Understanding Precisely How Scan, Fold, and Reduce Work

Vitthal Srinivasan
CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

Recognise the importance of higher order methods

Apply higher order methods that operate on one element at a time

Understand higher order methods that operate on multiple elements

Higher Order Methods

Higher Order Methods

Apply function objects to contents of collection

For-loops

Iterate over collection, freely access contents

Higher Order Methods

Map, Foreach, Filter

Act on one element at a time

Scan, Fold, Reduce

Act on multiple elements at a time

For-each Style Higher Order Methods

Demo

Higher order methods

- foreach
- map
- filter
- partition
- sortBy

Scan-style Higher Order Methods

Scan, Fold and Reduce

Scan Right	Fold Right	Reduce Right
Scan Left	Fold Left	Reduce Left

Scan, Fold and Reduce

Scan Right	**Fold Right**	**Reduce Right**
Scan Left	**Fold Left**	**Reduce Left**

Demo

Scan Right

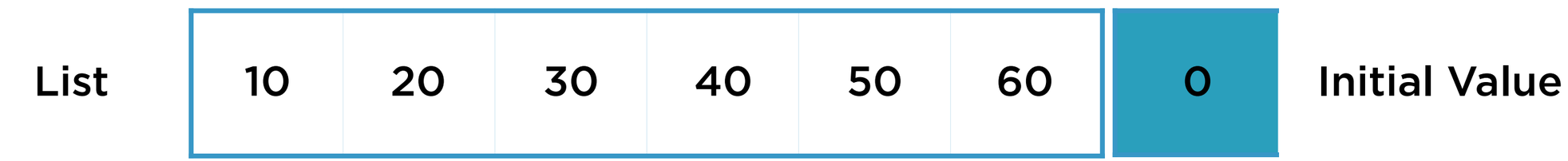
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight

A right-associative higher-order method that takes in an initial value and applies it pairwise, returning a list

```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

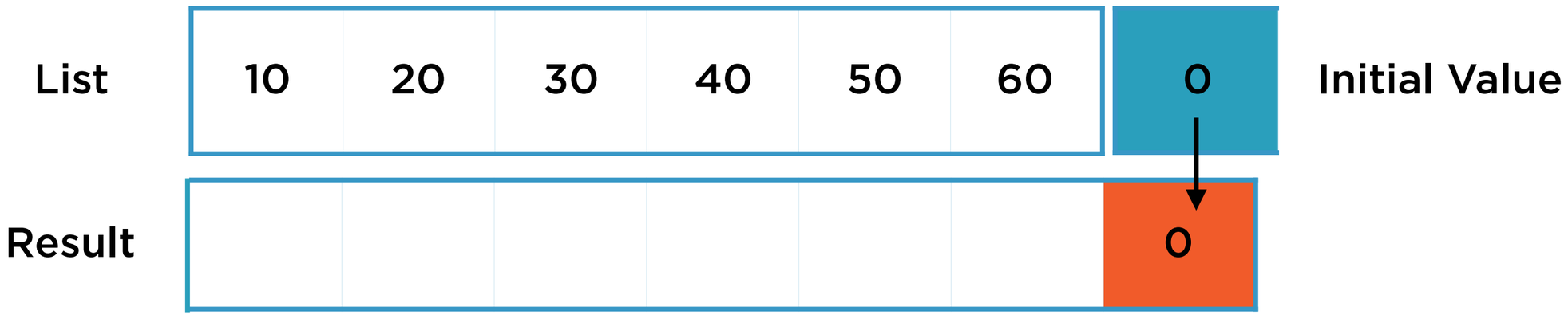
ScanRight



Function `_ - _`

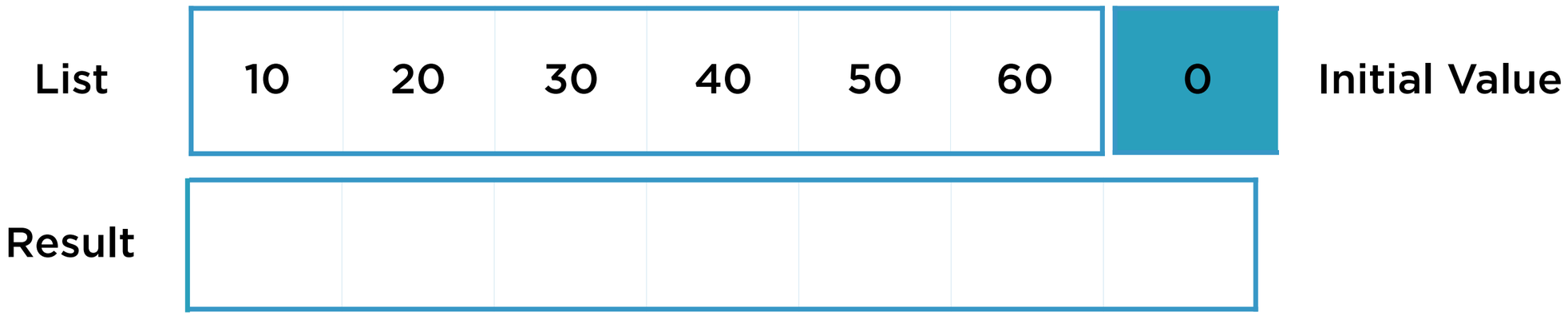
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



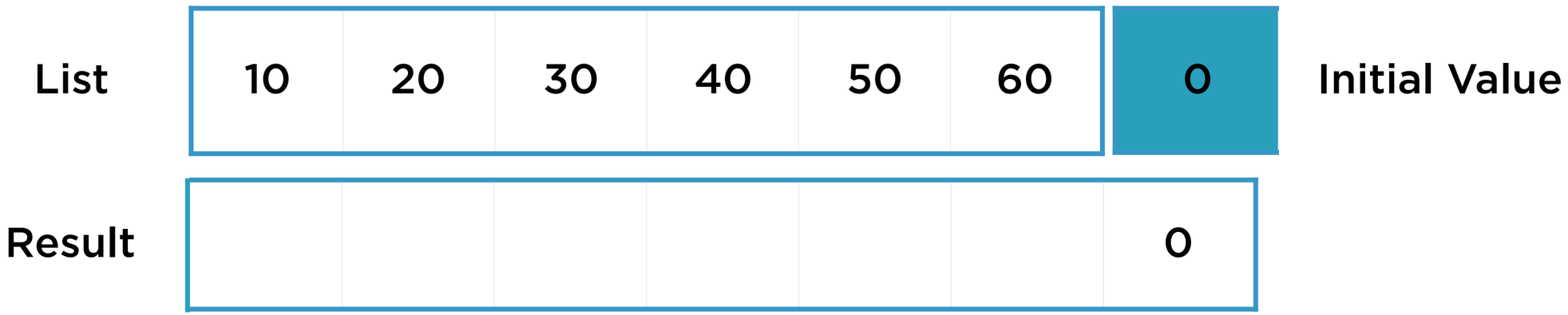
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



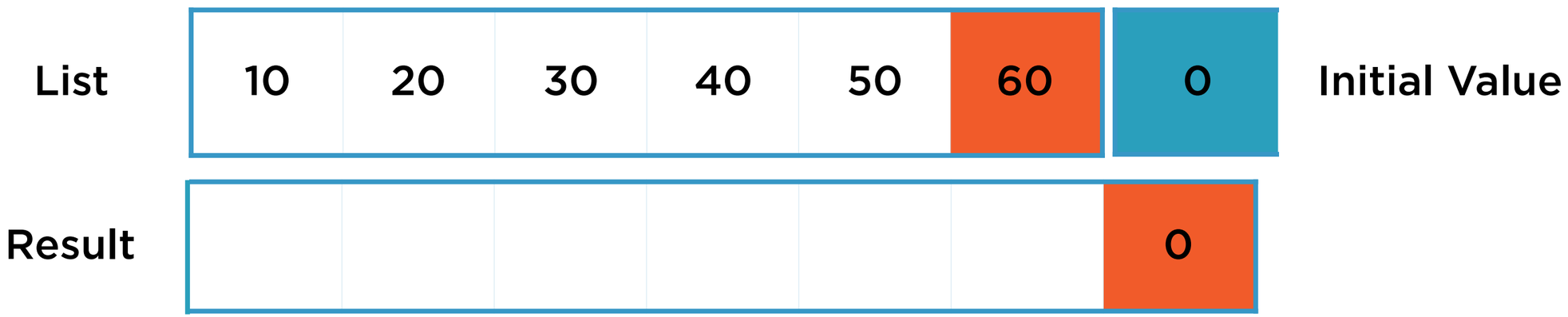
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



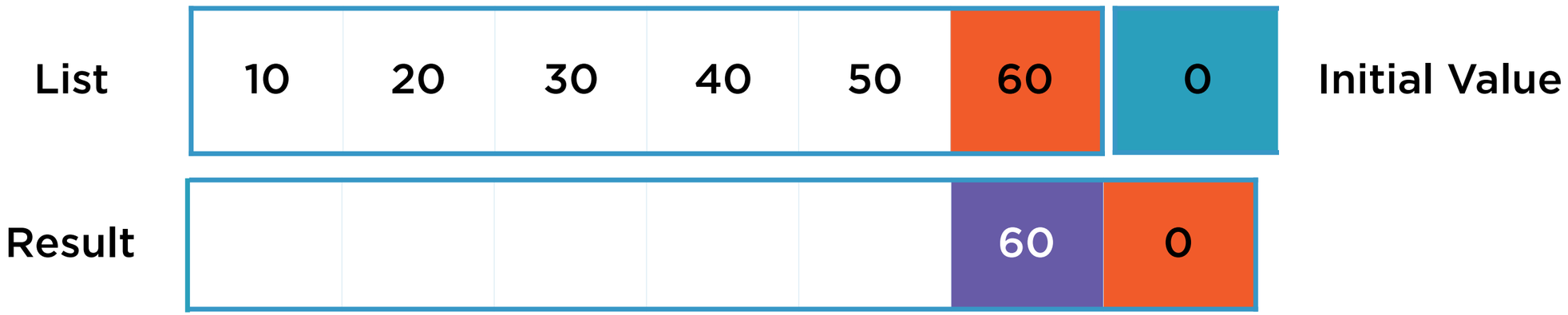
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



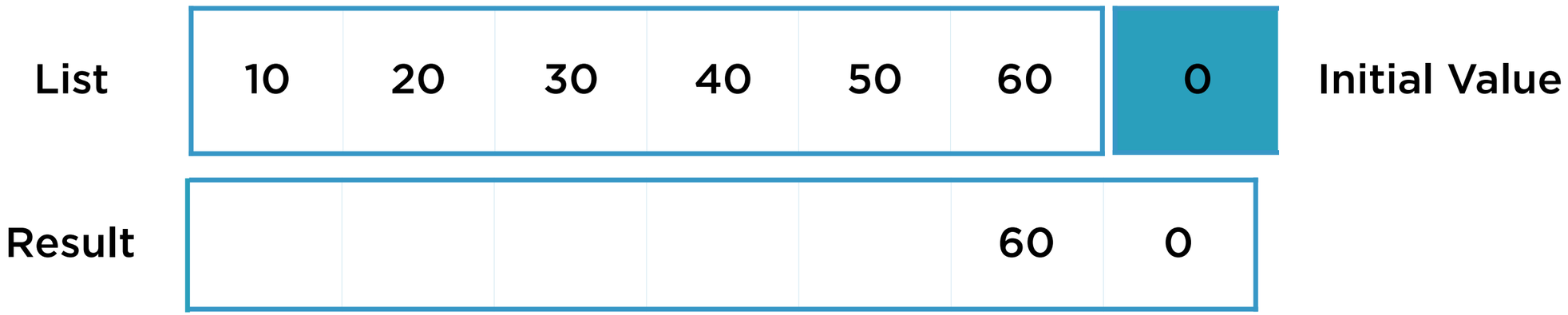
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



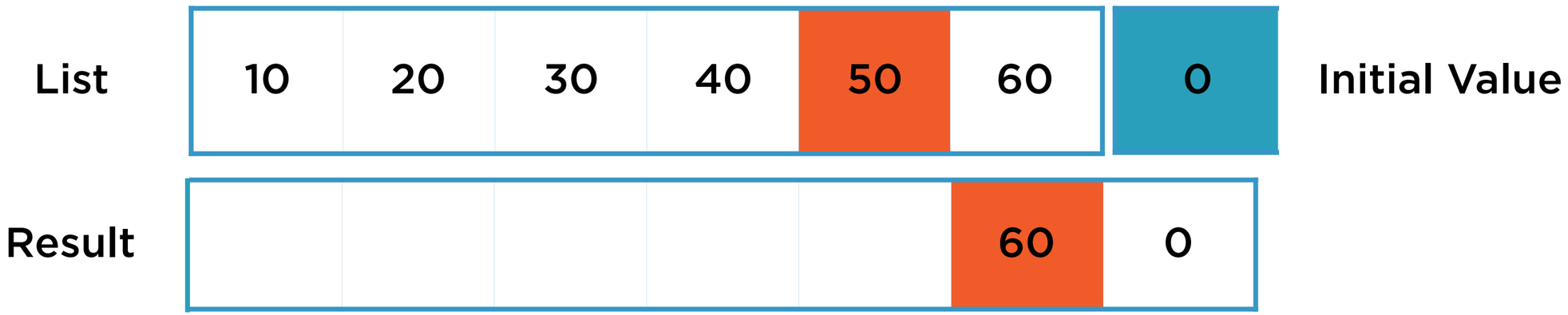
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



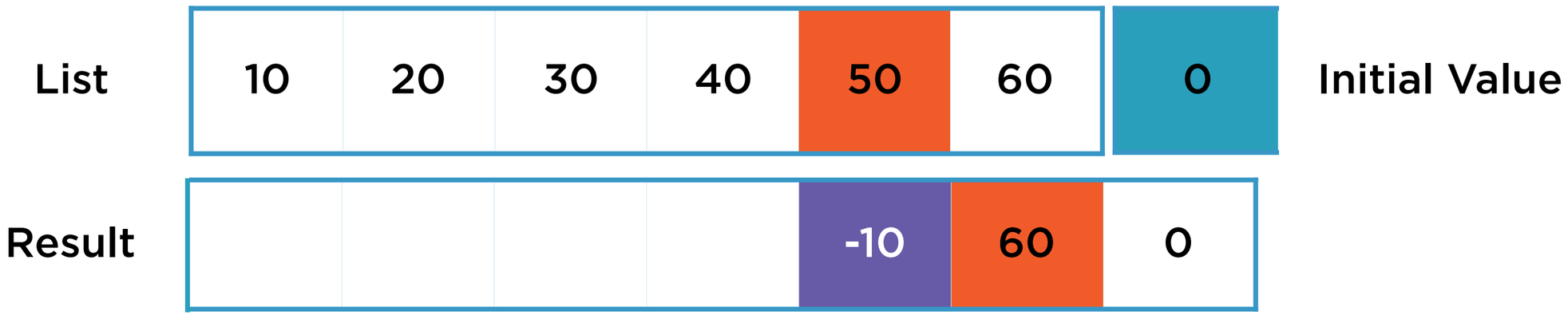
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



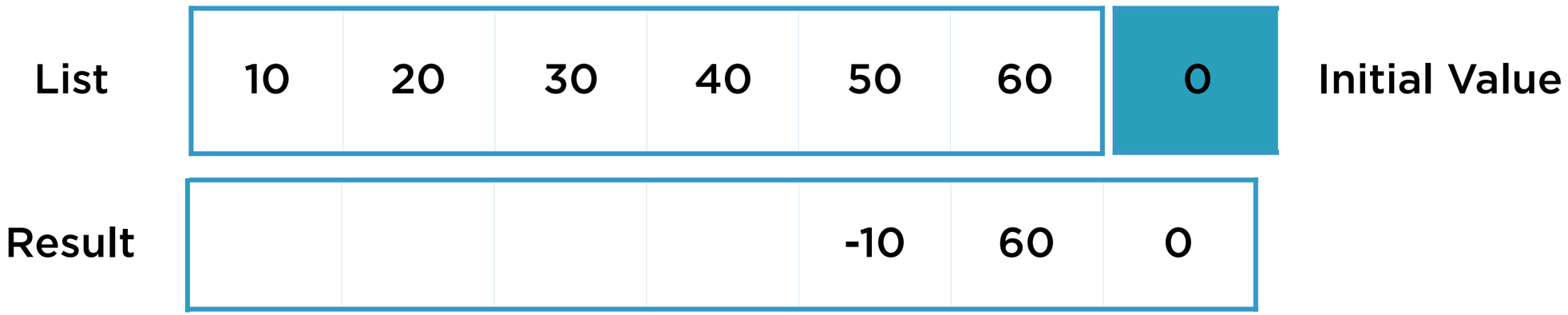
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



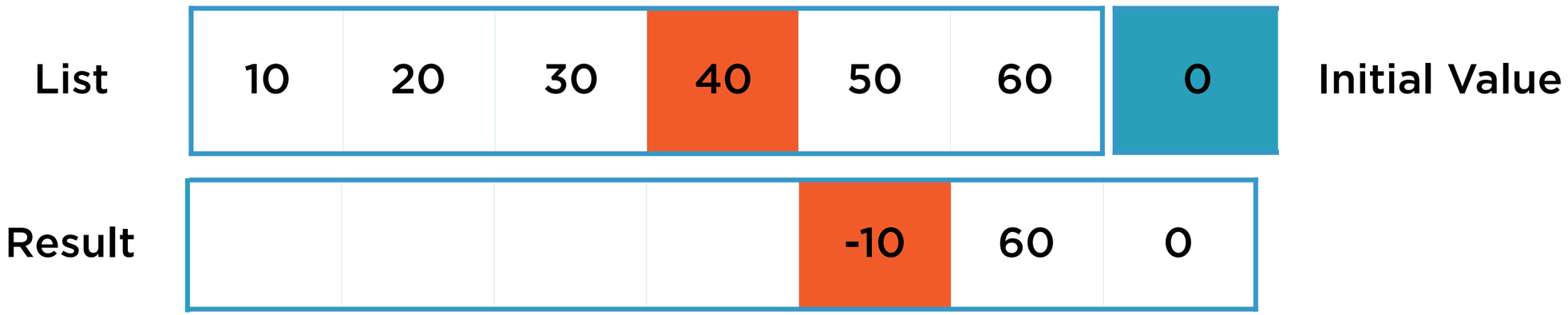
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



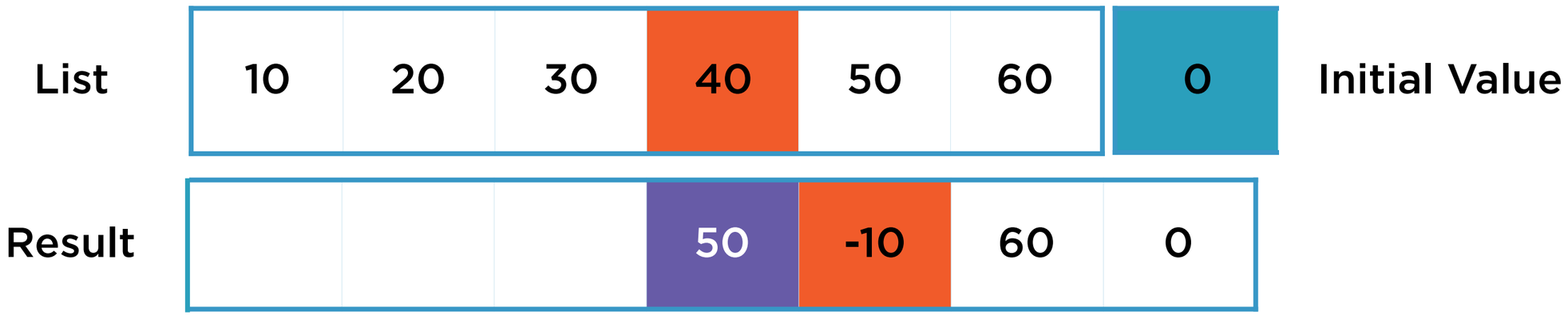
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



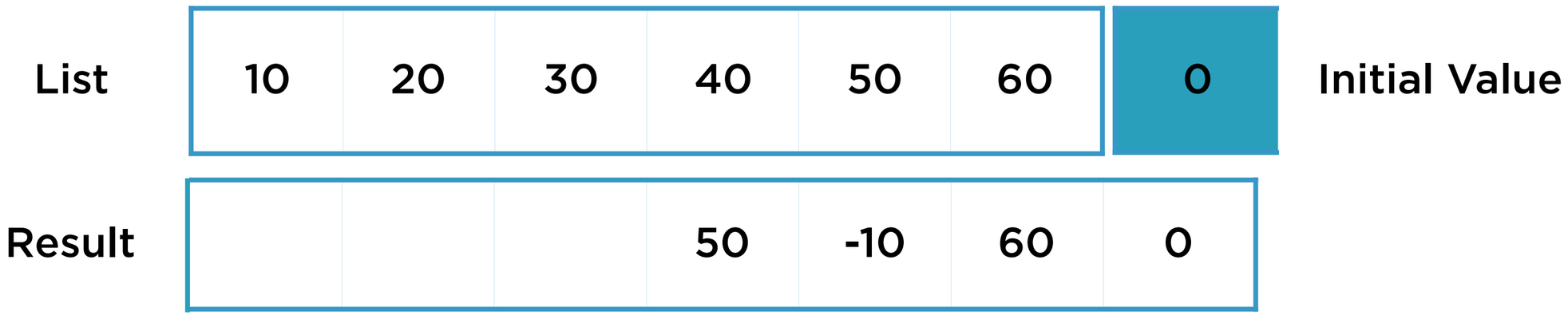
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



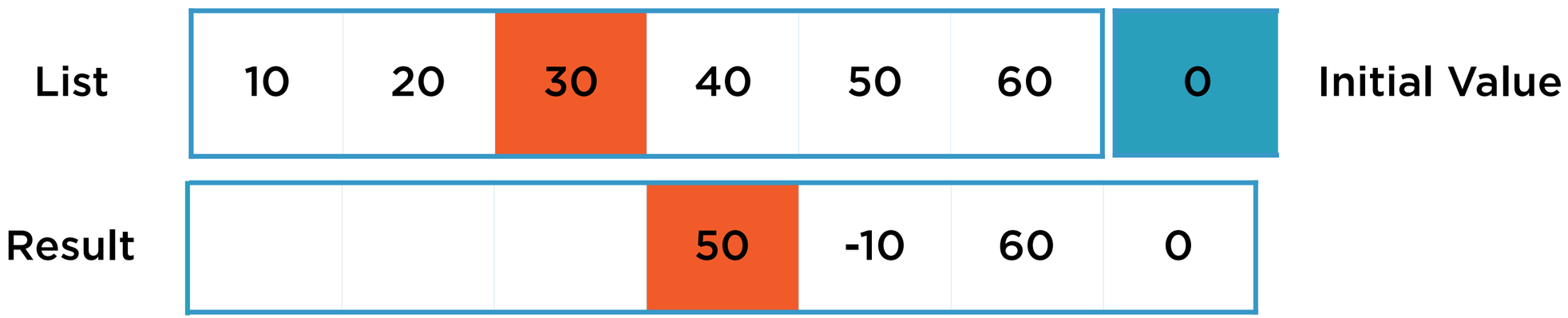
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



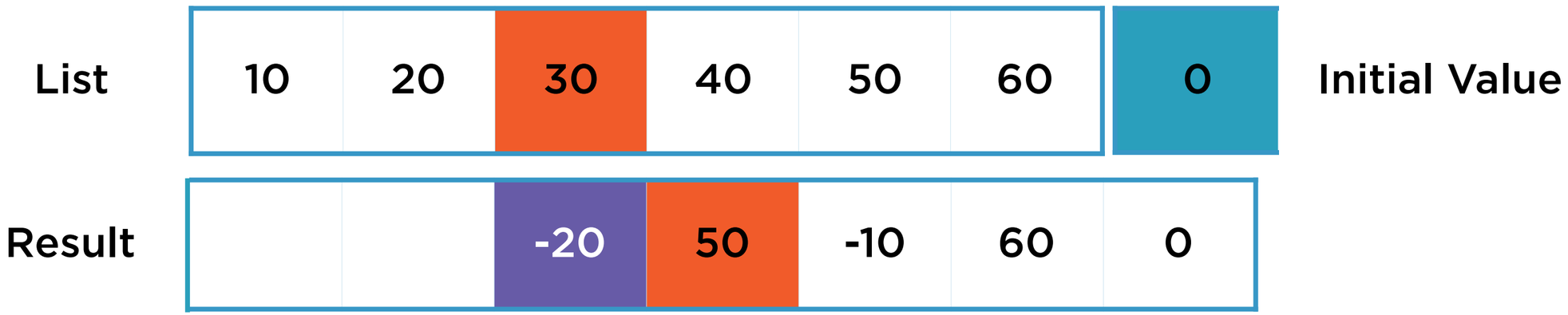
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



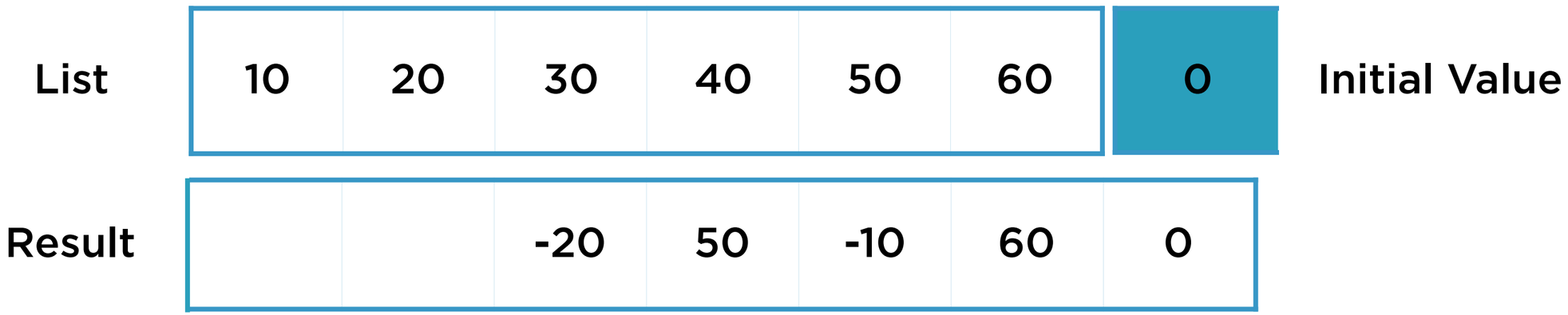
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



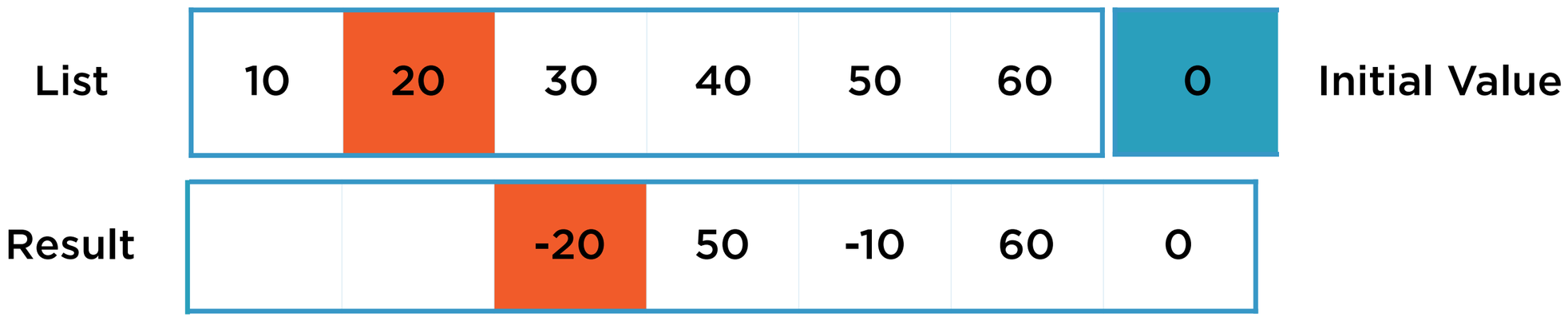
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



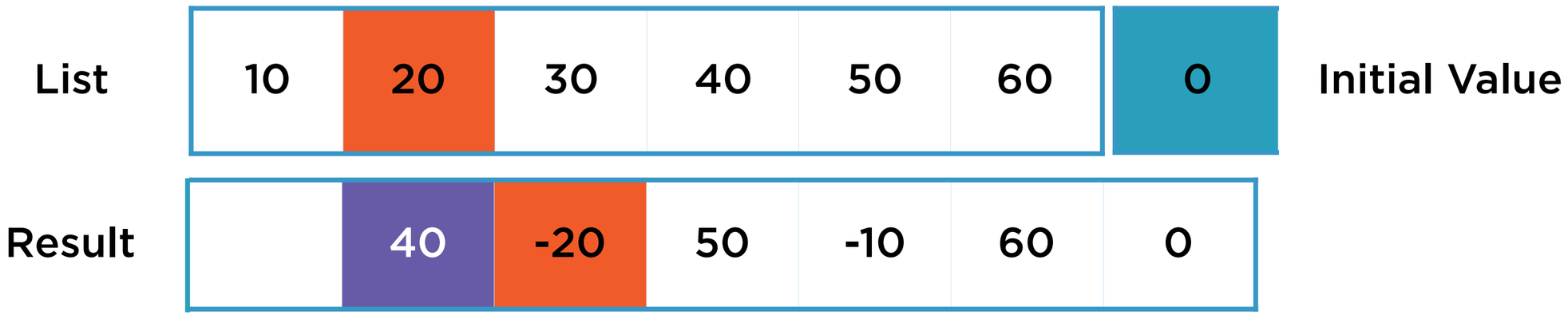
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



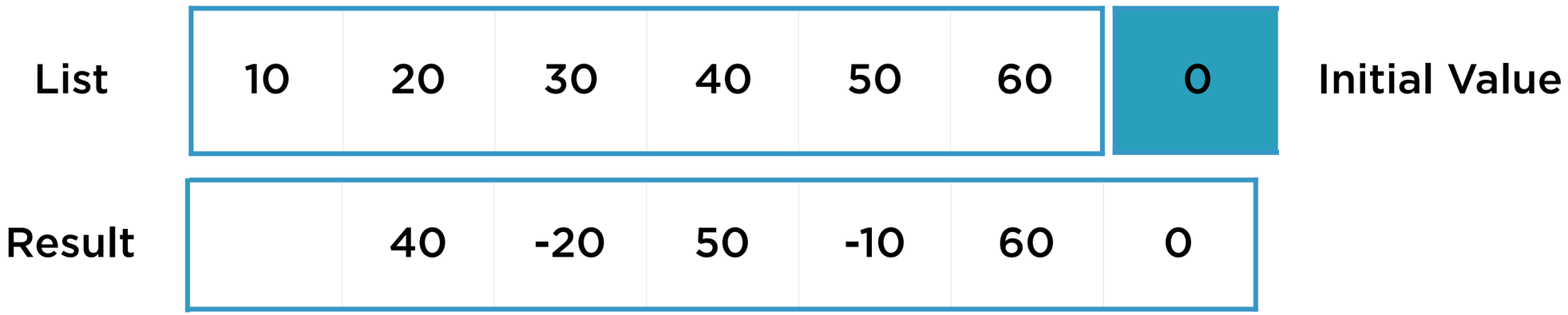
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



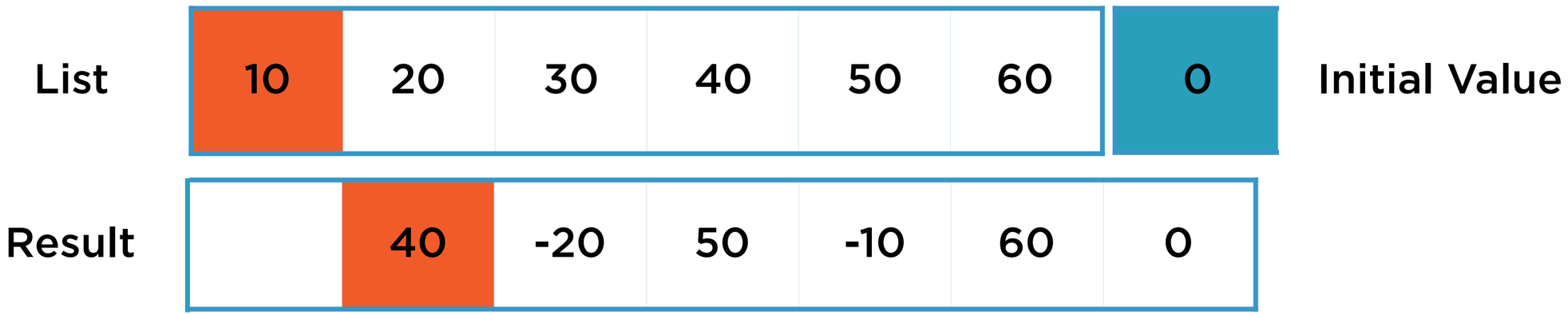
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



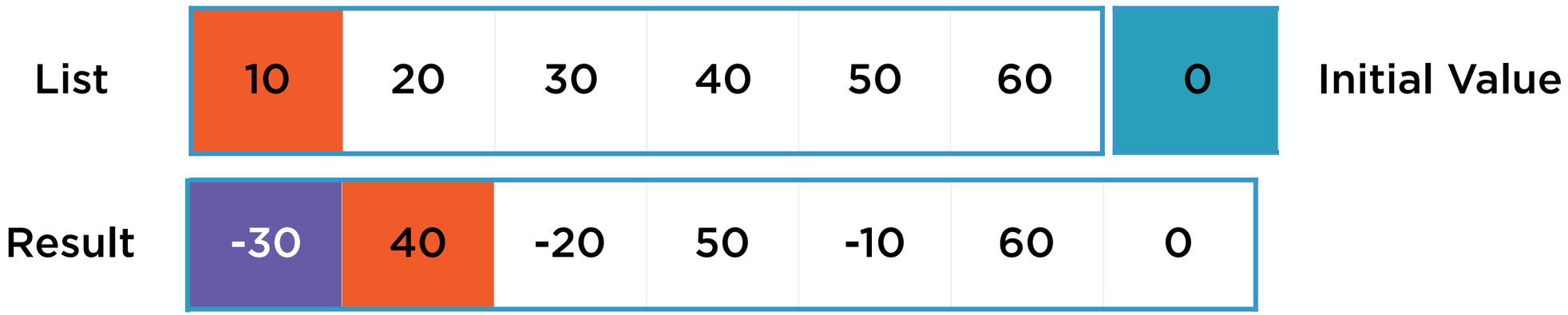
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



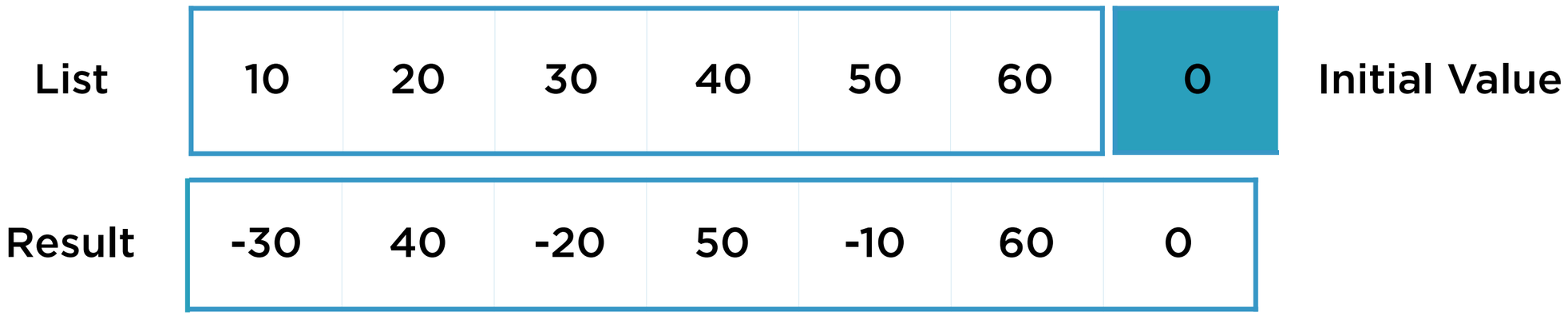
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



Scan, Fold and Reduce

Scan Right	Fold Right	Reduce Right
Scan Left	Fold Left	Reduce Left

Demo

Scan Left

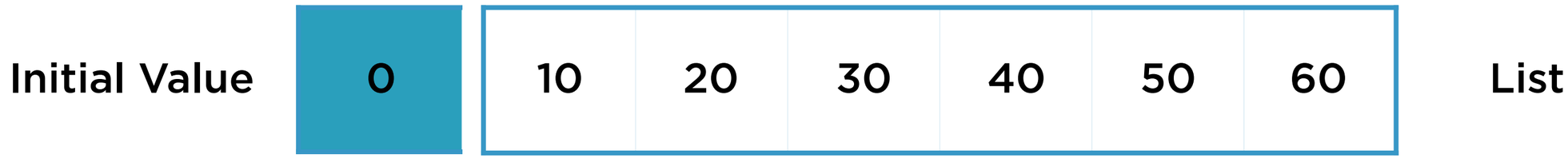
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft

A left-associative higher-order method that takes in an initial value and applies it pairwise, returning a list

```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

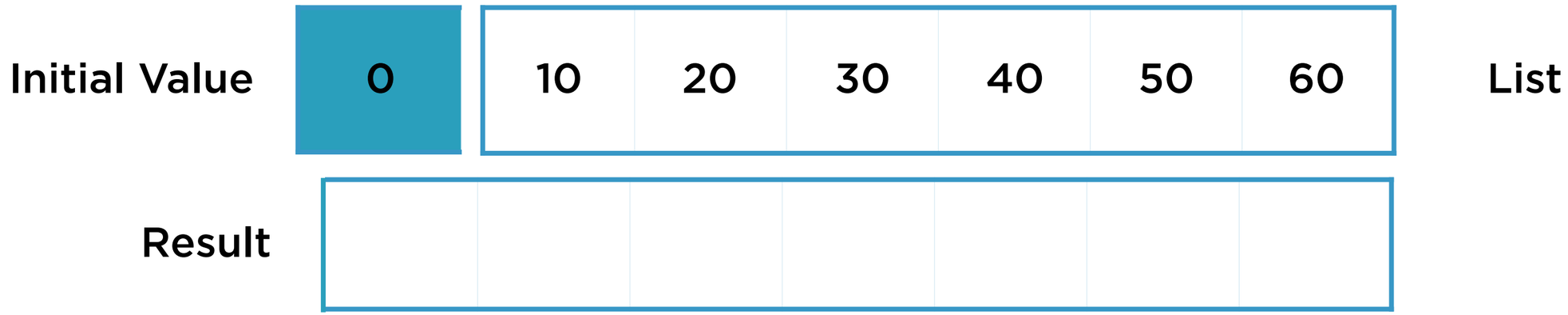
ScanLeft



Function _ – _

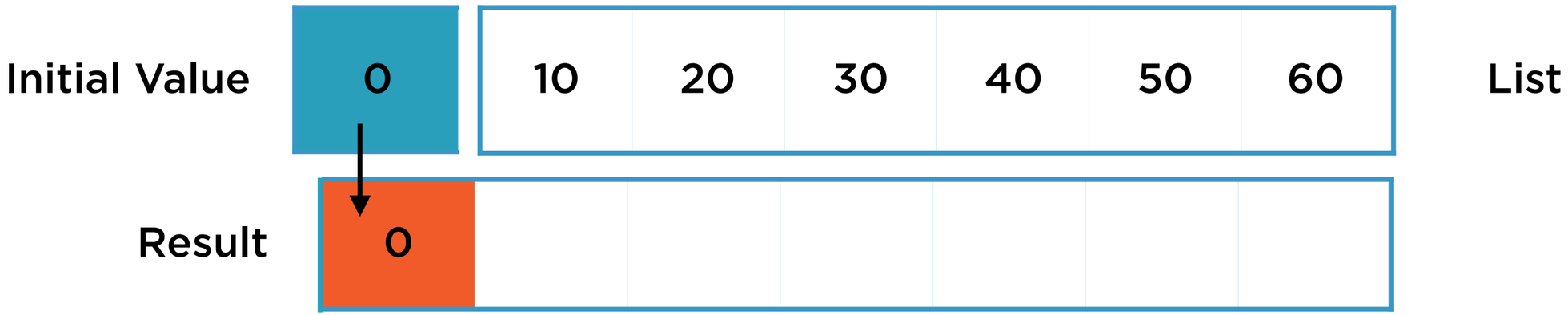
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



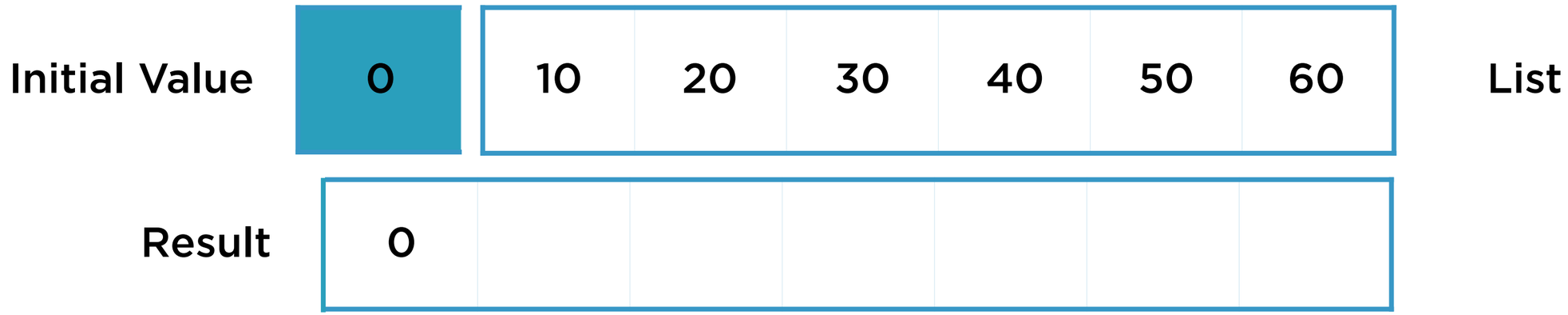
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



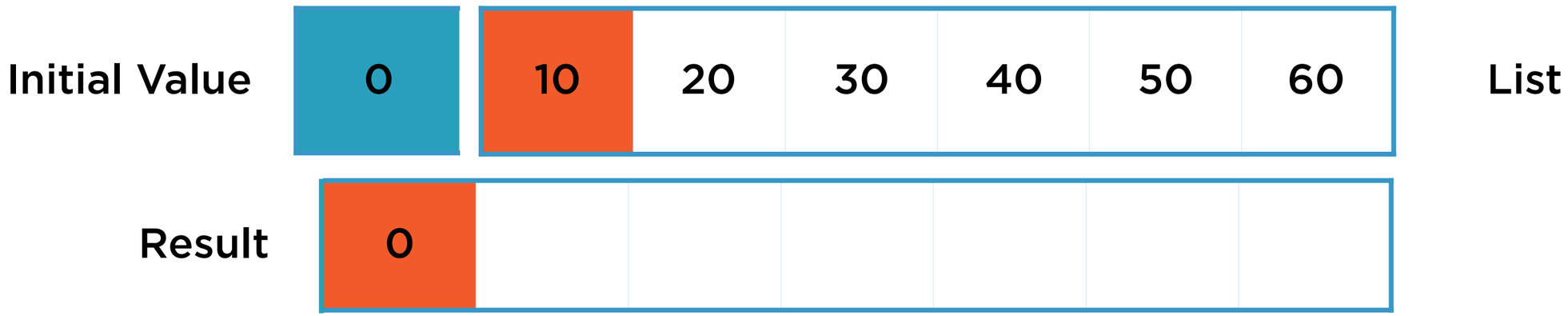
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



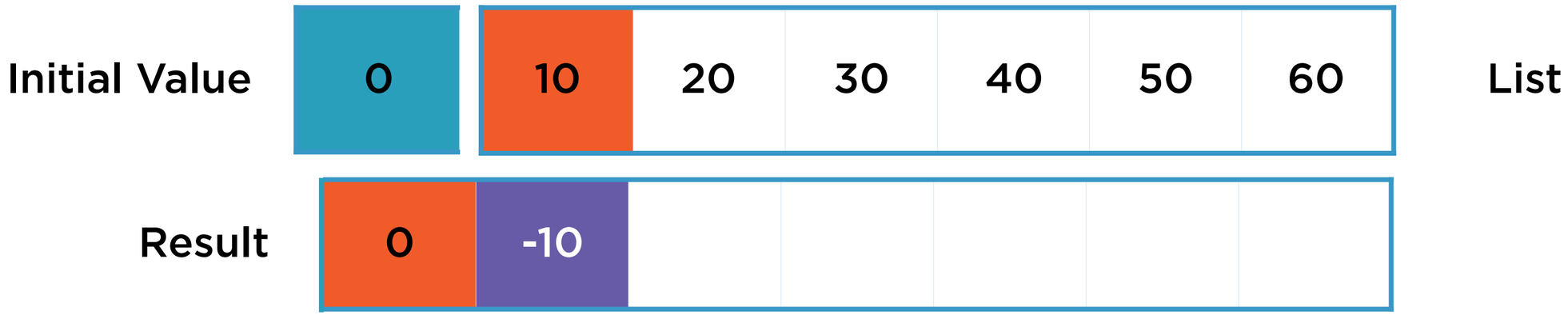
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



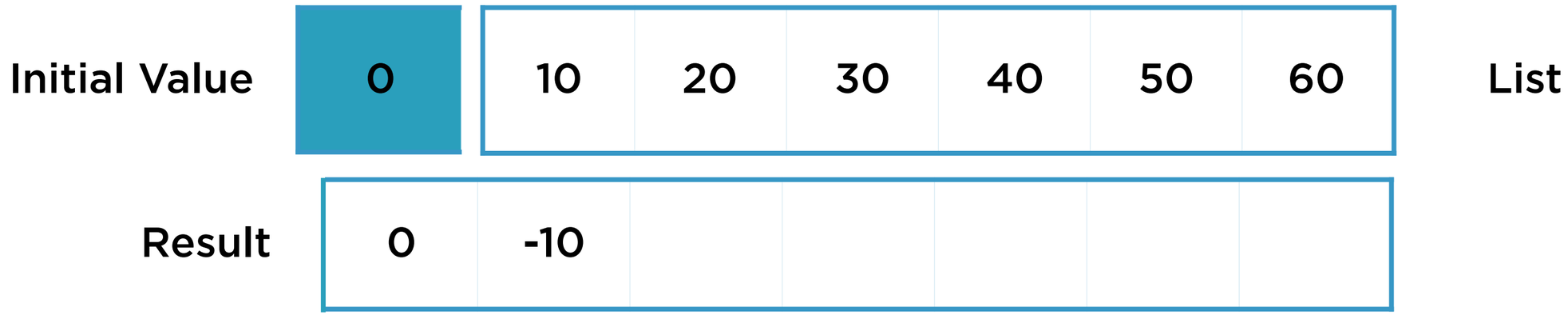
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



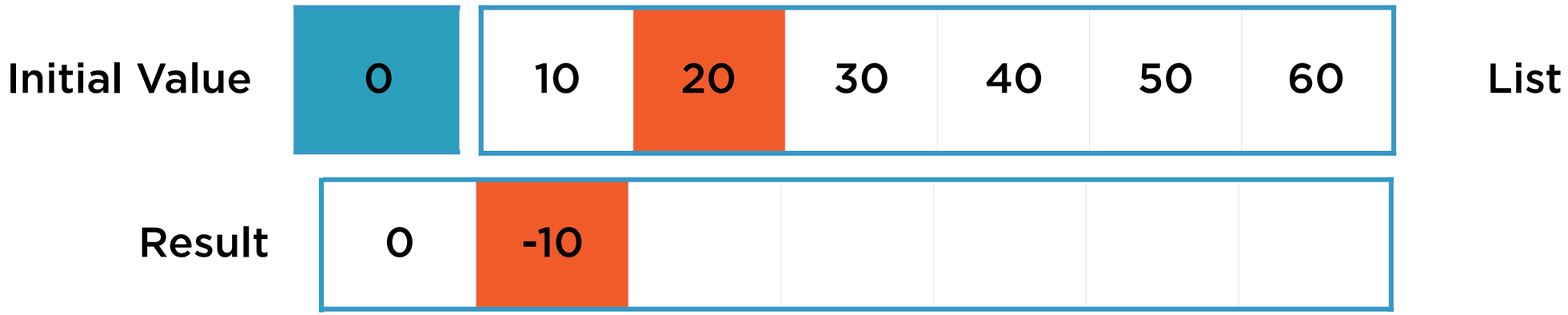
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft

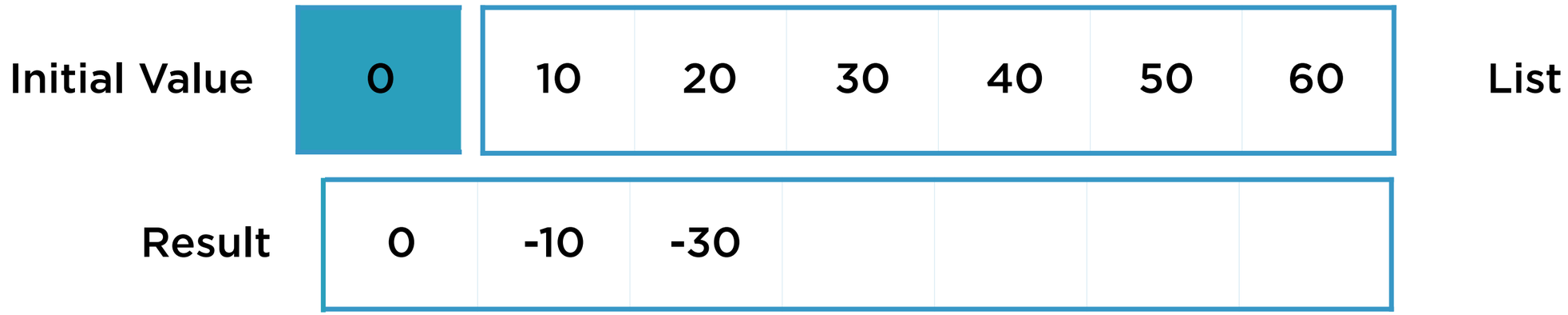


```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft

```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

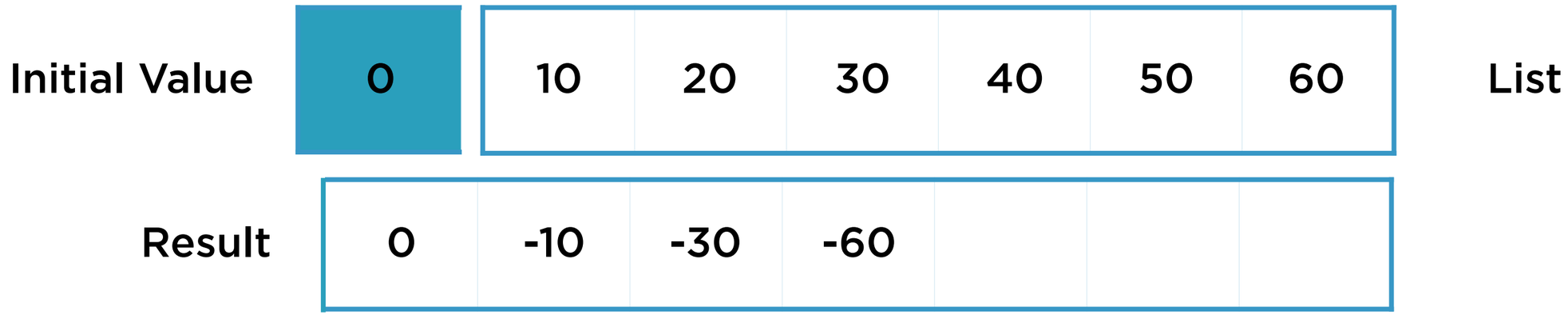
ScanLeft

```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft

```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

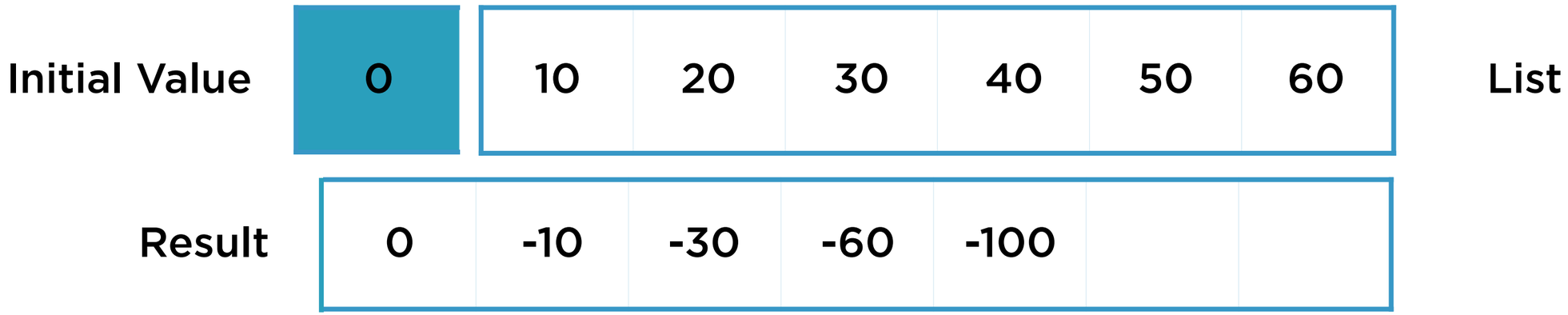
ScanLeft

```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft

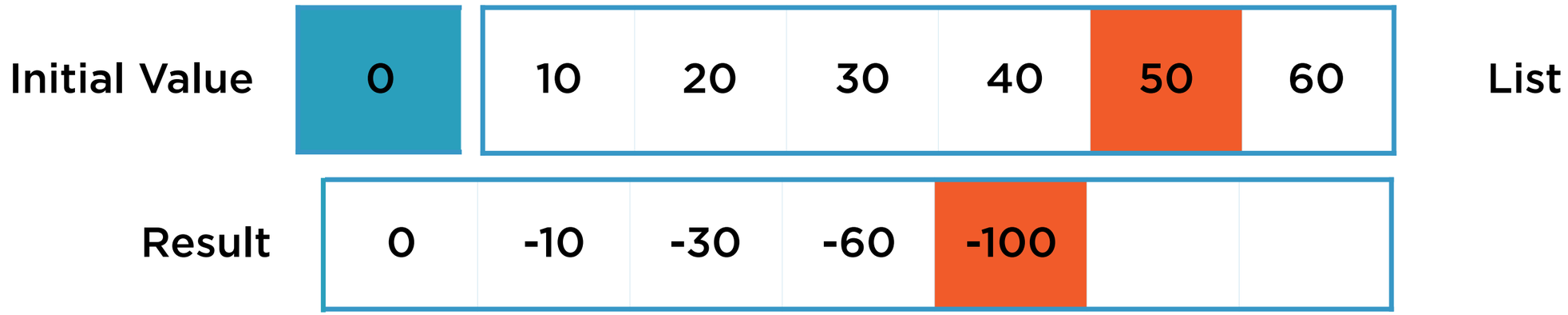
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



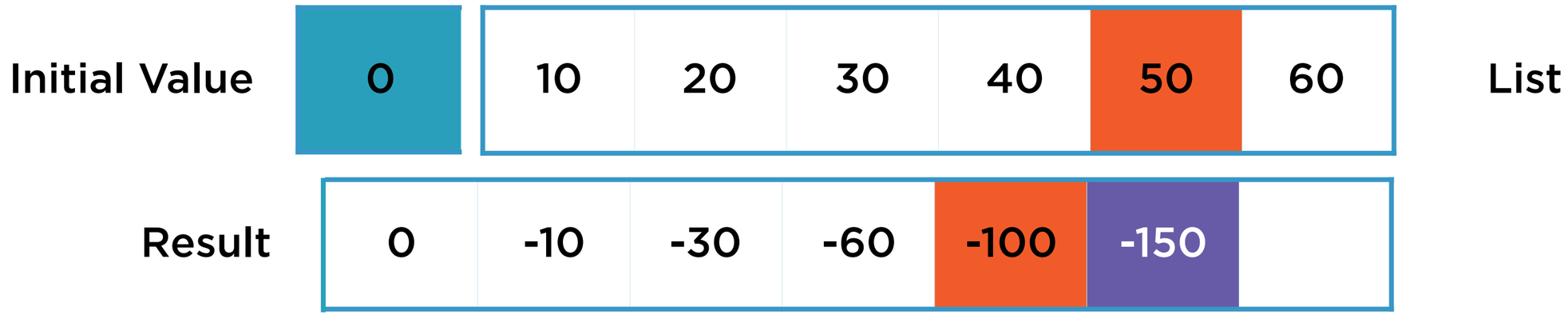
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



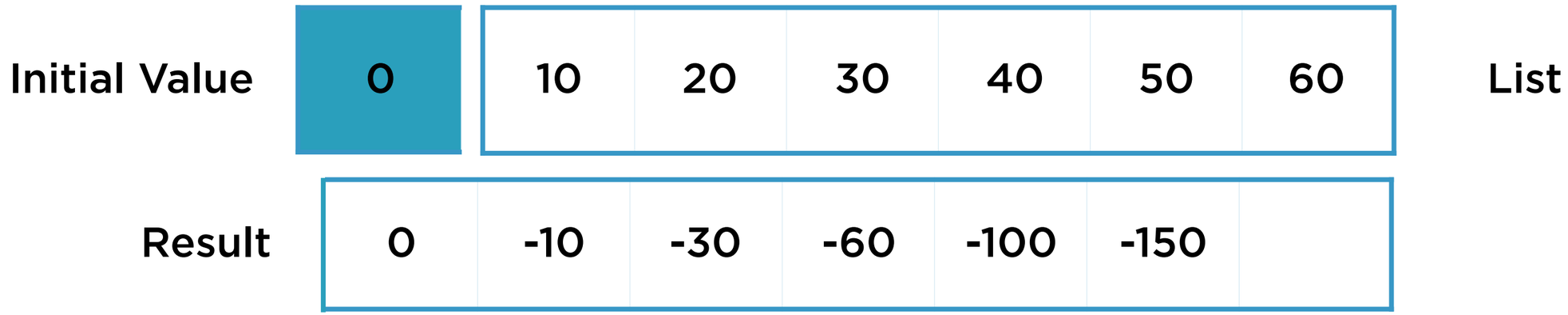
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



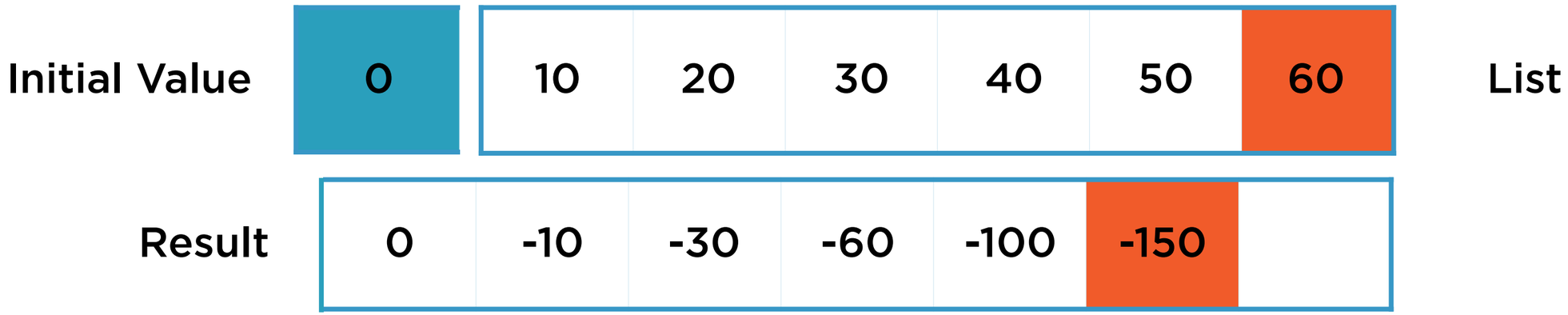
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



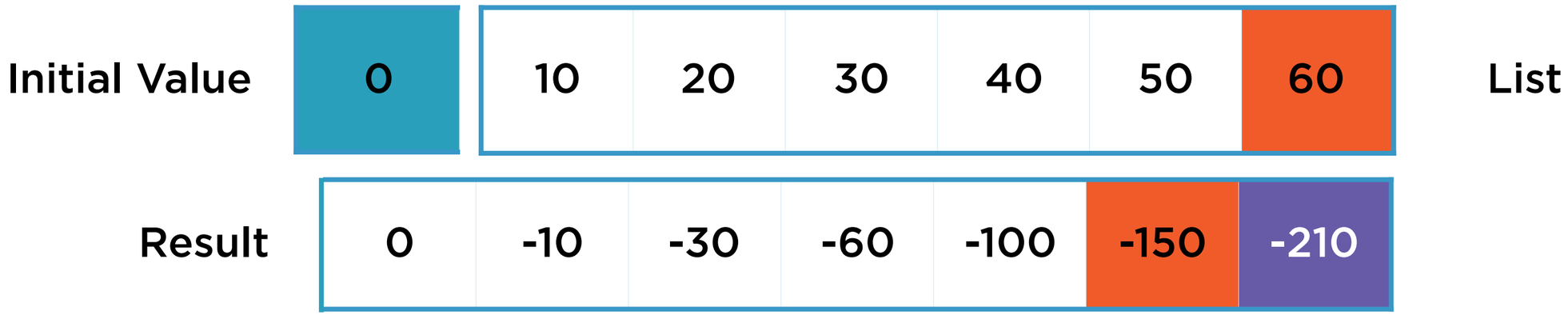
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



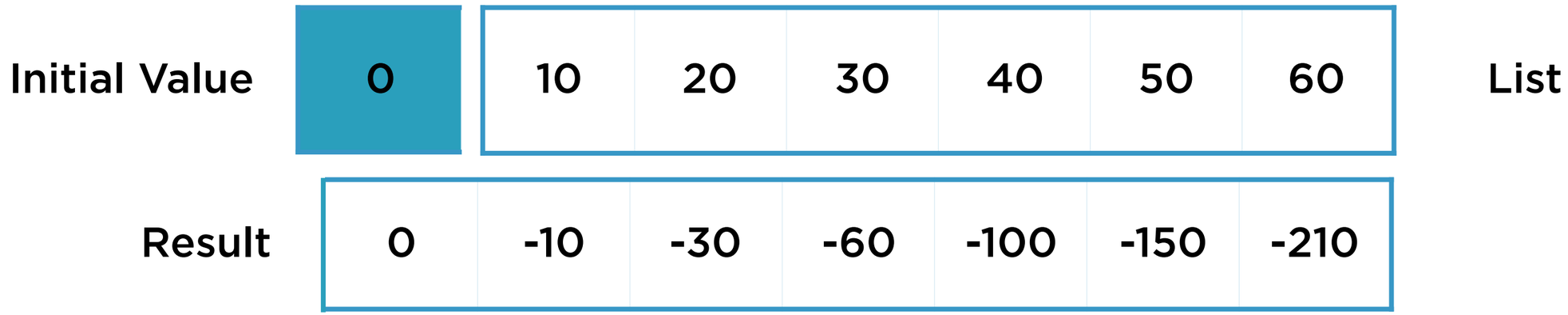
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



Scan, Fold and Reduce

Scan Right	**Fold Right**	Reduce Right
Scan Left	**Fold Left**	Reduce Left

Scan and Fold

Scan Left, Scan Right

Return the entire result list from the scan operation

Fold Left, Fold Right

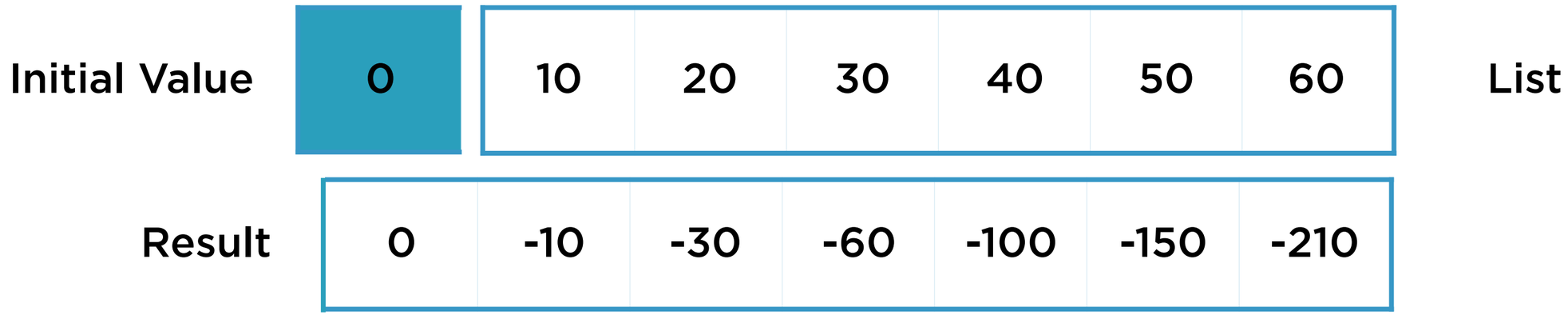
Return only the ‘last’ element of the result list

Demo

Fold Left and Fold Right

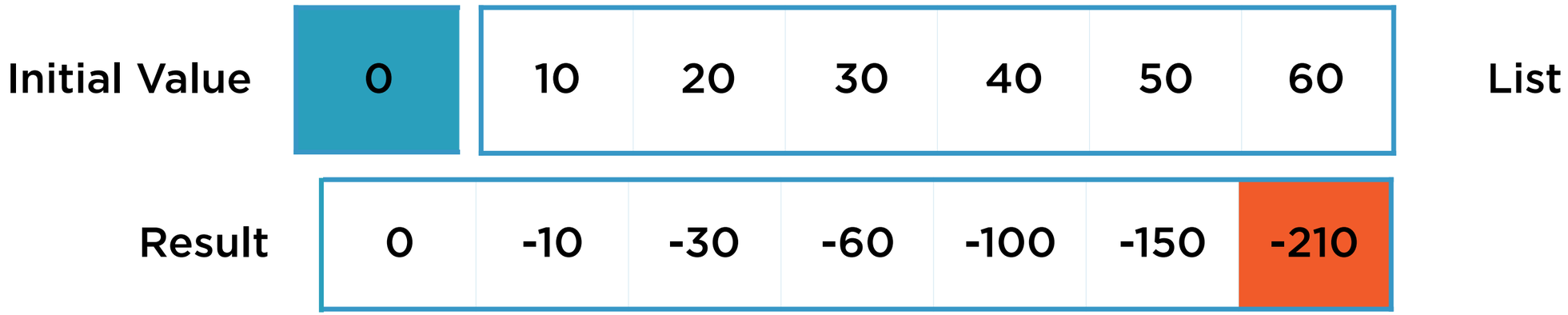
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



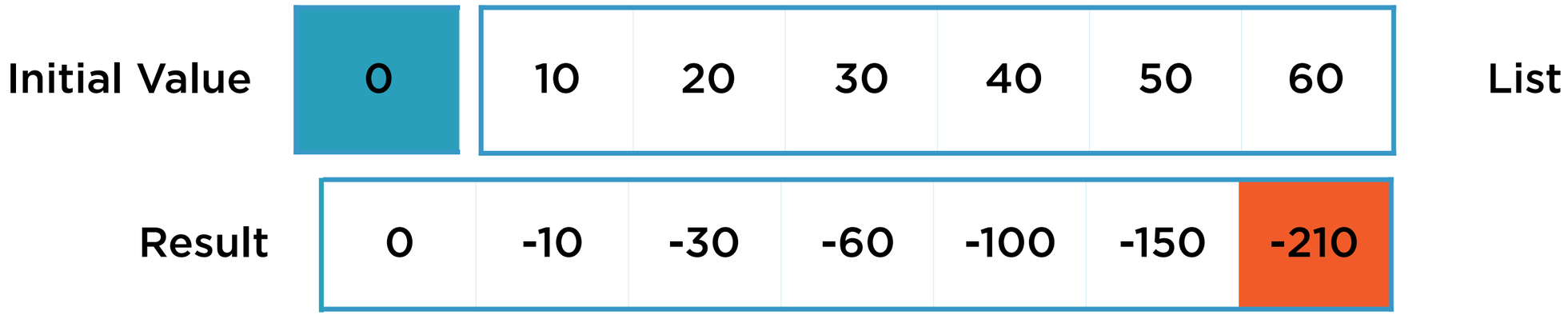
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanLeft(0)(_ - _)
List[Int] = List(0,-10,-30,-60,-100,-150,-210) // Result
```

ScanLeft



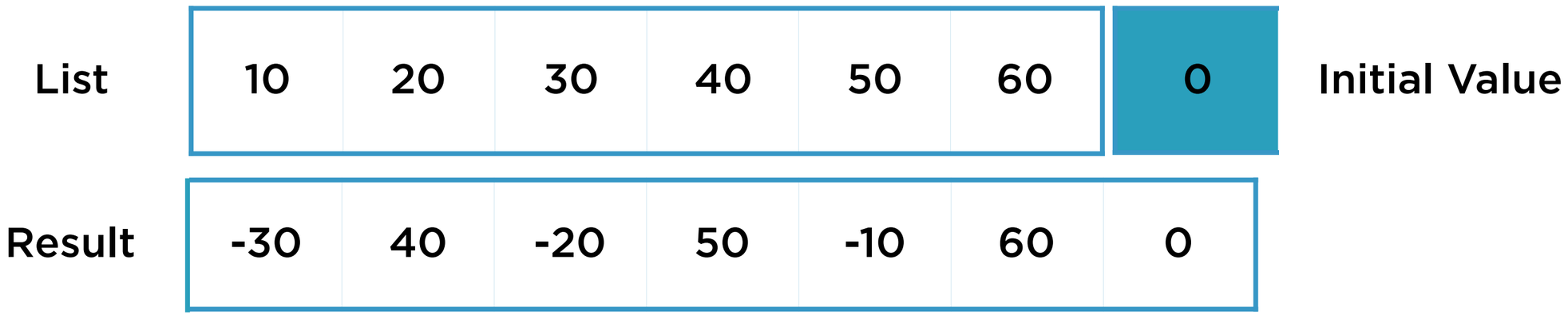
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.foldLeft(0)(_ - _)
Int = -210 // Result
```

FoldLeft



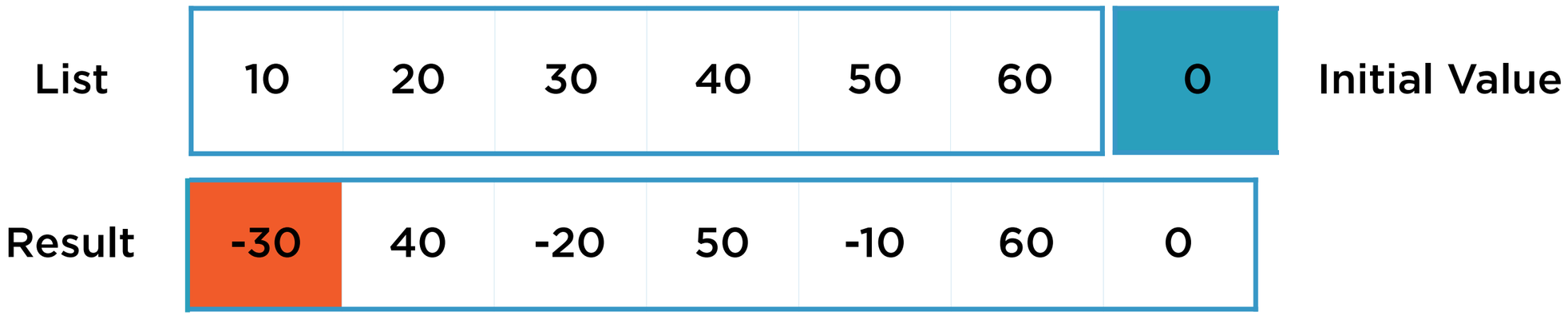
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



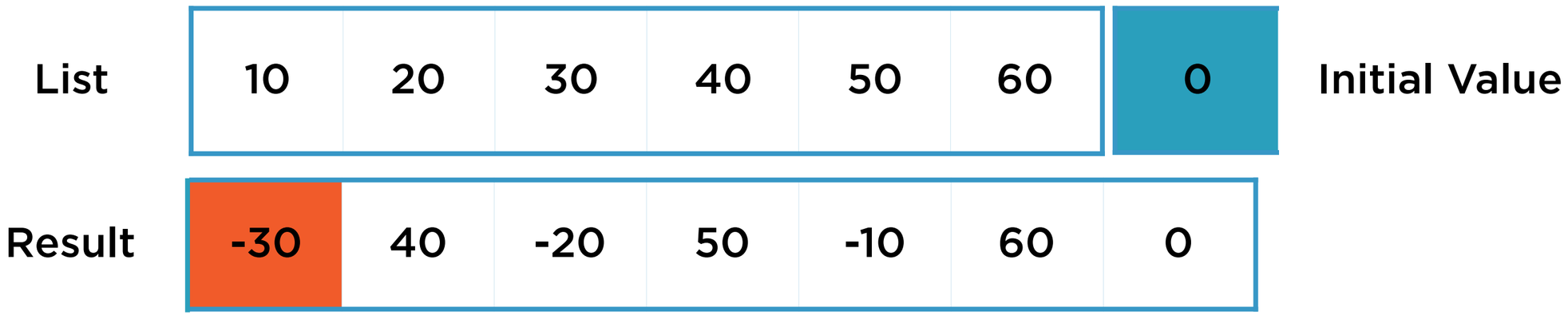
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.scanRight(0)(_ - _)
List[Int] = List(-30, 40, -20, 50, -10, 60, 0) // Result
```

ScanRight



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.foldRight(0)(_ - _)
 Int = -30     // Result
```

FoldRight



Scan, Fold and Reduce

Scan Right	Fold Right	Reduce Right
Scan Left	Fold Left	Reduce Left

Scan and Reduce

Scan Left, Scan Right

Take in an initial value; use this initial value as second operand in first step

Reduce Left, Reduce Right

No initial value - first two list elements as operands in first step

Demo

Reduce Right

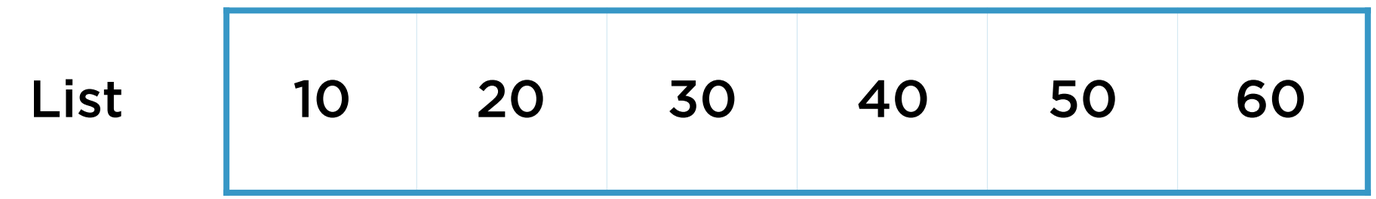
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30      // Result
```

ReduceRight

A right-associative higher-order method that does not take an initial value, instead using the first two list elements as operands

```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30       // Result
```

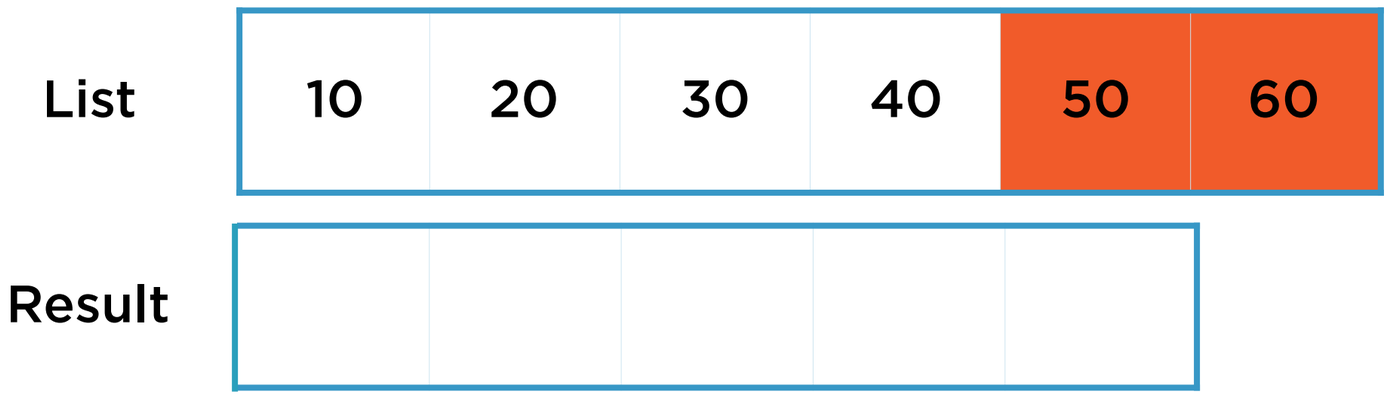
ReduceRight



Function `_ - _`

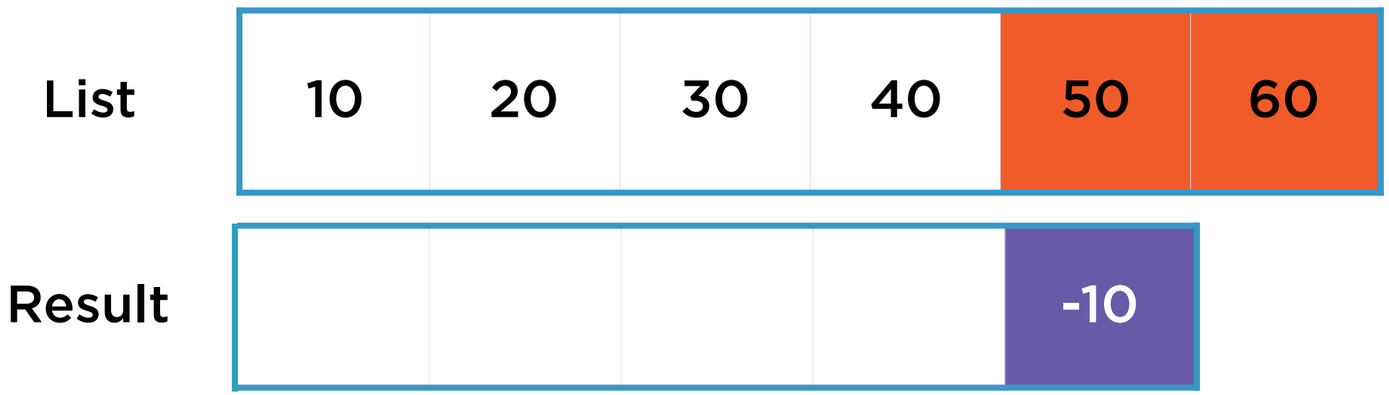
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
 Int = -30      // Result
```

ReduceRight



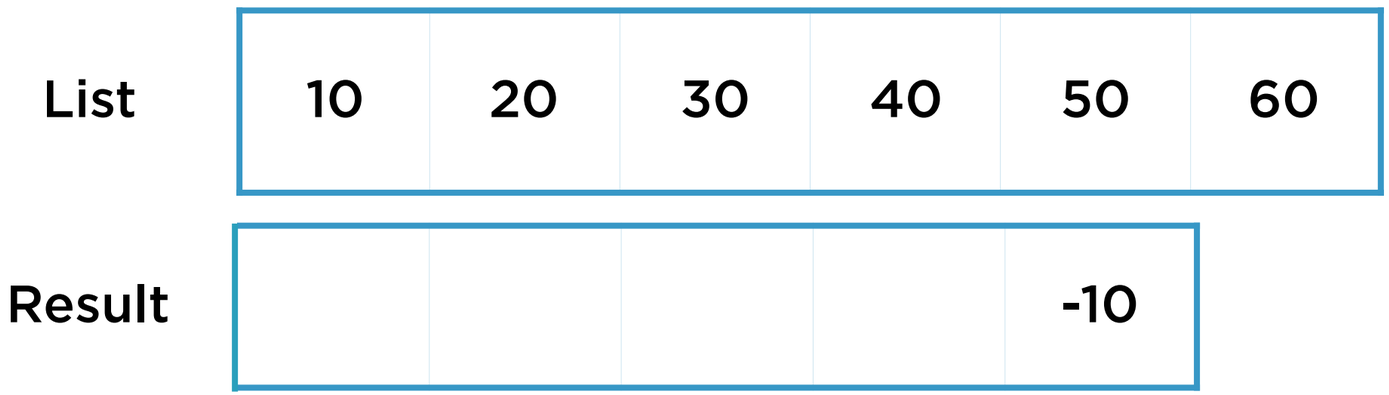
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
 Int = -30      // Result
```

ReduceRight



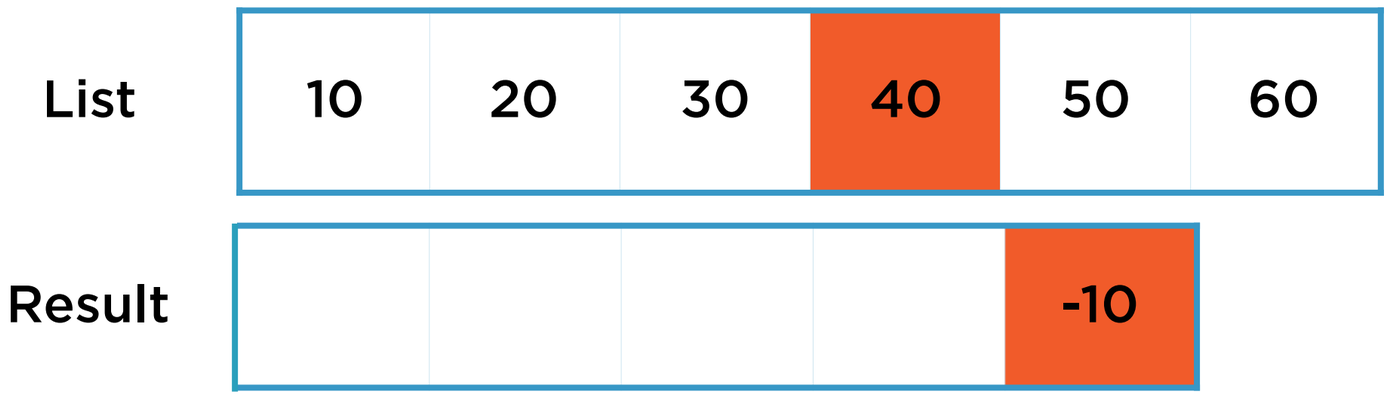
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
 Int = -30      // Result
```

ReduceRight



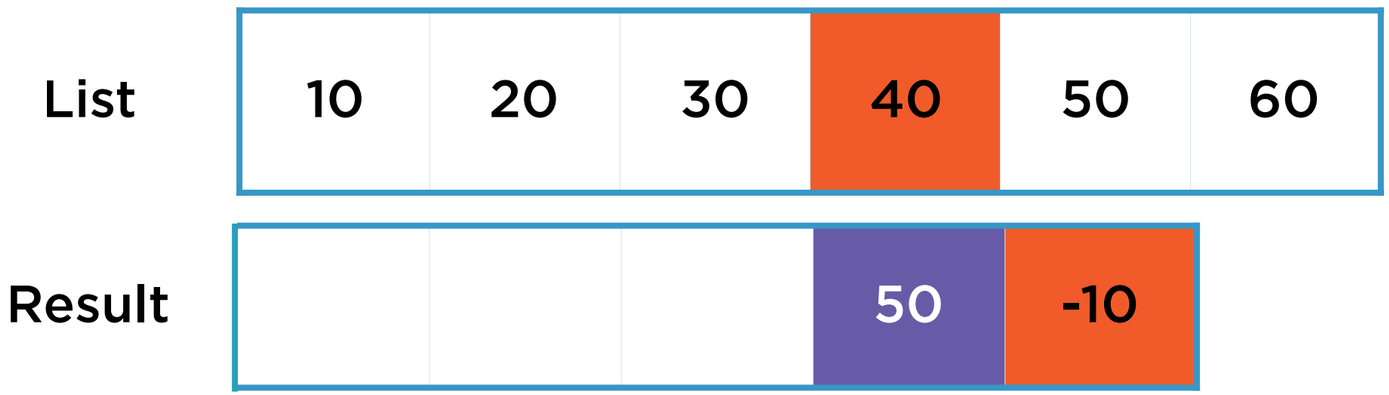
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30      // Result
```

ReduceRight



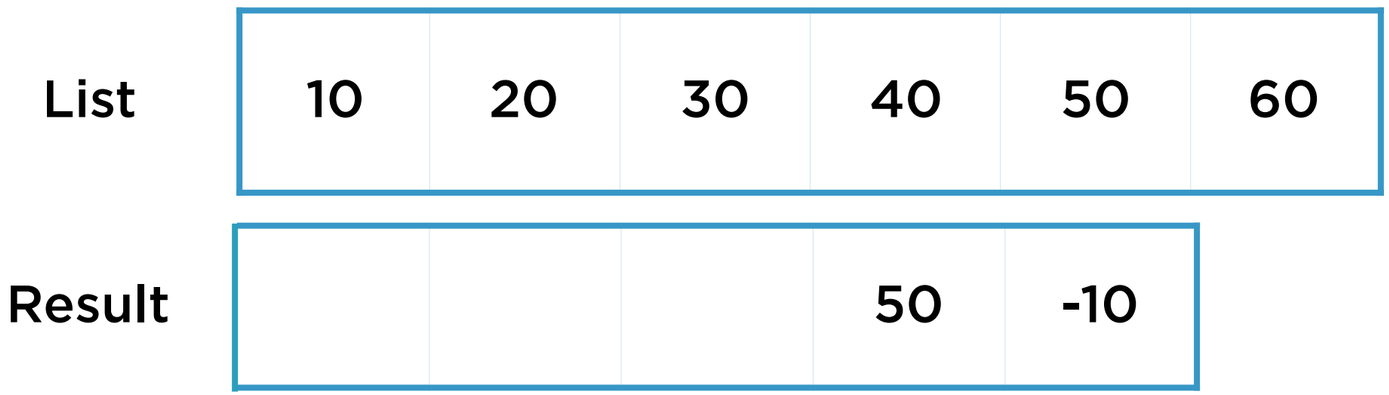
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30       // Result
```

ReduceRight



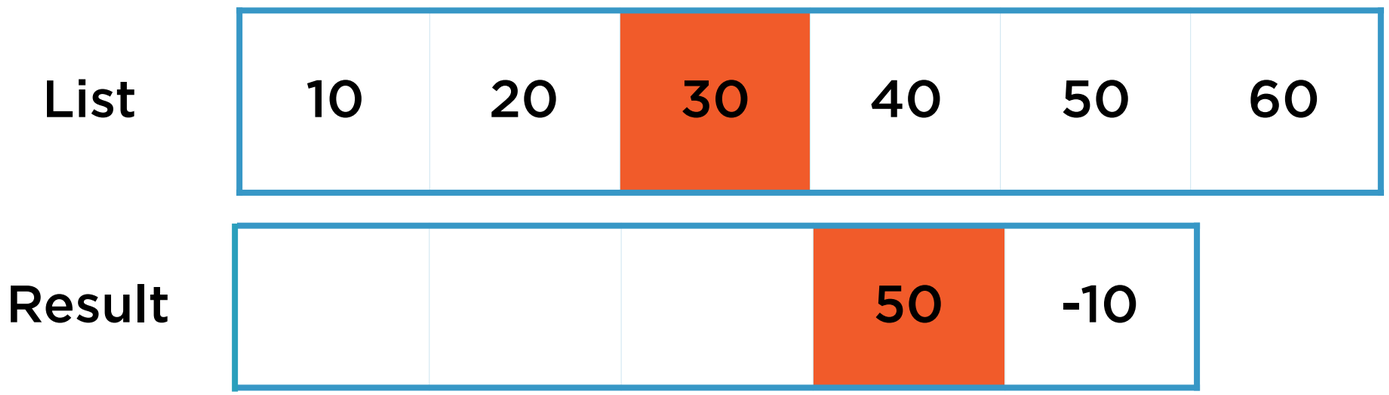
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30      // Result
```

ReduceRight



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30       // Result
```

ReduceRight

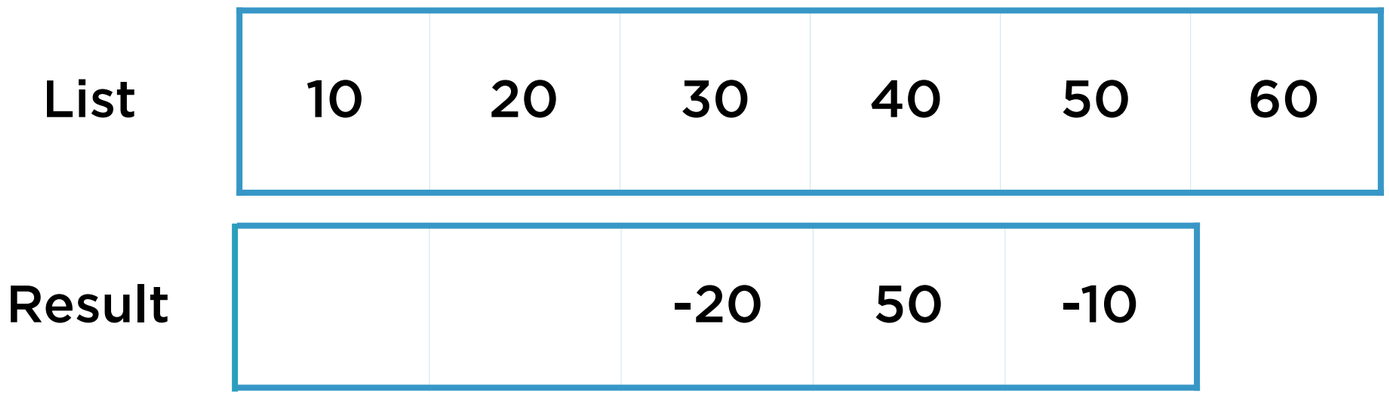


```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30       // Result
```

ReduceRight

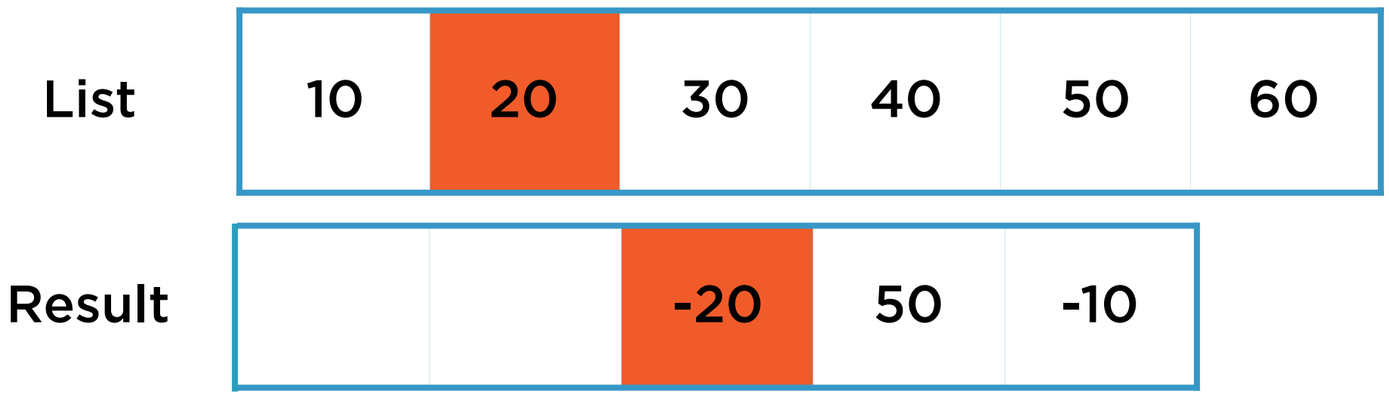
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30       // Result
```

ReduceRight



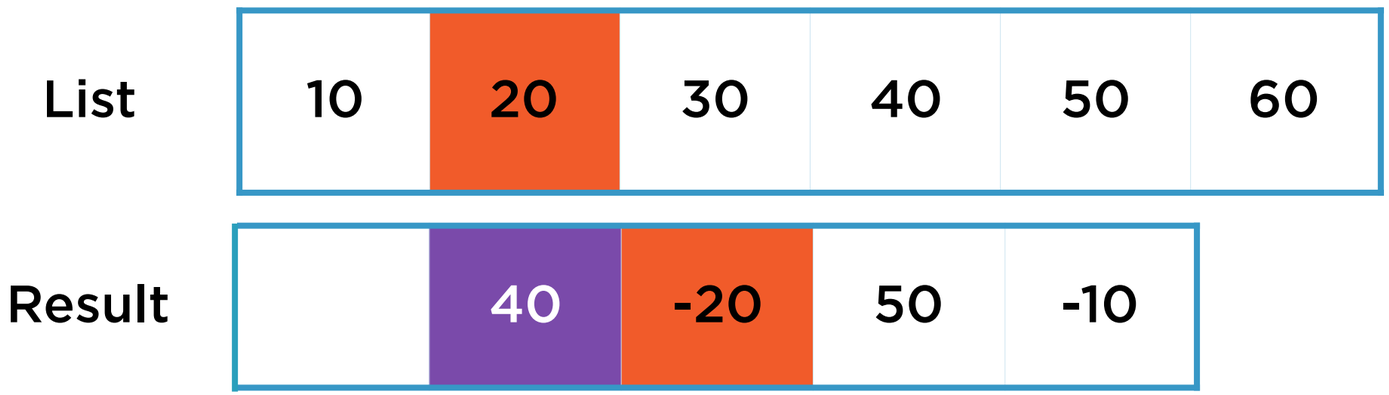
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30       // Result
```

ReduceRight



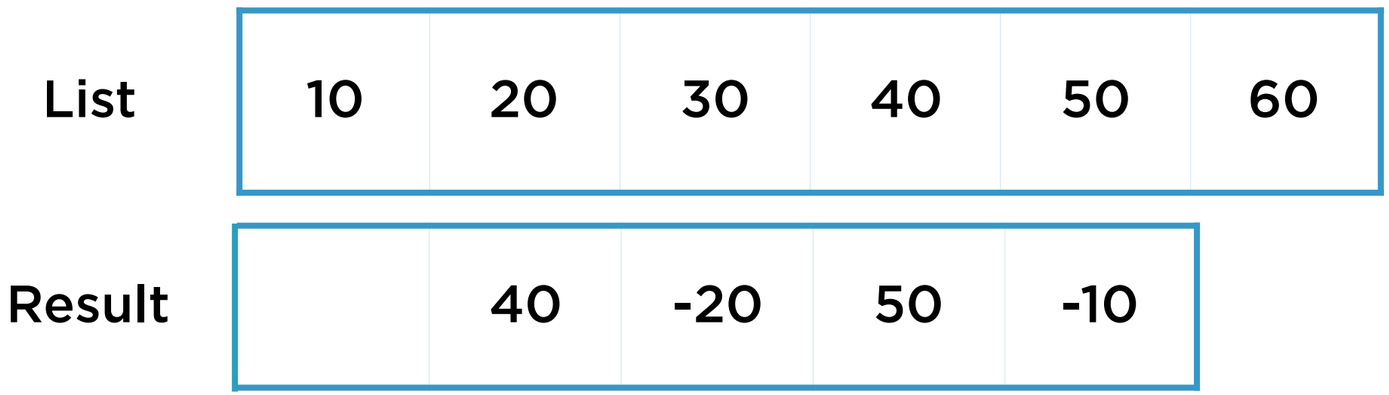
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30       // Result
```

ReduceRight



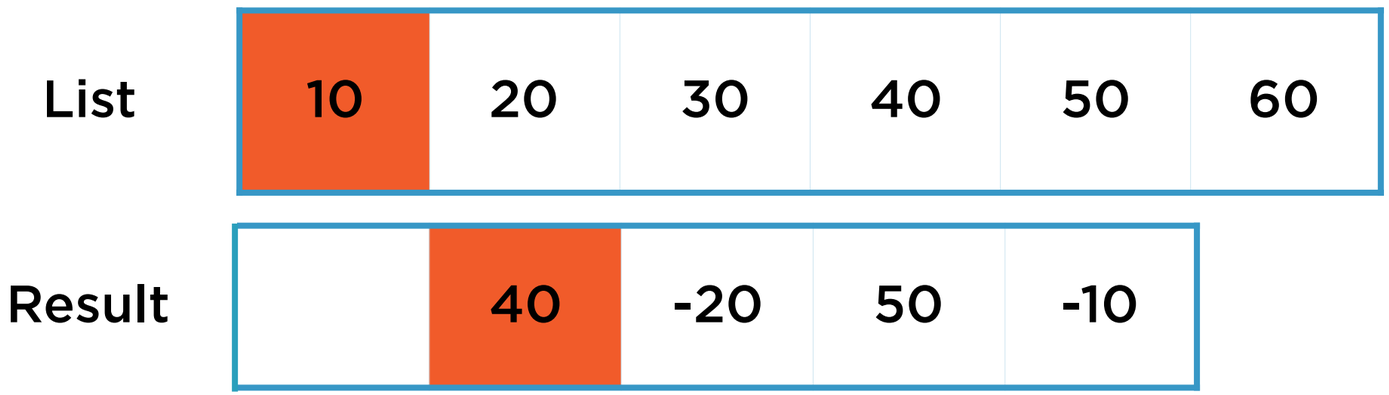
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30      // Result
```

ReduceRight



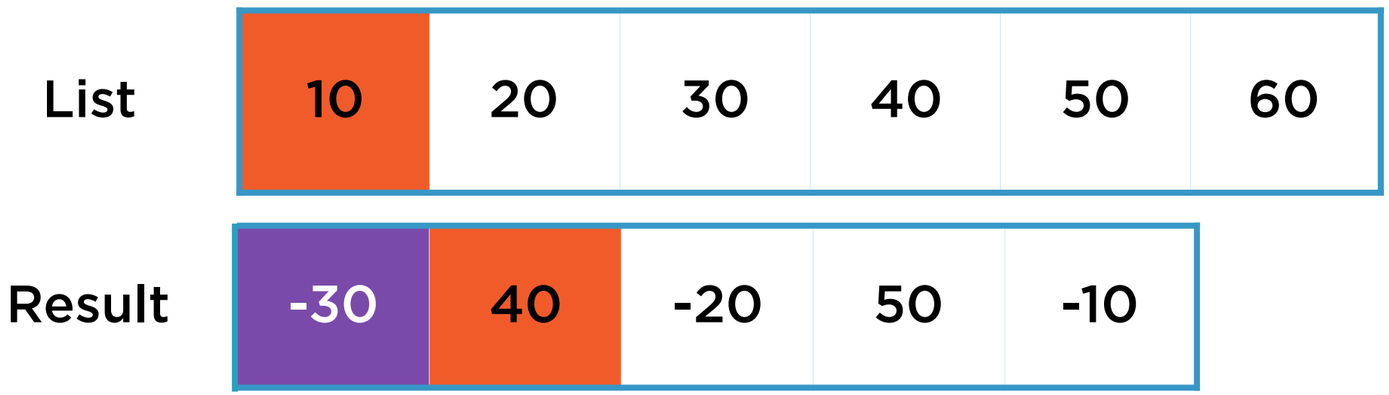
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
 Int = -30      // Result
```

ReduceRight



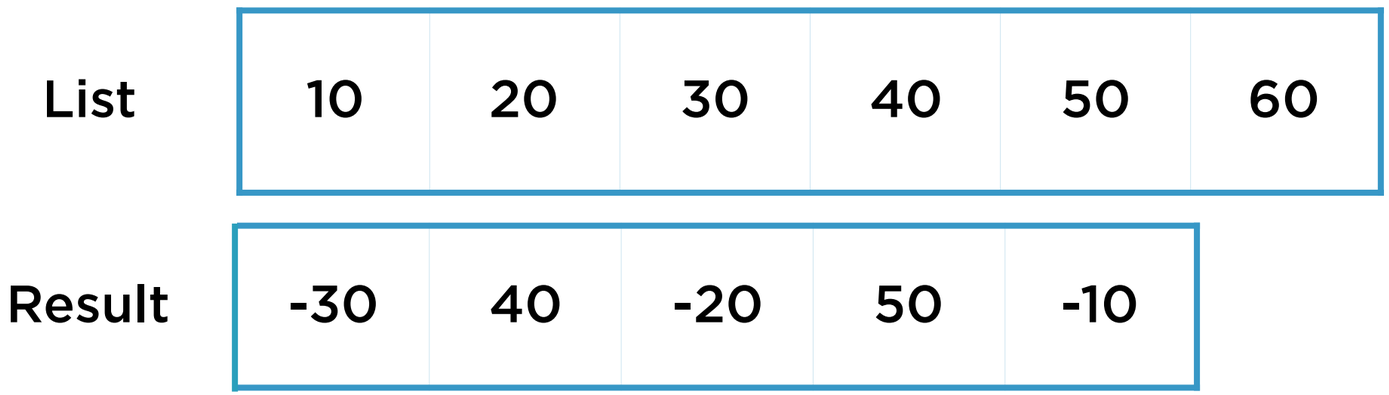
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
 Int = -30      // Result
```

ReduceRight



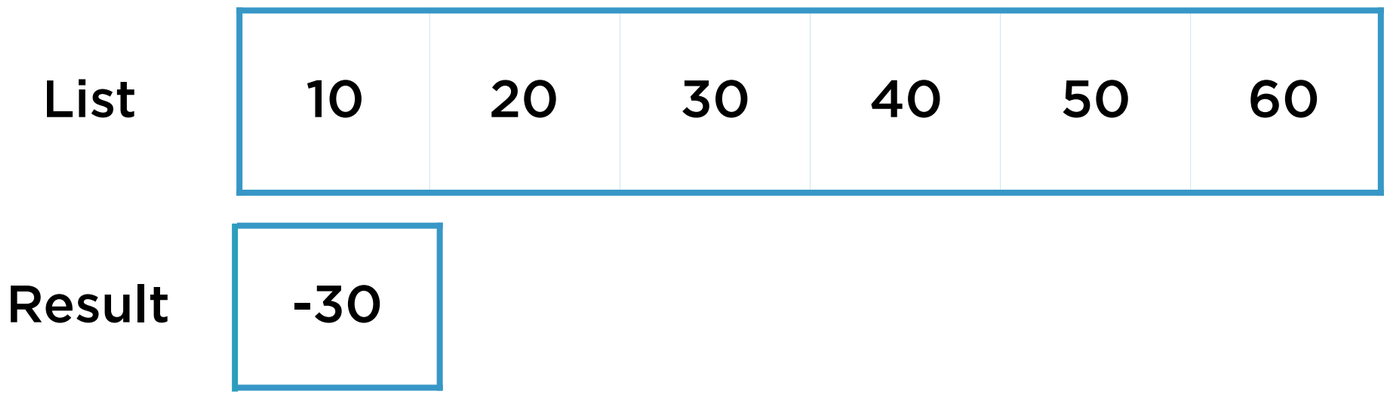
```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30      // Result
```

ReduceRight



```
val someNumbers = List(10,20,30,40,50,60)
someNumbers.reduceRight(_ - _)
Int = -30      // Result
```

ReduceRight



Scan, Fold and Reduce

Scan Right	Fold Right	Reduce Right
Scan Left	Fold Left	Reduce Left

Summary

Higher order methods of collections are a key functional construct

Map, for-each, filter apply function objects to one element of the collection at a time

Variants of scan, fold and reduce operate on multiple container elements at a time