Programmieren 1 (PRG1) Übung 6

6.2

Konstruktor:

Eine Methode die zur Definition von Klassenstrukturen in der OOP genutzt wird. Der Konstruktor wird einmal ausgeführt, sobald die Klassenstruktur (Definition der Klasse) zum ersten Mal durchgeführt wird. Mithilfe des Konstruktors können einer Instance der Klasse, bei ihrer Definition, benutzerdefinierte oder festgelegte Werte zugewiesen werden.

Quelle: <https://www.techopedia.com/definition/5656/constructor>; Datum: 30.11.17; Uhrzeit: 21:50

Destruktor:

Eine Methode die Ausgefürht wird, sobald ein Objekt für das sie definiert wurde zerstört wird. Die Methode wird häufig dazu verwendet übrige oder ungewollte „Datenreste“ zu entfernen, Verbindungen zu Servern zu beenden o.ä. („clean up“). Beim programmieren von Klassen, werden Destruktoren dazu verwendet eine bestehende Klassenstruktur aufzulösen.

Quelle: <https://www.techopedia.com/definition/24284/destructor>; Datum: 30.11.17; Uhrzeit: 22:00;

<https://en.wikipedia.org/wiki/Object_lifetime>; Datum: 30.11.17; Uhrzeit: 22:00

public:

Auch andere Klassen können Attribute/Methoden die als „public“ definiert sind, sowohl sehen, als auch modifizieren.

private:

Nur die eigene Klasse kann Änderungen an den Attributen/Methoden der Klasse durchführen.

Protected:

Nur die eigne Klasse und Subklassen von der „geschützten“ Klasse, können die Attribute/Methoden der „protected“ class modifizieren.

Quellen: Vorlesungsfolien: „V12-ObjektorientierteProgrammierung-Klassen“; Datum: 30.11.17; Folien 22,61,62; Vorlesungsskript: „V10-Klassen“; Datum: 30.11.17; Seiten 3-4 („Vorteile der Kapselung“)

<https://stackoverflow.com/questions/1020749/what-are-public-private-and-protected-in-object-oriented-programming>; Datum: 30.11.17; Uhrzeit: 22:30

6.3.1

a)

Output:



Ausgaben in gegebenem Beispiel sind die Attribute vehicel (bike1.vehicle), Kilometerstand (bike1.mileage) und Pferdestärken (bike1.horsepower) der Instance bike1.

Der Parameter vehicle wurde gewählt, um die Art des Fahrzeugs darzustellen. Die Pferdestärke war einfach eine Typische Angabe von Fahrzeuginformationen.

b)

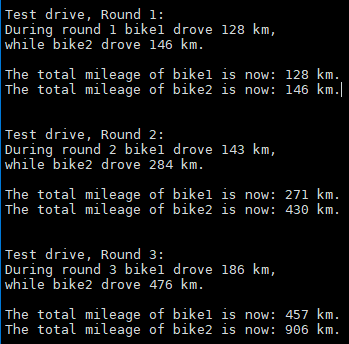
Output 1:



Print der in main() erzeugten Attribute bike1.color und bike2.color. Die Attribute wurden mittels bike1.set\_color(„x“) (bzw. bike2.set set\_color(„x“) ) erzeugt. Die untypischen Farben „Sarcoline“ (heißt so viel wie „Fleischfarben“), bzw. „Smaragdine“ (Smaragdgrün) wurden zur Erheiterung des Lesers gewählt.

**Siehe nächste Seite**

Output 2:



Mittels .randrange() Funktion, wurden in der Funktion test\_drive(bike1, bike2, 3) über drei Runden hinweg jeweils zwei Zahlen zwischen 100 und 500 ermittelst. Das bike.mileage() Attribut jedes Bikes wurde um den entsprechenden Wert angehoben und die Ergebnisse jeder Runde in der Konsole ausgedruckt.

6.3.2

c)

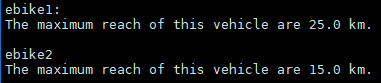
Output:



Mittels Konstruktor wurden die Attribute curr\_capacity und consumption bei der Definition von bike1 festgelegt. Im Anschluss wurde Funktion show\_reach() ausgeführt. Diese berechnet das Verhältnis von Kapazität und Verbrauch und druckt das Ergebnis in der Konsole aus.

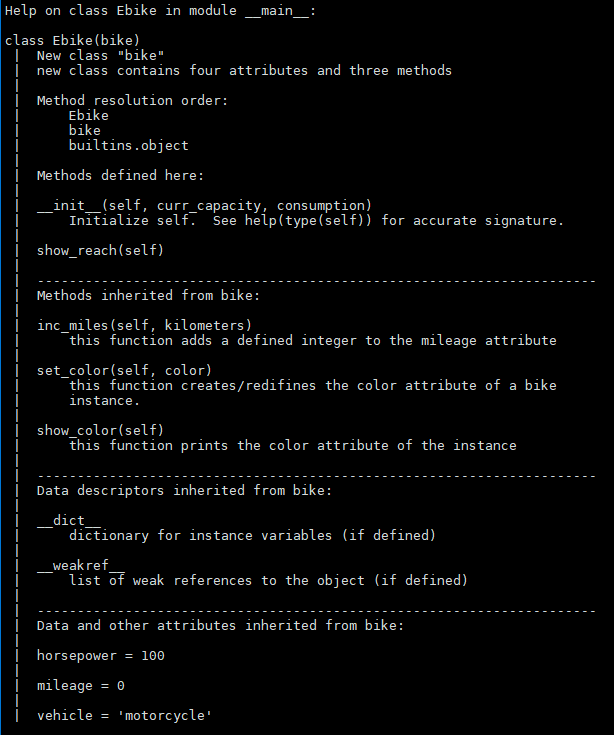
d)

Output:



Die Eingabe erfolgte analog zu Aufgabe c).

Output help(Ebike):



Die Funktion help() zeigt, dass Ebike eine Subklasse von Bike ist. Daher wurden zuerst alle in Bike definierten Funktionen und dann alle Attribute von Bike übernommen.