Algorithms and Data Structures for Data Science

Object Oriented Programming

CS 277

December 8, 2021

Brad Solomon

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Department of Computer Science

Learning Objectives

Concluding thoughts about CS 277

By request: A short tutorial on inheritance in OOP

Open office hours!

What is CS 277 about?

Algorithms and Data Structures

Logic and Relations

Runtime Analysis

Searching and Sorting

Algorithm Design

Lists, Stacks, Queues

Trees

Hashing and Hash Tables

Graphs

Data Science

Parsing and Processing

Visualization

Storage and Compression

Analysis

What will you get out of this class?

Understand fundamental algorithms and data structures

Experience applying data structures, algorithms, and algorithm design principles to real world problems

Justify implementation choices based on theoretical or practical considerations

Build a foundation for future data science projects

Objects

A **class** is a blueprint for an object

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def about(self):
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")
```

Object Inheritance

A **derived class** inherits from another class definition

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def about(self):
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")


```

```
class Teacher(Person):
    def __init__(self, name, age, salary):
        super().__init__(name, age)
        self.job = "teacher"
        self.__salary = salary

    def about(self):
        super().about()
        print(f"Job: {self.job}")
        print(f"Salary: {self.__salary}”)

```

Object Inheritance

Derived classes can call their parent derived function with **super()**

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def about(self):
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")
```

```
class Teacher(Person):
    def __init__(self, name, age, salary):
        super().__init__(name, age)
        self.job = "teacher"
        self.__salary = salary

    def about(self):
        super().about()
        print(f"Job: {self.job}")
        print(f"Salary: {self.__salary}”)
```

Object Inheritance

Derived classes can overwrite functions and contain novel functions

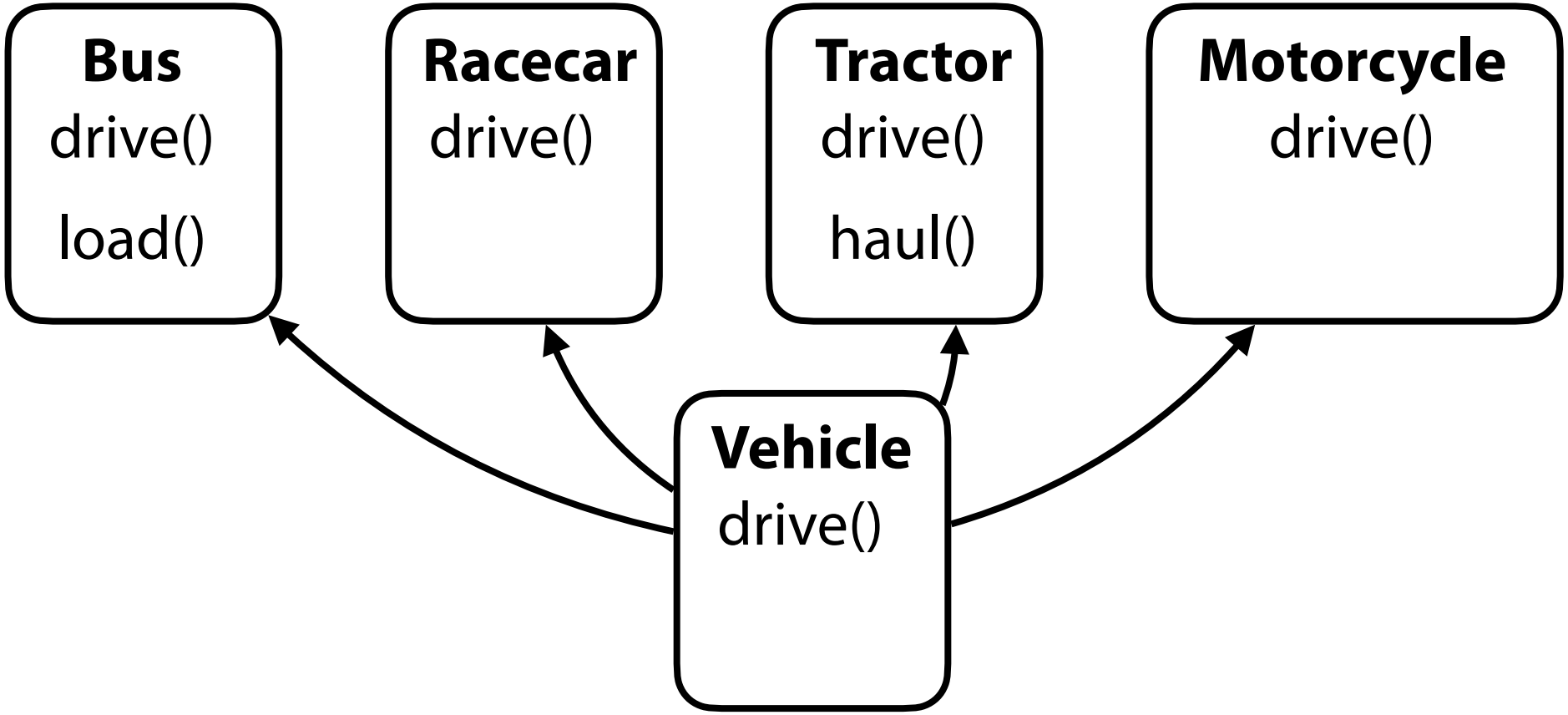
```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def about(self):
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")
```

```
class Teacher(Person):
    def __init__(self, name, age, salary):
        super().__init__(name, age)
        self.job = "teacher"
        self.__salary = salary

    def about(self):
        super().about()
        print(f"Job: {self.job}")
        print(f"Salary: {self.__salary}”)

    def teach(self):
        print("Data structures are cool")
```

Object Polymorphism

Polymorphic functions have shared arguments over many derived classes