**Search in a Rotated Sorted Array**

You are given a sorted array which is rotated at some random pivot point.

Example: [0,1,2,4,5,6,7] might become [4,5,6,7,0,1,2]

You are given a target value to search. If found in the array return its index, otherwise return -1.

You can assume there are no duplicates in the array and your algorithm's runtime complexity must be in the order of O(log n).

Example:

Input: nums = [4,5,6,7,0,1,2], target = 0, Output: 4

Here is some boilerplate code and test cases to start with:

**def** **rotated\_array\_search**(input\_list, number):

"""

Find the index by searching in a rotated sorted array

Args:

input\_list(array), number(int): Input array to search and the target

Returns:

int: Index or -1

"""

**pass**

**def** **linear\_search**(input\_list, number):

**for** index, element **in** enumerate(input\_list):

**if** element == number:

**return** index

**return** -1

**def** **test\_function**(test\_case):

input\_list = test\_case[0]

number = test\_case[1]

**if** linear\_search(input\_list, number) == rotated\_array\_search(input\_list, number):

print("Pass")

**else**:

print("Fail")

test\_function([[6, 7, 8, 9, 10, 1, 2, 3, 4], 6])

test\_function([[6, 7, 8, 9, 10, 1, 2, 3, 4], 1])

test\_function([[6, 7, 8, 1, 2, 3, 4], 8])

test\_function([[6, 7, 8, 1, 2, 3, 4], 1])

test\_function([[6, 7, 8, 1, 2, 3, 4], 10])