BINARY SEARCH

(I) Left = mid + 1 & Right = mid – 1 ➔ while (left <= right) (Acc. To)

Case	Before	Value of Mid	array[mid] > target {value of *Right* changes }	array[mid] < target {value of *Left* changes }
For 3 elements Array = [0, 1, 2]	Left = 0 Right = 2	Mid = 1	Left = 0 Right = 0	Left = 2 Right = 2
For 2 elements Array = [0, 1]	Left = 0 Right = 1	Mid = 0	Left = 0 Right = -1	Left = 1 Right = 1
For 1 elements Array = [0]	Left = 0 Right = 0	Mid = 0	Left = 0 Right = -1	Left = 1 Right = 0

(II) Left = mid + 1 & Right = mid ➔ while (left < right) (Acc. To)

Case	Before	Value of Mid	array[mid] > target {value of *Right* changes }	array[mid] < target { value of *Left* changes }
For 3 elements Array = [0, 1, 2]	Left = 0 Right = 2	Mid = 1	Left = 0 Right = 1	Left = 2 Right = 2
For 2 elements Array = [0, 1]	Left = 0 Right = 1	Mid = 0	Left = 0 Right = 0	Left = 1 Right = 1
For 1 elements Array = [0]	Left = 0 Right = 0	Mid = 0	Left = 0 Right = 0	Left = 1 Right = 0

(III) Left = mid & Right = mid – 1 ➔ while (left + 1 < right) (Acc. To)

Case	Before	Value of Mid	array[mid] > target {value of *Right* changes }	array[mid] < target { value of *Left* changes }
For 3 elements Array = [0, 1, 2]	Left = 0 Right = 2	Mid = 1	Left = 0 Right = 0	Left = 1 Right = 2
For 2 elements Array = [0, 1]	Left = 0 Right = 1	Mid = 0	Left = 0 Right = -1	Left = 0 Right = 1
For 1 elements Array = [0]	Left = 0 Right = 0	Mid = 0	Left = 0 Right = -1	Left = 0 Right = 0

(IV) Left = mid & Right = mid ➔ while (left + 1 < right) (Acc. To)

Case	Before	Value of Mid	array[mid] > target {value of *Right* changes }	array[mid] < target { value of *Left* changes }
For 3 elements Array = [0, 1, 2]	Left = 0 Right = 2	Mid = 1	Left = 0 Right = 1	Left = 1 Right = 2
For 2 elements Array = [0, 1]	Left = 0 Right = 1	Mid = 0	Left = 0 Right = 0	Left = 0 Right = 1
For 1 elements Array = [0]	Left = 0 Right = 0	Mid = 0	Left = 0 Right = 0	Left = 0 Right = 0