Swift Sets

A set is a collection of unique data. That is, elements of a set cannot be duplicate. For example,

Suppose we want to store information about **student IDs**. Since **student IDs** cannot be duplicate, we can use a set.

Set of Student ID

```
// create a set of integer type
var studentID : Set = [112, 114, 116, 118, 115]
```

Add Elements to a Set

We use the `insert()` method to add the specified element to a set. For example,

```
var numbers: Set = [21, 34, 54, 12]
// using insert method
numbers.insert(32)
```

adds to end uf numbers set

```
[21,34,54,12,32]
```

Create an Empty Set

In Swift, we can also create an empty set. For example,

```
var emptySet = Set<Int>()

print("Set:", emptySet)
```

Remove an Element from a Set

We use the `remove()` method to remove the specified element from a set. For example,

```
var languages: Set = ["Swift", "Java", "Python"]


print("Initial Set: \(languages)")


// remove Java from a set

let removedValue = languages.remove("Java")


print("Set after remove(): \(languages)")
```

OUTPUT:

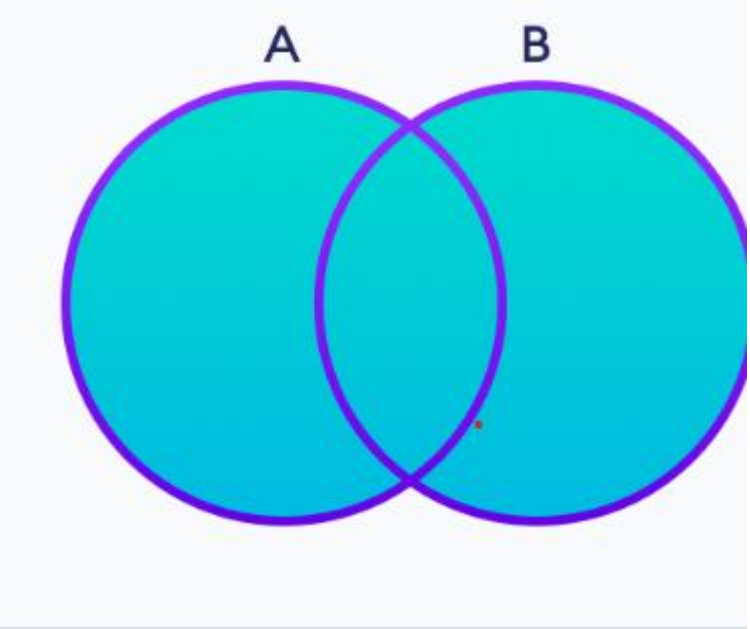
Initial set was : ["Swift", "Java", "Python"]

After remove: = ["Swift", "Python"]

Similarly, we can also use

- `removeFirst()` - to remove the first element of a set
- `removeAll()` - to remove all elements of a set

1. Union of Two Sets



We use the `union()` method to perform the set union operation. For example,

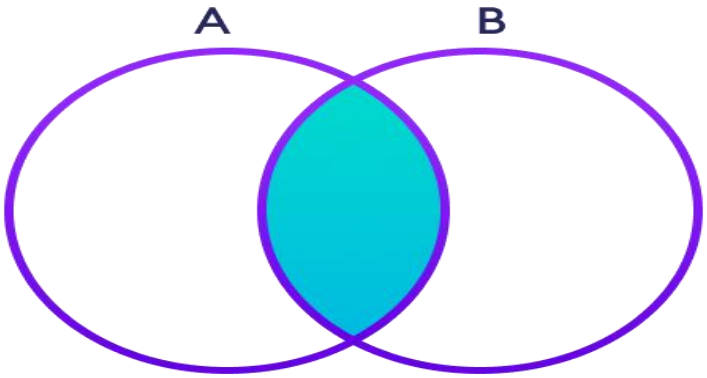
```
// first set
let setA: Set = [1, 3, 5]


// second set
let setB: Set = [0, 2, 4]


// perform union operation
print("Union: ", setA.union(setB))
```

output:

Union: [0,5,2,4,1,3]

2. Intersection between Two Sets

The intersection of two sets **A** and **B** include the common elements between set **A** and **B**.

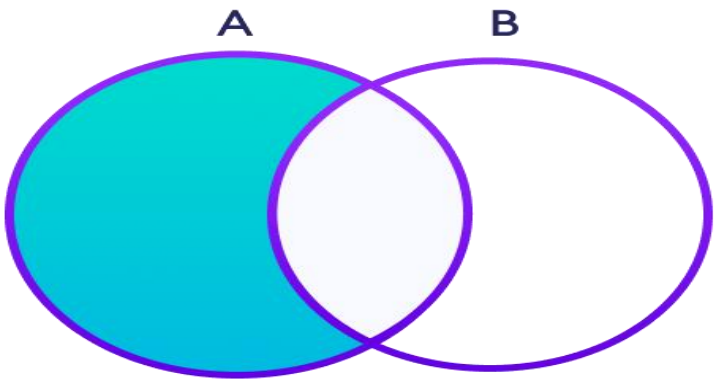



```
// first set
let setA: Set = [1, 3, 5]
print("Set A: ",  setA)

// second set
let setB: Set = [1, 2, 3]
print("Set B: ",  setB)

// perform intersection operation
print("Intersection: ", setA.intersection(setB))
```

```
Intersection:  [3, 1]
```

3. Difference between Two Sets

The difference between two sets **A** and **B** include elements of set **A** that are not present on set **B**.



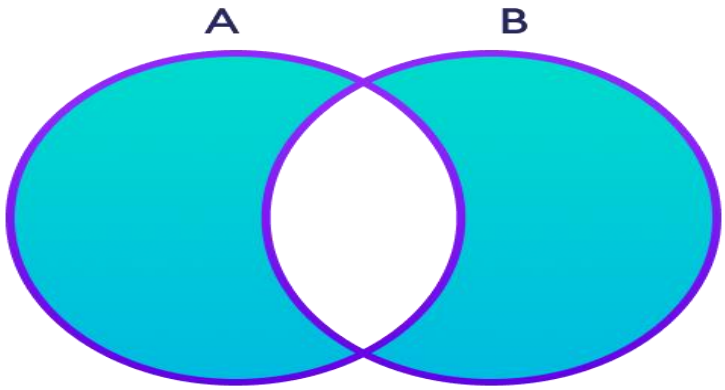
We use the `subtracting()` method to perform the difference between two sets. For example,

```
// first set
let setA: Set = [2, 3, 5]
print("Set A: ",  setA)


// second set
let setB: Set = [1, 2, 6]
print("Set B: ",  setB)


// perform subtraction operation
print("Subtraction: ", setA.subtracting(setB))
```

```
Set A:  [3, 5, 2]
Set B:  [1, 6, 2]
Subtraction:  [3, 5]
```

4. Symmetric Difference between Two Sets

The symmetric difference between two sets **A** and **B** includes all elements of **A** and **B** without the common elements.



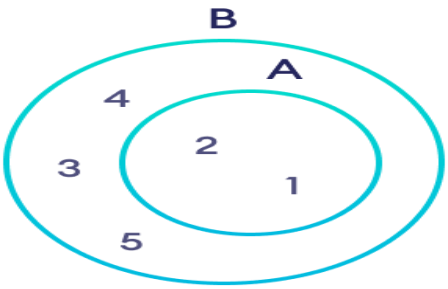
We use the `symmetricDifference()` method to perform symmetric difference between two sets. For example,

```
// first set

let setA: Set = [2, 3, 5]

print("Set A: ", setA)


// second set

let setB: Set = [1, 2, 6]

print("Set B: ", setB)


// perform symmetric difference operation

print("Symmetric Difference: ", setA.symmetricDifference(setB))
```

```
Set A:  [5, 2, 3]
Set B:  [2, 6, 1]
Symmetric Difference:  [1, 6, 3, 5]
```

5. Check Subset of a Set

Set **B** is said to be the subset of set **A** if all elements of **B** are also present in **A**.



We use the `Subset()` method to check if one set is a subset of another or not. For example,

```
// first set
let setA: Set = [1, 2, 3, 5, 4]
print("Set A: ", setA)

// second set
let setB: Set = [1, 2]
print("Set B: ", setB)

// check if setB is subset of setA or not
print("Subset: ", setB.isSubset(of: setA))
```

```
Set A:  [3, 1, 2, 5]
Set B:  [1, 2]
Subset:  true
```