

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(log(n)) time | O(1) space
5     func shiftedBinarySearch(_ array: [Int], _ target: Int) -> Int {
6         var leftPointer = 0
7         var rightPointer = array.count - 1
8
9         return shiftedBinarySearchHelper(array, target, &leftPointer,
10     }
11
12     func shiftedBinarySearchHelper(_ array: [Int], _ target: Int, _ left: Int, _ right: Int) -> Int {
13         while leftPointer <= rightPointer {
14             let middle = (leftPointer + rightPointer) / 2
15             let potentialMatch = array[middle]
16             let leftNumber = array[leftPointer]
17             let rightNumber = array[rightPointer]
18
19             if target == potentialMatch {
20                 return middle
21             } else if leftNumber < potentialMatch {
22                 if target < potentialMatch, target >= leftNumber {
23                     rightPointer = middle - 1
24                 } else {
25                     leftPointer = middle + 1
26                 }
27             } else {
28                 if target <= rightNumber, target > potentialMatch {
29                     leftPointer = middle + 1
30                 } else {
31                     rightPointer = middle - 1
32                 }
33             }
34         }
35     }
36 }
```

Solution 1

Solution 2

Solution 3

```
1 class Program {
2     func shiftedBinarySearch(_ array: [Int], _ target: Int) -> Int {
3         // Write your code here.
4         return -1
5     }
6 }
7
```

Our Tests

Custom Output

Submit Code

```
1 class Program {
2     func shiftedBinarySearch(_ array: [Int], _ target: Int) -> Int {
3         // Write your code here.
4         return -1
5     }
6 }
```

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 class Program {
4     // O(log(n)) time | O(1) space
5     func shiftedBinarySearch(_ array: [Int], _ target: Int) -> Int {
6         var leftPointer = 0
7         var rightPointer = array.count - 1
8
9         return shiftedBinarySearchHelper(array, target, &leftPointer,
10     }
11
12     func shiftedBinarySearchHelper(_ array: [Int], _ target: Int, _ left: Int, _ right: Int) -> Int {
13         while leftPointer <= rightPointer {
14             let middle = (leftPointer + rightPointer) / 2
15             let potentialMatch = array[middle]
16             let leftNumber = array[leftPointer]
17             let rightNumber = array[rightPointer]
18
19             if target == potentialMatch {
20                 return middle
21             } else if leftNumber < potentialMatch {
22                 if target < potentialMatch, target >= leftNumber {
23                     rightPointer = middle - 1
24                 } else {
25                     leftPointer = middle + 1
26                 }
27             } else {
28                 if target <= rightNumber, target > potentialMatch {
29                     leftPointer = middle + 1
30                 } else {
31                     rightPointer = middle - 1
32                 }
33             }
34         }
35     }
36 }
```

```

18         self.eventQueue.put(self.program.waitForResponse(200, 10, 10))
19     }
20     self.testQueue.put(1) if (self.timeout == None) {
21         self.eventQueue.put(self.program.waitForResponse(200, 200,
22     }
23     self.testQueue.put(2) if (self.timeout == None) {
24         self.eventQueue.put(self.program.waitForResponse(200, 40, 10)
25     }
26     self.testQueue.put(3) if (self.timeout == None) {
27         self.eventQueue.put(self.program.waitForResponse(200, 70, 10)
28     }
29     self.testQueue.put(4) if (self.timeout == None) {
30         self.eventQueue.put(self.program.waitForResponse(200, 70, 10)
31     }
32     self.testQueue.put(5) if (self.timeout == None) {

```

Run or submit code when you're ready.