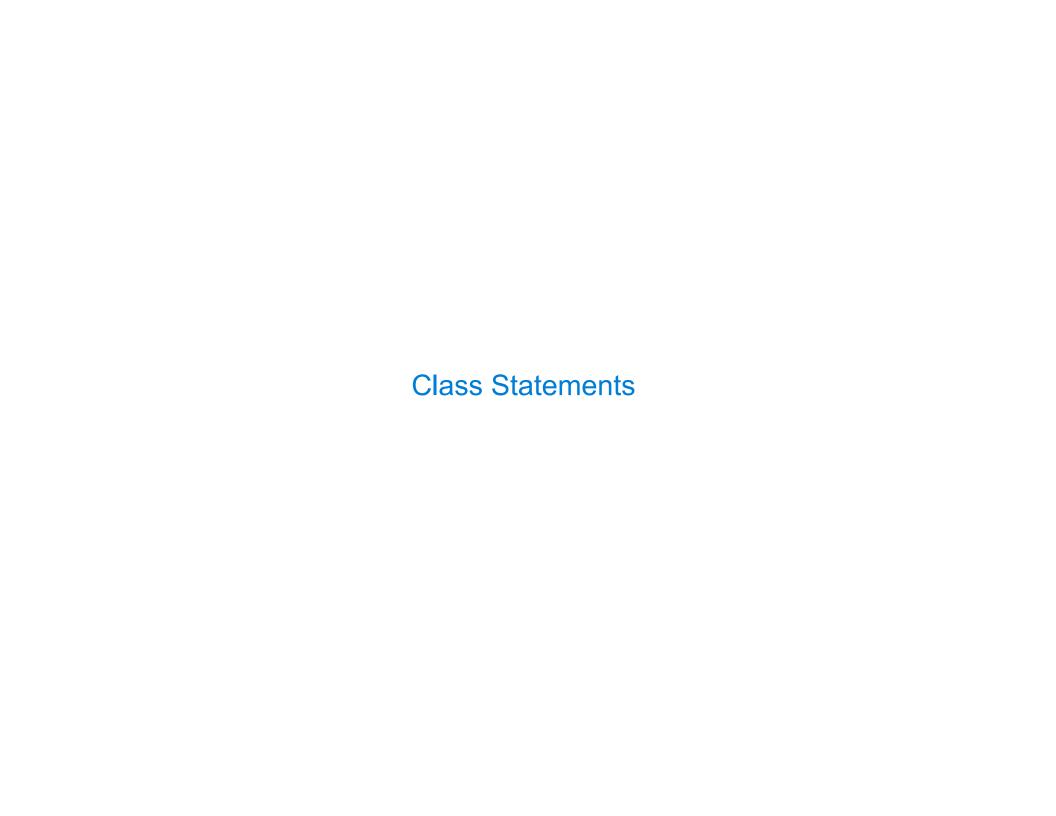




**Review: Aggregation Functions** 

## Project 2 Question 10: Fastest Words

```
def time(match, player_num, word_index):
    """Returns the time it took player_num to type the word at word_index in match."""
def fastest player(match, num players, word index):
    """Return which player typed word index fastest.
   >>> p0 = [2, 2, 3]
   >>> p1 = [6, 1, 2]
    >>> words = ['what', 'great', 'luck']
    >>> [fastest_player(match(words, [p0, p1]), 2, i) for i in range(3)]
    [0, 1, 1]
   fastest p, fastest time = None, None
   for p in range(num players):
       p time = time(match, p, word index)
       if fastest time is None or fastest time > p time:
           fastest p, fastest time = p, p time
   return fastest p
   return min(range(num_players), key=lambda p: time(match, p, word_index))
```



## Classes

## A class describes the behavior of its instances

**Idea:** All bank accounts have a balance and an account holder; the Account class should add those attributes to each newly created instance

Idea: All bank accounts share a withdraw
method and a deposit method

```
>>> a = Account('John')
>>> a.holder
                       balance and
'John'
                        holder are
>>> a.balance
                        attributes
                       deposit and
>>> a.deposit(15)
                       withdraw are
15
                         methods
>>> a.withdraw(10)
>>> a.balance
>>> a.withdraw(10)
'Insufficient funds'
```

6

## The Account Class

```
class Account:
   _init__ is a special method name for the function that constructs an Account instance
         def __init__(self, account_holder):
             self_balance = 0
             self.holder = account holder
 self is the instance of the Account class on which deposit was invoked: a.deposit(10)
         def deposit(self, amount):
                                                                       >>> a = Account('John')
             self.balance = self.balance + amount
                                                                       >>> a.holder
             return self.balance
                                                                       'John'
         def withdraw(self, amount):
                                                                       >>> a.balance
             if amount > self.balance:
                  return 'Insufficient funds'
                                                                       >>> a.deposit(15)
             self.balance = self.balance - amount
              return self.balance
                                                                       >>> a.withdraw(10)
                                                                       >>> a.balance
Methods are functions defined in a class statement
                                                                       >>> a.withdraw(10)
                                          (Demo)
                                                                       'Insufficient funds'
```

Animation Demos by Hany Farid