2 Objektorientierte Modellierung und Programmierung

2.1 Wiederholung der Basics (1 DS)

Klassendefinition, vereinbaren & initialisieren

1. Erstelle eine Klasse `AUTO`. Vereinbare die Attribute `marke`, `baujahr`, `farbe`, `reparaturNoetig`, `kilometerstand` und `tankfuellung` (beides Dezimalwerte). Die Konstruktormethode setzt Marke, Farbe, Baujahr und Kilometerstand auf einzugebende Werte, die Tankfüllung auf 50 und den Zustand auf fahrtüchtig.

2. Erstelle die Klasse `WERKSTATT`. Vereinbare die Attribute `name` und `anzahlMechaniker`. Beide Attribute werden von der Konstruktormethode abgefragt und gespeichert.

3. Erstelle die Klasse `TEST`. Vereinbare hier zwei beliebige Autos `auto1` und `auto2` sowie zwei Werkstätten `ws1` und `ws2` und initialisiere sie in der Konstruktormethode mit Werten die du dir ausdenkst.

Methodenkopf, kommunizieren mit Methoden

4. Schreibe die Methoden `markeGeben()` und `tachostandGeben()`, welche die Marke bzw. den Kilometerstand des Autos zurückgeben.

5. Erstelle die Methode `farbeAendern()`, welche die Farbe eines Autos in den einzugebenden Wert abändert.

6. Schreibe eine Methode `tanken()`, die den Tankinhalt um einen einzugebenden Wert erhöht und am Bildschirm ausgibt: „(Marke) hat (tankfuellung) im Tank.“

7. Erstelle die Methode `reparaturbedarf()`, die das entsprechende Attribut auf einen einzugebenden Wert setzt.

Wiederholungen mit fester Anzahl, Bildschirmausgabe

8. Schreibe eine Methode `tankwarnung()`, die am Bildschirm zehnmal hintereinander eine Warnung ausgibt, dass der Tank bald leer ist.

9. Schreibe eine Methode `werbung()`, die auf Eingabe einer Zahl wartet und entsprechend oft am Bildschirm ausgibt „Kommen Sie zu (Werkstattname)!“.

Bedingte Anweisungen, Zufallszahlen

10. Schreibe eine Methode `autoStabil()`, die prüft, ob eine Reparatur nötig ist oder nicht und jeweils eine passende Meldung „(Marke) …“ ausgibt.

11. Schreibe eine Methode `zufallsdefekt()`, die bei jeder Fahrt eines Autos (Aufgabe 13) aufgerufen wird und abhängig vom Wert einer Zufallszahl das Auto fahruntüchtig macht. Sollte dies passieren, erfolgt eine Meldung am Bildschirm.

12. Schreibe eine Methode `lackieren()`, die in einer der Werkstätten einem bestimmten Auto eine neue, einzugebende Lackfarbe verpasst.

13. Schreibe eine Methode `fahren()`, die den Kilometerstand um einen einzugebenden Wert erhöht und den Tankinhalt um einen Zehntel dieses Wertes verringert, wenn genug Benzin im Tank ist; falls nach der Fahrt weniger als zehn Liter verbleiben, soll die Methode `tankwarnung()` aufgerufen werden. Langt der Tankinhalt nicht, soll eine passende Warnung ausgegeben werden.

Objekte kommunizieren

14. Erstelle eine `testmethode()`, die Folgendes macht: beide Autos fahren eine bestimmte Strecke, ein Auto wird in einer der Werkstätten umlackiert, von einem Auto werden Kilometerstand und Marke am Bildschirm ausgegeben, eine Werkstatt macht Werbung und (nachdem du Aufgabe 17 programmiert hast) die Größe einer Werkstatt wird ausgegeben.

Unterklassen

15. Erstelle die Klasse `ZEITMASCHINE` als Unterklasse. Die Konstruktormethode greift auf die Oberklasse zu; dabei ist die Marke *DeLorean*, das Baujahr *1982* und die Farbe *silber*. In der Klasse gibt es ein zusätzliches Attribut `jahr`, in dem das Jahr gespeichert ist, in welchem man sich befindet. Es wird anfangs auf 1984 gesetzt.

16. Erstelle die Klasse `TAXI` als Unterklasse mit den zusätzlichen Attributen `fahrpreis` mit Anfangswert 0 und `sitzplaetze`, dessen Wert in der Konstruktormethode abgefragt und gespeichert wird. Die Farbe von jedem Taxi ist beige. Verwende wieder den Konstruktor der Oberklasse. Überschreibe die Methode `fahren()`, sodass der Fahrpreis als Hälfte der Strecke berechnet und am Bildschirm ausgegeben wird. Ergänze die Klasse `TEST` um ein Taxi und um eine Fahrt mit diesem Taxi.

17. Erstelle die Klasse `RENNWAGEN` als Unterklasse. Schreibe den Konstruktor so, dass als Baujahr 2025 gespeichert wird und man anfangs 150 Liter im Tank hat. Überschreibe die Methode `fahren()`, sodass sich der Tankinhalt um ein Viertel der einzugebenden Strecke verringert und keine Warnungen ausgegeben werden.

Mehrfachauswahl

18. Schreibe eine Methode `werkstattgroesse()`, die mit einer Case-Anweisung auswertet, ob es eine große (4-5 Mitarbeiter) oder eine mittelgroße (2-3)

Werkstatt ist, oder sogar nur ein Ein-Mann-Laden. Am Bildschirm soll eine passende Meldung ausgegeben werden. Im default-Zweig kannst du unterscheiden zwischen ungültiger Mitarbeiterzahl oder sehr großer Werkstatt.

19. Schreibe eine Methode `taxigroesseGeben()`, die mit einer Case-Anweisung auswertet und als Text zurückgibt, ob es ein Großraumtaxi (5-7 Plätze) ein normales Taxi (3-4) oder ein Tuk Tuk ist (1-2). Im default-Zweig wird eine Fehlermeldung zurückgegeben. Ergänze die Testmethode um eine Bildschirmausgabe, welche Größe das bereits erstellte Taxi hat.

Wiederholung mit Bedingung

20. Erstelle in `ZEITMASCHINE` eine Methode `zeitsprung()`, die auf ein Zieljahr in der Zukunft wartet und alle Jahreszahlen vom aktuellen Jahr bis zum Zieljahr am Bildschirm ausgibt. Verwende auf jeden Fall `while`!

21. Erstelle in `TAXI` eine Methode `zusteigen()`, die auf Eingabe einer Zahl von Fahrgästen wartet. Nun erstellst du ein Attribut `besetzt` mit Anfangswert 0. Solange die besetzten Plätze weniger sind als die Sitzplätze und solange noch Fahrgäste zusteigen wollen erhöhst du schrittweise die besetzten Plätze um 1 und reduzierst die Wartenden ebenfalls um 1. Verwende wieder `while`. Am Ende gibst du aus, wie viele Personen nicht ins Taxi passen. Ergänze einen Aufruf der Methode in der Testmethode.

Zusatzaufgaben

22. Schreibe eine Methode `langeFahrt()`, die ein Auto solange fahren lässt, bis der Tank fast leer ist (< 10 Liter). In jedem Durchlauf wird eine zufällige Strecke zwischen 50 und 100 km fahren und der aktuellen Kilometerstand ausgeben. Am Ende soll die Gesamtstrecke der Fahrt ausgegeben werden.

23. Ergänze die Testmethode, sodass die beiden Rennwägen ein Rennen gegeneinander fahren, indem beide eine zufällig erzeugte Anzahl von Kilometern fahren und anschließend am Bildschirm ausgegeben wird, wer weiter gefahren ist „(Marke) + „ gewinnt“.