Set Methods:

1) add(value): Adds an element to the set. Returns true if the element was added (not already present).

Ex: void main()

```
{
  var s = {1, 2, 3};
  s.add(4); Print(s); // s becomes {1, 2, 3, 4}
}
```

2) addAll(iterable): Adds all elements from an iterable to the set.

Ex: void main()

```
{
    var s = {1, 2};
    s.addAll([3, 4]);
    print(s); // {1, 2, 3, 4}
}
```

3) clear(): Removes all elements from the set.

Ex: void main()

```
{
    var s = {1, 2, 3};
    s.clear(); print(s); // {}
}
```

4) remove(value): Removes an element from the set. Returns true if the element was present.

Ex: void main()

```
{
    var s = {1, 2, 3};
   s.remove(2);
   print(s); // {1, 3}
```

```
}
```

5) removeAll(iterable): Removes all elements that are present in the given iterable.

Ex:
```
void main()
 {
    var s = {1, 2, 3, 4};
    s.removeAll([2, 4]);
    print(s); // {1, 3}
 }
```

6) retainAll(iterable): Keeps only elements that are in the given iterable (intersection in place).

Ex:
```
void main()
{
    var s = {1, 2, 3, 4};
    s.retainAll([2, 3]);
    print(s); // {2, 3}
}
```

7) Map: Applies a function to each element in the Set. Returns an Iterable of the results. The output is not a Set automatically, but an Iterable.

Ex:
```
void main()
 {
    Set<int> num = {1, 2, 3};
    var sq = num.map((n) => n * n);
    print(sq); // (1, 4, 9) — Iterable, not Set yet
    var sqSet = sq.toSet();
    print(sqSet); // {1, 4, 9}
}
```

8) where **:** Filters elements based on a condition. Returns an Iterable of elements that satisfy the condition.

Ex:
```
void main()
```

```
{
    Set<int> num = {1, 2, 3, 4};
    var evenNum = num.where((n) => n.isEven);
    print(evenNum); // (2, 4) — Iterable
    var evenSet = evenNum.toSet();
    print(evenSet); // {2, 4}
}
```

9) Expand: Takes each element and expands it into zero or more elements. Returns an Iterable of the expanded elements.

Example: from a set of strings, expand each string into its characters.

```
Ex: void main()
    {
      Set<String> words = {'Hi', 'Ok'};
      var letters = words.expand((word) => word.split(''));
      print(letters); // (H, i, O, k)
      var letterSet = letters.toSet();
      print(letterSet); // {H, i, O, k}
    }
```

10) simple: Returns the only element in the set if it contains exactly one element.

```
Ex: void main()
    {
      Set<int> singleSet = {42};
      print(singleSet.single); // 42
    }
```