ZIEL DER ÜBUNG

Sie wiederholen und üben die Konzepte der Objektorientierten Programmierung anhand kleiner ausgewählter Beispiele. Die Komplexität der Beispiele steigert sich von einer Übung zur nächsten.

- **Übung A** dient für Sie als leichter Einstieg in ein überschaubares kompaktes Beispiel zur Anwendung der OOP.
- **Übung B** wird etwas kniffliger. Sie erstellen ein kleines Programm zur Berechnung von Distanzen zwischen zweier Punkten in einer Ebene, sowie auch als Weiterführung zweier Punkte im dreidimensionalen Raum.
- **Übung C** beschäftigt sich mit Datenstrukturen. Sie bauen in Java einen „Stack" nach. Ein Stack = „Stapel" speichert Werte nach dem FILO Prinzip. Den ersten Wert den Sie hinzufügen, werden Sie als letztes aus diesem Stapelspeicher entfernen.
- **Übung D** schließt mit einem weiteren Beispiel aus Datenstrukturen ab. Sie entwickeln in Java eine „Queue" = Warteschlange, die nach dem FIFO Prinzip Daten speichern und ausgeben kann.

Nach Abschluss der Übungseinheit sollten Sie folgende Punkte für sich selbst abhaken können!

- Sie können ein **einfach und auch weiterführende Angaben analysieren**!
- Sie können **Angaben in der OOP umsetzen!**
- Sie konnten Ihr Wissen betreffend OOP festigen.

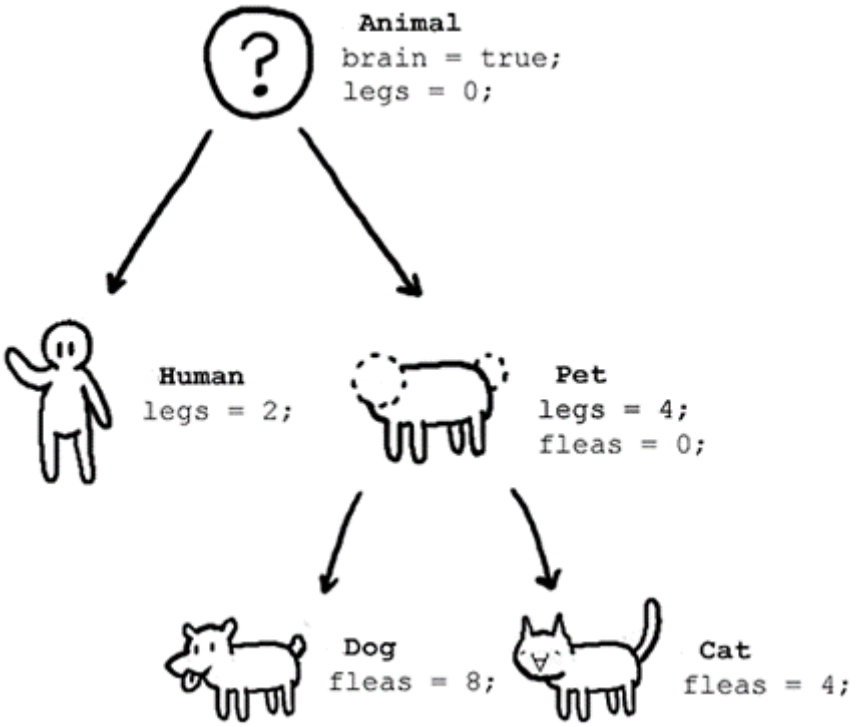
Des Weiteren sind Sie vorbereitet für:

- Die nächsten Übungseinheiten, insbesondere Testen von Source Code.

AUFGABENSTELLUNG

ÜBUNG A – OOP EINSTIEG

- Sie erstellen 5 Java-Klassen laut der angegebenen Grafik (siehe Eclipse exportiertes Projekt)
- Testen Sie Ihre Klassen indem Sie eine weitere Klasse inkl. main-Methode erstellen, mit der Sie Objekte aus den 5 Klassen initialisieren können und verwenden können.
- Setzen Sie Werte der einzelnen Variablen innerhalb der Konstruktoren, z.B. fleas = 4; definieren Sie z.B. den Konstruktor public Animal() { ... } für die Animal-Klasse bzw. public Dog() { ... } für die Dog-Klasse, bzw. überlegen Sie sich wie die Konstruktoren für die Klassen aussehen könnten.
- Erstellen Sie zu jeder Variable einen eigene "Getter"-Methode. Überlegen Sie wie Sie die Getter Methoden Bezeichnungen.
- Erstellen Sie auch die dazugehörigen „Setter“-Methoden.
- Achten Sie auf die Sichtbarkeiten / Scope der Variablen, sodass Variablen auch in abgeleitenden Klassen verwendet werden können
- Achten Sie dabei auf die genauen Inhalte, fragen Sie bei Unklarheiten!
- Halten Sie sich an das bisherige Namensschema aus der ersten Übung, z.B. abgabe02a-ulmmi.zip



ÜBUNG B – DISTANZBERECHNUNG

Dieses Beispiel sollte aus dem Schulunterricht noch bekannt sein. Sie haben ein zweidimensionales Koordinatensystem, bzw. auch ein dreidimensionales Koordinatensystem. In der Ebene bzw. im Raum können Sie Punkte darstellen. Der Abstand zwischen zweier Punkte ist für uns interessant. Um diesen Abstand zu erhalten implementieren Sie zwei Methoden, einerseits der Abstand zweier Punkte und andererseits der Abstand zum Ursprung.

- Sie erstellen 2 Java-Klassen (siehe Eclipse exportiertes Projekt)
- Testen Sie Ihre Klassen indem Sie eine weitere Klasse inkl. main-Methode erstellen, mit der Sie Objekte aus den 2 Klassen initialisieren können und verwenden können, ein Beispiel dazu finden Sie bereits im exportierten Projekt. Versuchen Sie weitere Testfälle zu schreiben und Ihre 2D und 3D Punkte sowie Abstände zu verifizieren.
- Achten Sie auf die Sichtbarkeiten / Scope der Variablen, sodass Variablen auch in abgeleitenden Klassen verwendet werden können
- Achten Sie darauf, dass Sie selbstständig beim 3D Punkt die dritte Koordinate implementieren.
- Achten Sie dabei auf die genauen Inhalte, fragen Sie bei Unklarheiten!
- Halten Sie sich an das bisherige Namensschema aus der ersten Übung, z.B. abgabe02b-ulmmi.zip

ÜBUNG C

Der Stack = Stapel dient zur Speicherung von Daten in der Informationsverarbeitung. Werte werden in diesen Stapel geladen und auch wieder ausgelesen.

Mit diesem Beispiel sollten Sie eine Stapelverarbeitung nachprogrammieren können. Nach dem LIFO Prinzip funktioniert das Konzept des Stacks. Das letzte Element dass Sie diesem "Stapel" / Liste hinzufügen, entnehmen Sie in Folge als erstes Element.

- Sie erstellen eine Stack Implementierung (siehe Eclipse exportiertes Projekt)
- Testen Sie Ihre Klasse indem Sie eine weitere Klasse inkl. main-Methode erstellen, mit der Sie Objekte aus der Stack Klasse initialisieren können und verwenden können.
- Achten Sie dabei auf die genauen Inhalte, fragen Sie bei Unklarheiten!
- Halten Sie sich an das bisherige Namensschema aus der ersten Übung, z.B. abgabe02c-ulmmi.zip

ÜBUNG D

Ein zweites Prinzip der Informationsverarbeitung dient uns als 4. Beispiel in dieser Übungseinheit.

Mit diesem Beispiel sollten Sie eine Warteschlange / Queue nachprogrammieren können. Nach dem FIFO Prinzip funktioniert das Konzept der Queue. Das letzte Element, dass Sie dieser "Warteschlange" / Liste hinzufügen, entnehmen Sie ebenso als letztes Element.

- Sie erstellen eine Queue Implementierung (siehe Eclipse exportiertes Projekt)
- Testen Sie Ihre Klasse indem Sie eine weitere Klasse inkl. main-Methode erstellen, mit der Sie Objekte aus der Stack Klasse initialisieren können und verwenden können.
- Achten Sie dabei auf die genauen Inhalte, fragen Sie bei Unklarheiten!
- Halten Sie sich an das bisherige Namensschema aus der ersten Übung, z.B. abgabe02d-ulmmi.zip

HILFESTELLUNG

Oftmals hilft es, sich die Übungsangaben zu visualisieren. Nehmen Sie einen Stift und Zettel zur Hand und skizzieren Sie Ihr Vorhaben. Welche Schritte sind notwendig? Welche Abläufe müssen Sie in diesen Minianwendungen berücksichtigen? Welche Datentypen / Datenstrukturen werden verwendet?

LITERATUR

WEITERFÜHRENDE LINKS FÜR ÜBUNG B

http://www.ehow.com/how_4461258_find-distance-between-two-points.html

http://www.ehow.com/how_6299829_calculate-between-two-points-3_d.html

http://en.wikipedia.org/wiki/Cartesian_coordinates

WEITERFÜHRENDE LINKS FÜR ÜBUNG C

http://en.wikipedia.org/wiki/LIFO_%28computing%29

http://de.wikipedia.org/wiki/Stapelspeicher

WEITERFÜHRENDE LINKS FÜR ÜBUNG D

http://de.wikipedia.org/wiki/Warteschlange_%28Datenstruktur%29

http://en.wikipedia.org/wiki/FIFO