Sets in Python

- You can cast data to set using the set(...) function
- By default, empty curly braces denote a dictionary: {}
- But if you put items inside them like a list, then it becomes a set:

  ```
  {1, 2, 3, 2, 3, 1} # this is a set containing only 1, 2, 3
  ```

- Convenient way to remove duplicates from data:

  ```
  myList = [1, 2, 3, 2, 3, 4, 5, 4, 3, 6, 7]
  myListNoDupes = list(set(myList))
  ```

Hashability

- In Python, set keys need to be "hashable"
- Generally, must be immutable:
  - Lists are mutable (can be modified), so cannot be keys in a set or dictionary
  - Strings, integers, floats, bools are immutable, so can be used
- If something is mutable, then when passed into a hash function it may have a different result. Keys cannot change their values - violates expectation of hash table and may not be able to find keys again later
- Example: if A = [1, 2] were allowed to be a key, A.append(3) could make it impossible to look up the elements assigned to A using A's hash, because when hashed, [1, 2, 3] could point to a different bucket from [1, 2]