Make an empty set: S1 = set()

Make a populated set: S1 = {3, 5, 7}

Set operations that return sets

logical or .union(s2, s3...) s1 | s2 | ...

logical and .intersection(s2, s3, ...) s1 & s2 & ...

difference .difference(s2, s3, ...) s1 – s2 – s3 ...

logical xor .symmetric\_difference(s2) s1 ^ s2 ^ s3 ...

(only accepts one arg) (multiple sets okay)

Set operations that checks (and returns bools)

... if s1 and s2’s intersection is empty .isdisjoint(s2)

... if every element of s1 is in s2 .issubset(s2) s1 <= s2

... if s1 is a proper subset of s2 s1 < s2

... if every element of s2 is in s1 .issuperset(s2) s1 >= s2

... if s2 is a proper subset of s1 s1 > s2

Set operations

S1 = {1, 2, 3, 6}

add 5: S1.add(5)

merge nondupes from my\_list into S1: S1.update(my\_list)

remove 3: S1.remove(3)

remove 7: will cause error!

remove 7 but not create error: S1.discard(7)

clears all entries S1.clear()

Set Element properties:

unordered, unindexed, unique values only, can’t modify values