Python Set Methods:

Sure, here are some more methods for sets in Python:

10. `update()`: Adds all elements from another set (or any iterable) to the current set.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set1.update(set2)
print(set1) # Output: {1, 2, 3, 4, 5}
```

11. `intersection_update()`: Modifies the current set to contain only the elements that are also in another set.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set1.intersection_update(set2)
print(set1) # Output: {3}
```

12. `difference_update()`: Modifies the current set to remove the elements that are also in another set.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set1.difference_update(set2)
print(set1) # Output: {1, 2}
```

13. `symmetric_difference_update()`: Modifies the current set to contain only the elements that are in either the current set or another set, but not in both.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set1.symmetric_difference_update(set2)
print(set1) # Output: {1, 2, 4, 5}
```

14. `issubset()`: Returns `True` if all elements of the current set are also in another set.

```
set1 = {1, 2, 3}
set2 = {1, 2, 3, 4, 5}
subset_check = set1.issubset(set2)
print(subset_check) # Output: True
```

15. `issuperset()`: Returns `True` if all elements of another set are also in the current set.

```
set1 = {1, 2, 3}
set2 = {1, 2, 3, 4, 5}
superset_check = set2.issuperset(set1)
print(superset_check) # Output: True
```

16. `copy()`: Returns a new copy of the current set.

```
set1 = {1, 2, 3}
set2 = set1.copy()
print(set2) # Output: {1, 2, 3}
```

17. `isdisjoint()`: Returns `True` if the current set has no common elements with another set.

```
set1 = {1, 2, 3}
set2 = {4, 5, 6}
disjoint_check = set1.isdisjoint(set2)
print(disjoint_check) # Output: True
```

18. `pop()`: Removes and returns an arbitrary element from the current set.

```
set1 = {1, 2, 3}
x = set1.pop()
print(x) # Output: 1
print(set1) # Output: {2, 3}
```

19. `clear()`: Removes all elements from the current set.

```
set1 = {1, 2, 3}
set1.clear()
print(set1) # Output: set()
```

20. `union()`: Returns a new set containing all elements from the current set and another set.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set3 = set1.union(set2)
print(set3) # Output: {1, 2, 3, 4, 5}
```

21. `intersection()`: Returns a new set containing only the elements that are also in another set.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set3 = set1.intersection(set2)
print(set3) # Output: {3}
```

22. `difference()`: Returns a new set containing the elements that are in the current set but not in another set.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set3 = set1.difference(set2)
print(set3) # Output: {1, 2}
```

23. `symmetric_difference()`: Returns a new set containing only the elements that are in either the current set or another set, but not in both.

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
set3 = set1.symmetric_difference(set2)
print(set3) # Output: {1, 2, 4, 5}
```

```

These are some more useful methods for sets in Python that can be used for various set operations.
```