Collections

**Python built-in collections:**

list [,] – generic mutable array type.

tuple (,) – immutable array type.

dict {a:b, …} – lookup-table-like type.

* .items() – return a list of (key, value) pairs
* .keys() – return list of keys.
* .values() – return list of values

set {,} – unordered, mutable, no duplicates, act like sets in math. Useful in niche situations.

* Useful for removing duplicates by converting a list to a set and back.
* Use set() to initialize an empty set, as {} returns an empty dict.
* Standard set operations work in python
  + Proper and improper sub/superset comparisons: <, <=, >, >=
  + Union, intersection, difference, symmetric difference: |, &, -, ^
    - These operations support |=, &=, -=, and ^=
  + .isdisjoint(other) – whether the set has no elements in common with other.
  + .add(elem) and .discard(elem) to add/remove elements

**Collections module (only useful parts):**

deque – double ended queue with fast appending on both the right and left side.

* .rotate(n) – rotate n times to the right. Negative n to go left.
* .reverse() – reverse order in place.

Counter – similar to using a dict to count elements, but elements don’t need to be initialized to begin counting.

* Note that the C is capital.
* Initialize from a list or string to quickly count everything.
* .elements() – return a list of all the elements, repeated as many times as their count.
* .most\_common([n]) – return list of n most common counts as tuples. Not setting n will return all values.
* .total() – sum of all counts