VINCENT MUTETHIA

SCT211-0017/2019

ABSTRACT DATA TYPES AND ALGORITHM

ICS 2300

ASSIGNMENT 1

Question 1: Search in Rotated Sorted Array

```
from typing import List

class Solution:
def search(self, nums: List[int], target: int) -> int:
l, r = 0, len(nums) - 1
while l <= r:
mid = (l + r) // 2
if target == nums[mid]:
return mid
# Left sorted portion of the array
if nums[l] <= nums[mid]:
if target > nums[mid] or target < nums[l]:
l = mid + 1
else:
r = mid - 1
# Right sorted portion of the array
else:
if target < nums[mid] or target > nums[r]:
r = mid - 1
else:
l = mid + 1
return -1 # Return -1 if nothing found


class Solution:
def search(self, nums: List[int], target: int) -> int:
l,r = 0, len(nums) -1
while l <= r:
mid = (l+r)//2
```

```
if target == nums[mid]:
return mid
#left sorted portion of the array
if nums[l] <= nums[mid]:
if target > nums[mid] or target < nums[l]:
l = mid + 1
else:
r = mid - 1
#right sorted portion of the array
else:
if target < nums[mid] or target > nums[r]:
r = mid -1
else:
l = mid + 1
#else return -1 if nothing found
# Create an instance of the Solution class
solution = Solution()

# Test case: a rotated sorted array
nums = [4, 5, 6, 7, 0, 1, 2]
target = 0

# Call the search method
result = solution.search(nums, target)

# Check the result
if result != -1:
print(f"Target {target} found at index {result}")
else:
print("Target not found in the array")
```

Question 2: Kth Largest Element in Array

```
from typing import List

class Solution:
def findkthLargest(self, nums: List[int] , k:int) -> int:
k = len(nums) -k
def quickSelect(l, r):
pivot, p = nums[r], l
for i in range(l,r):
```

```
if nums[i] <= pivot:
nums[p], nums[i] = nums[i], nums[p]
p += 1
nums[p], nums[r] = nums[r], nums[p]
if p > k:
return quickSelect(l , p -1)
elif p < k:
return quickSelect(p + 1, r)
else:
return nums[p]
return quickSelect(0 , len(nums) - 1)
# Create an instance of the Solution class
solution = Solution()

# Test case: an array and k value
nums = [3,2,3,1,2,4,5,5,6]
k = 4

# Call the findkthLargest method
result = solution.findkthLargest(nums, k)

# Check the result
print(f"The {k}th largest element is: {result}")
```

Question 3: Two Sum II _ Input Array is sorted.

```
from typing import List

class Solution:
def twoSum(self, numbers: List[int], target: int) -> List[int]:
l, r = 0, len(numbers) - 1
while l < r:
curSum = numbers[l] + numbers[r]
if curSum > target:
r -= 1
elif curSum < target:
l += 1
else:
return [l + 1, r + 1]
return []


# Create an instance of the Solution class
solution = Solution()

# Test case: a sorted array and a target value
numbers = [2, 7, 11, 15]
target = 9

# Call the twoSum method
result = solution.twoSum(numbers, target)

# Check the result
if result:
print(f"Indices of the two numbers that add up to {target}: {result[0]},
{result[1]}")
else:
print("No solution found.")
```

Question 4: Valid Palindrome

```
class Solution:
def Palindrome(self, s:str) -> bool:
newStr = ""
for c in s:
if c.isalnum():
newStr += c.lower()
# -> [::-1] is the syntax for reversing a string in python
return newStr == newStr[::-1]

# Create an instance of the Solution class
solution = Solution()

# Test case: a string to check for palindrome
s = "A man, a plan, a canal, Panama"

# Call the Palindrome method
result = solution.Palindrome(s)
```

```python
# Check the result
if result:
print(f"'{s}' is a palindrome.")
else:
print(f"'{s}' is not a palindrome.")
```