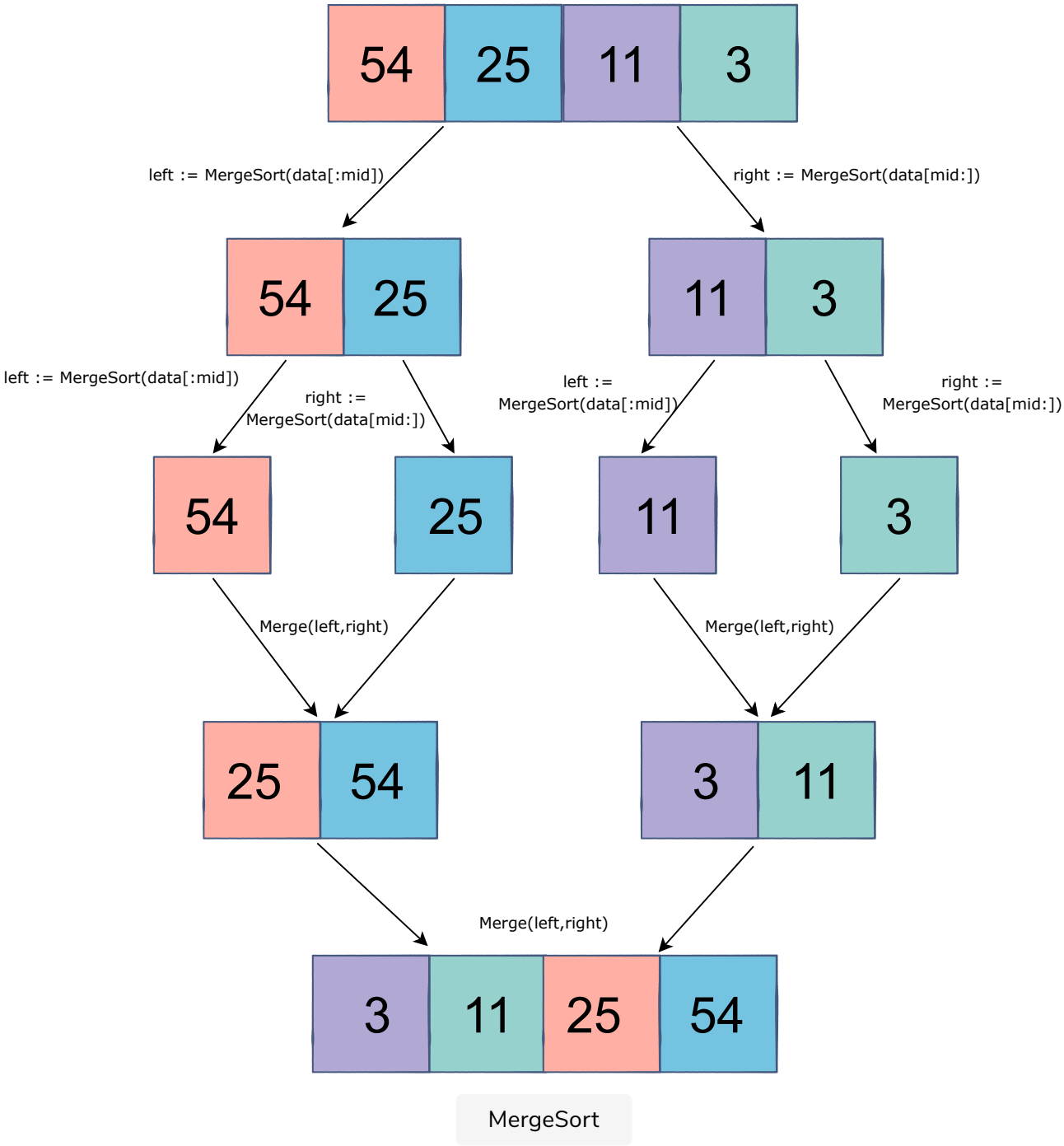
Exercise: Merge Sort

Let's dive into concurrent programming by implementing the merge sort concurrently.

In this exercise, you are required to write a concurrent solution to the Merge Sort problem using `goroutines` and `channels`.

The gist of the Merge Sort Algorithm can be found below :

```
func MergeSort(data [] int) [] int {
  if len(data) <= 1 {
    return data
  }

  mid := len(data)/2
  left := MergeSort(data[:mid])
  right := MergeSort(data[mid:])
  return Merge(left,right)
}
```



MergeSort

Think about the steps that can be executed independently and be synchronized to get the correct final outcome.

The MergeSort has already been implemented sequentially:

```
package main
import "fmt"


func Merge(left, right [] int) [] int{
  merged := make([] int, 0, len(left) + len(right))
  for len(left) > 0 || len(right) > 0{
    if len(left) == 0 {
      return append(merged,right...)
    }else if len(right) == 0 {
```

```
      return append(merged,left...)
    }else if left[0] < right[0] {
      merged = append(merged, left[0])

      left = left[1:]
    }else{
      merged = append(merged, right [0])
      right = right[1:]
    }
  }
  return merged
}

func MergeSort(data [] int) [] int {
  if len(data) <= 1 {
    return data
  }

  mid := len(data)/2
  left := MergeSort(data[:mid])
  right := MergeSort(data[mid:])
  return Merge(left,right)

}

func main(){
  data := [] int{9,4,3,6,1,2,10,5,7,8}
  fmt.Printf("%v\n%v\n", data, MergeSort(data))
}
```

MergeSort

You can code your concurrent solution below:

Current Output:

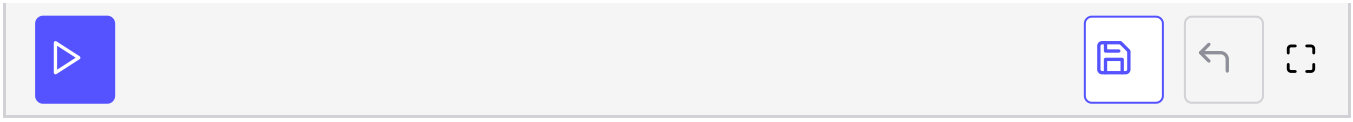
```
[9,4,3,6,1,2,10,5,7,8]
```

Expected Output:

```
[1 2 3 4 5 6 7 8 9 10]
```

```
package main
import "fmt"


func Merge(left, right [] int) [] int{
  merged := make([] int, 0, len(left) + len(right))
  for len(left) > 0 || len(right) > 0{
    if len(left) == 0 {
      return append(merged,right...)
```

```
    }else if len(right) == 0 {
      return append(merged,left...)
    }else if left[0] < right[0] {
      merged = append(merged, left[0])
      left = left[1:]
    }else{
      merged = append(merged, right [0])
      right = right[1:]
    }
  }
  return merged
}

func MergeSort(data [] int) [] int {

  return data
}

func main(){
  data := [] int{9,4,3,6,1,2,10,5,7,8}
  fmt.Printf("%v\n%v\n", data, MergeSort(data))
}
```



Do try it once or twice or as much as you can before moving on to the solution review in the next lesson.