forEach(), map(), filter() and reduce()

sort() ==> ascending order OR descending order

[1, 2, -3, 4, 1, 3, 4, -3, 5]
k = 3
Find the max sum of the continuos subarray of size k
sub string

Sliding Window ==> patterns

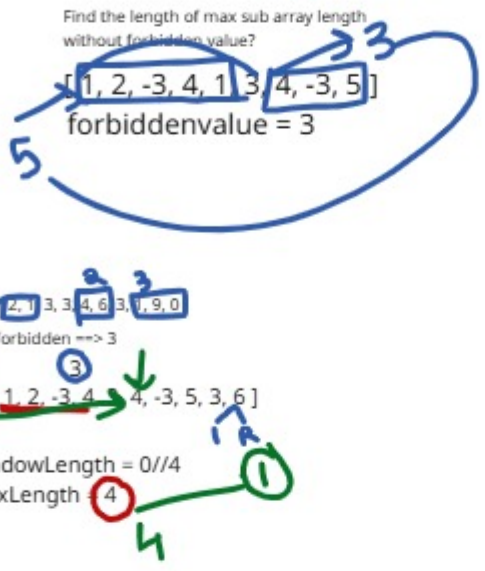
[1, 2, -3, 4, 1, 3, 4, -3, 5]

1, 2, -3 ==> 0
2, -3, 4 ==> 3
.....

[1, 2, -3, 4, 1, 3, 4, -3, 5]

left right
windowSum
maxSum

Find the length of max sub array length
without forbidden value?



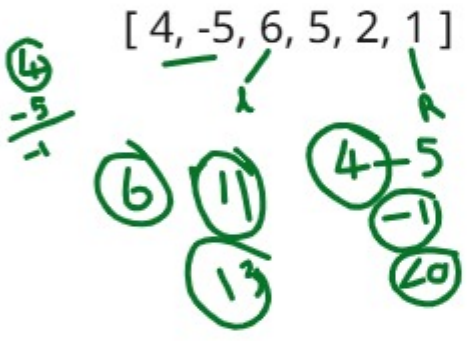
=, ==

5 = "5" ==> T
5 == "5" ==> F

[4, -5, 6]
find the maximum sum of sub array

[4, -5, 6] ==> k = 2 ==> 1
[4, -5, 6] ==> k = 3 ==> 5

[4, -5, 6, 5, 2, 1]



Longest subarray with k unique/distinct characters

[1, 2, 1, 2, 3] k = 2

[1, 2] ==> 2
[1, 2, 1] ==> 3
[1, 2, 1, 2] ==> 4

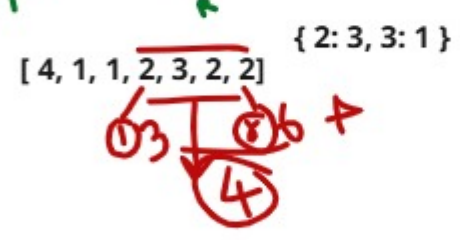
[4, 1, 1, 2, 3, 2, 2] k = 2
4, 1 ==> 2 ==> 2
4, 1, 1 ==> 2 ==> 3
[2, 3, 2, 2] ==> 4

Map ==> dictionary(key value pairs)

k = 2

Map ==> { 1: 2, 2: 2 }
Map ==> { 1: 2, 2: 2, 3: 1 }

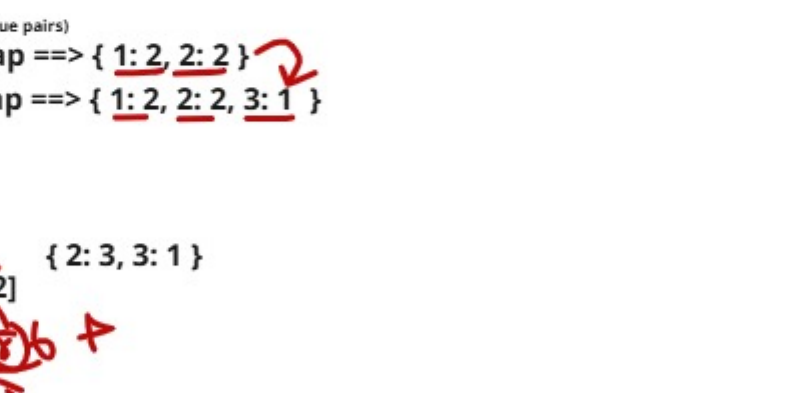
[1, 2, 1, 2, 3]



{ 4: 10 }
get(4) ==> 10 + 1

Input: nums = [1,5,4,2,9,9,9], k = 3
Output: 15

[1, 5, 4] — 10
[5, 4, 2] — 11
[4, 2, 9] — 15
~~[2, 9, 9]~~
~~[9, 9, 9]~~



4 5 6 ==> 4:1, 5:1, 6:1
4 5 4 ==> 4:2, 5:1
4 4 4 ==> 4:3