Lösung 2 - Aggregation implementieren

Diese Aufgabe wird selbständig step by step gelöst. Das Programm ist dann lauffähig, wenn es die geforderten Ausgaben realisiert. Hinweis: Der Code ist im Repo Solution abgelegt (nur einsehbar für Lerhpersonen)

Hier dennoch die Musterlösung für die Klasse `Wheel`:

car.py

```
class Wheel:
    """
    Definiert ein Autorad.
    """
    def __init__(self, size: int, type: str):
        """
        Erzeugt ein Autorad mit der gegebenen Grösse und dem Reifentyp.
        :param size: Die Grösse
        :param type: Der Reifentyp
        """
        self.__size = size
        self.__type = type

    @property
    def size(self) -> int:
        """
        Gibt die gesetzte Radgrösse retour.
        :return: Die Radgrösse
        """
        return self.__size

    @property
    def type(self) -> str:
        """
        Gibt den gesetzten Reifentyp retour.
        :return: Der Reifentyp
        """
        return self.__type

    def print(self) -> None:
        """
        Gibt die Eigenschaften des Rades auf der Konsole aus.
        """
        print(f"Rad: {self.type} / Grösse {self.size} Zoll")


class Engine:
    pass
```

```
class Car:
    pass


if __name__ == '__main__':
    pass
```

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/archiv/m320/learningunits/lu98/loesungen/lu06-aufgabe_2

Last update: **2026/08/30 14:06**