Create an array:

```
num_array = [1, 2, 3, 4, 5]
str_array = ["This", "is", "a", "small", "array"]
```

An array can also be created by calling the **Array.new method**. When you call this method, include up to 2 optional arguments (initial size and default value):

```
Array.new               #=> []
Array.new(3)            #=> [nil, nil, nil]
Array.new(3, "Hello")   #=> ["Hello", "Hello", "Hello"]
```

Accessing Elements

Ruby also allows the use of negative indices, which return elements starting from the *end* of an array, starting at [-1].

```
str_array = ["This", "is", "a", "small", "array"]
str_array[0]            #=> "This"
str_array[-1]           #=> "array"
```

Ruby provides the **#first** and **#last array methods**

these methods can take an integer argument, e.g., `myArray.first(n)` or `myArray.last(n)`, which will return a new array that contains the first or last `n` elements of `myArray`, respectively.

```
str_array = ["This", "is", "a", "small", "array"]
str_array.first         #=> "This"
str_array.first(2)      #=> ["This", "is"]
str_array.last(2)       #=> ["small", "array"]
```

Adding and Removing Elements

Adding an element to an existing array is as simple as using the #**push method** or the shovel operator **<<.** Both methods will add elements to the end of an array and return that array with the new elements. The #**pop method** will remove the element at the end of an array and return the element that was removed.

```
num_array = [1, 2]

num_array.push(3, 4)      #=> [1, 2, 3, 4]
num_array << 5            #=> [1, 2, 3, 4, 5]
num_array.pop             #=> 5
num_array                 #=> [1, 2, 3, 4]
```

The **methods #shift and #unshift** are used to add and remove elements at the beginning of an array. The #unshift method adds elements to the beginning of an array and returns that array (much like #**push**). The #shift method removes the first element of an array and returns that element (much like #**pop**).

```
num_array = [2, 3, 4]

num_array.unshift(1)      #=> [1, 2, 3, 4]
num_array.shift           #=> 1
num_array                 #=> [2, 3, 4]
```

It's also useful to know that both `#pop` and `#shift` can take integer arguments:

```
num_array = [1, 2, 3, 4, 5, 6]

num_array.pop(3)          #=> [4, 5, 6]
num_array.shift(2)        #=> [1, 2]
num_array                 #=> [3]
```

Adding and Subtracting Arrays

```
a = [1, 2, 3]
b = [3, 4, 5]

a + b         #=> [1, 2, 3, 3, 4, 5]
a.concat(b)   #=> [1, 2, 3, 3, 4, 5]
```

To find the difference between two arrays, you can subtract them using -. This method returns a copy of the first array, removing any elements that appear in the second array.

```
[1, 1, 1, 2, 2, 3, 4] - [1, 4]  #=> [2, 2, 3]
```

Basic Methods

```
num_array.methods       #=> A very long list of methods


[].empty?               #=> true

[[]].empty?             #=> false
[1, 2].empty?           #=> false

[1, 2, 3].length        #=> 3

[1, 2, 3].reverse       #=> [3, 2, 1]

[1, 2, 3].include?(3)   #=> true
[1, 2, 3].include?("3") #=> false

[1, 2, 3].join          #=> "123"
[1, 2, 3].join("-")     #=> "1-2-3"
```