Double). Das Pony überschreibt die geerbte Methode zum Geräusche machen. Die Methode gibt jetzt “Wiehiehie” in der Konsole aus. Zusätzlich können Besucher auf dem Pony reiten, aber nur, wenn das Pony mindestens 1 Jahr alt ist und der Besucher mindestens 6 Jahre alt ist. Schreibe dafür eine passende Methode. Die Methode gibt in der Konsole aus, ob der Besucher reiten darf und wenn ja, welcher Besucher auf welchem Pony reitet.

Test:

Erstelle ein Pony mit dem Namen “Gustav”, dem Gewicht 315.0, dem Alter 3, dem Geschlecht “Männlich” und einer Geschwindigkeit von 30.5.
Erstelle einen Besucher mit dem Namen “Franz” und Alter 35 und
einen Besucher mit dem Namen “Sissi” und Alter 4.

Lass Franz und Sissi das Pony reiten. Wird Franz das Pony reiten erlaubt und Sissi nicht?

Aufgabe 9 - Streichelzoo erstellen

Jetzt erstellen wir unseren Streichelzoo. Dazu erstellst du eine Klasse mit dem Namen **Streichelzoo**.

Die Klasse hat eine Eigenschaft, eine MutableList vom Typ Tier. In dieser Liste werden alle Tiere aufbewahrt. Die Liste wird gefüllt mit zwei Schafen, zwei Hühnern, einer Kuh und einem Pony. Du darfst dir beim Erstellen der Tiere eigene Parameter ausdenken, oder du verwendest die Tiere, die du in den Aufgaben bereits erstellt hast.

Der Streichelzoo bietet verschiedene Interaktionen an.

1. Die Tiere freuen sich immer, wenn ein Besucher den Streichelzoo betritt. Dann machen alle Tiere Geräusche. Erstelle dafür eine Methode, die einen Besucher als Parameter erhält. Die Methode gibt eine Nachricht aus, dass der Besucher den Streichelzoo betreten hat. Anschließend machen alle Tiere Geräusche. Verwende dafür eine Schleife.
2. Der Zoowärter füttert alle Tiere. Schreibe eine Methode, in der alle Tiere gefüttert werden. Verwende dafür eine Schleife.
3. Der Zoo bietet ein Rennen zwischen zwei Ponys an. Schreibe eine Methode, bei der zwei Ponys als Parameter übergeben werden und das schnellere der zwei in der Konsole als Gewinner ausgegeben wird.

Aufgabe 10 - Ein Tag im Streichelzoo!

Erstellt eine Datei Tag.kt mit einer main()-Funktion. Zusätzlich erstellst du die zwei Besucher deines Streichelzoos. :

```
var seb : Besucher = Besucher("Seb", 15)
var boris : Besucher = Besucher("Boris", 4)
```

Lass die zwei Besucher den Tagesablauf durchführen:

1. Zunächst stellen sich die beiden Besucher vor.
2. Daraufhin macht eins der Hühner ein Geräusch.
3. Seb füttert das Schaf.
4. Boris will auf einem der Ponys reiten.
5. Eines der Hühner läuft durchs Gehege.
6. Seb streichelt eins der Ponys.
7. Danach reitet Seb auf dem Pony.
8. Jetzt versucht Boris auf dem Pony zu reiten.
9. Boris versucht die Kuh zu melken.
10. Alle Tiere machen zusammen ihre Geräusche.
11. Fütterungszeit. Boris füttert alle Tiere.
12. Zuletzt haben die beiden Ponys ein Rennen.

Bonusaufgaben:

Unser heutiges Projekt sollte euch als Inspiration dienen, selbstständig weiteren Code zum Üben zu programmieren. Statt weiteren Bonusaufgaben gibt es also Vorschläge für weitere Features, die ihr freiwillig noch im Projekt einbauen könnt. Sie sind offen geschrieben, es gibt also viele Lösungsmöglichkeiten. Versuche sie so gut wie möglich umzusetzen.

1) **Futterbeutel**

- Um Tiere zu füttern, brauchen wir natürlich Futter. Erstelle einen Futterbeutel. Der Beutel enthält Futter, angegeben in Gramm. Jeder Besucher hat einen Futterbeutel. Der Besucher kann jetzt das Futter zum Füttern verwenden. Dabei steigt das Gewicht des Tieres, je nachdem wie viel Futter verfüttert wurde. Das verfütterte Futter verschwindet natürlich aus dem Futterbeutel.

- Der Besucher kann den Futterbeutel an einer Kasse wieder auffüllen. Dafür muss der Besucher Geld bezahlen. Erweitere die Besucher-Klasse, sodass ein Besucher nun auch Geld mit sich trägt. Das Startgeld kann zu Beginn zufällig zugewiesen werden.

2) **Fütterung**

- Jedes Tier mag anderes Futter. Erstelle verschiedene Sorten an Futter. Bei der Kasse kann der Besucher sie zu unterschiedlichen Preisen kaufen und im Futterbeutel aufbewahrt werden.

- Die Kasse hat auch ein Infoblatt. In dem Infoblatt steht, welches Tier welches Futter mag. Die Tiere sind jetzt auch wählerisch und fressen kein Futter, das sie nicht mögen.

3) **Patenschaft**

- An der Kasse gibt es die Möglichkeit, eine Patenschaft mit einem Tier einzugehen.
  Im Infoblatt ist aufgelistet, welches Tier wie viel kostet.
  Zusätzlich wird an der Kasse aufgelistet, welches Tier welche Besucher als Paten hat.