CS 209
Data Structures and Mathematical Foundations

02 / 02 / 2024

Instructor: Michael Eckmann

Today's Topics

- Questions/Comments?
- Announcements
- Continue Python review
  - Writing our own classes – the classes we write will be new types

class

- Think of a **class** as containing
  - Data (aka **instance variables**)
  - **Methods** (functions that work on one or more instance variables of one **object** of the class)
- An **object** is an instance of a class.
- Example:

  word = ‘Skidmore’

  another_word = ‘College’
- word and another_word are both objects of the class str.

class

- To allow a class to have objects created of that class type (that is, for a class to be an instantiable class) a special method called the constructor should be defined.

def **__init__**(self)

- Can have more parameters but self is first.
- Instance variables are preceded by **self.** within methods of the class.
- The __init__ method is called automatically when instantiating an object of a class type.
- e.g. __init__ in str class is called when we do:

word = 'Skidmore'

- Or more explicitly:

another_word = str('College')

class

- Let's define a class that represents a Card.
- Then let's create Card objects in a program and use them.
- If there's time, let's start to create a Deck class.
- Things to think about:
  - What data represents a Card? That is, what does a Card have? Answers should become instance variables in the Card class.
  - What methods should exist in Card? That is, what can we do to/with a Card? Answers should become methods in the Card class.
  - Similar questions should be asked about a Deck,

class

- For Card we said each Card object should have a rank and a suit.
  - Instance variables:
    - __rank
    - __suit
- For Deck we said each Deck object should have a list of Cards (52 of them to start) and we can deal, shuffle and cut a Deck.
  - Instance variable: __the_deck # a list of Cards
  - Methods:
    - deal
    - shuffle
    - cut

class

- Instance variables that start with __ (two underscores) are private otherwise they are public.
- The effect of private instance variable are they cannot be referenced via an object. Public instance variables may be referenced via an object.

Special methods / operator overloading

- Some special methods in classes
  - __str__ --- gets called by str(__)
  - __eq__ --- overloads the == operator
  - __gt__
  - __lt__
  - __ge__
  - __le__