4/3/2024

Day 7 of DSA

Tasks:-

Checkbox	Hashing
☑	Linear Probing : Handling collision using Linear probing.
☑	Quadratic probing : Handling collision using quadratic probing.
☑	Implement hash table : collision handling Using linear probing :-
☑	Implement hash table collision handling u

* Open Addressing :-

It is method of Collision handling.

① Linear Probing :-

rehash(key) = (n+1) % table-size

Eg. 50, 70, 76, 93

key mod 5

0	50
1	70
2	76
3	93
4	

50 % 5 = 0
70 % 5 = 0
76 % 5 = 1
93 % 5 = 3

* Quadratic probing :-

$hash(x) = (hash(x) + i^2) \% S$

↑
Table size

22, 30 & 50, S = 7

0	
1	22
2	30
3	
4	
5	50
6	

22 % 7 = 1
30 % 7 = 2
50 % 7 = 1

$50 \% 7 + i^2$
$50 \% 7 + 1^2$
1 + 1 = 2

$1 + 2^2$
$\boxed{1 + 4 = 5}$

PAGE No.
DATE / /

credit goes to
pickup from # striver Bhaiya

① Two Sum :-

2	6	5	8	11
0	1	2	3	4

target = 14.

Approach :-

① First creating hash table.

② Every time we subtract curr. element & from target.
& check it in Hash Table
if we found it
then we return curr & that element's index.

③ or if not then we simply insert it in Hash Table

Time Complexity : O(N)
Space Complexity : O(N)

Time Complexity : O(N) + O(N) + O(N) = O(3N)
Space Complexity: O(N)

② Longest Consecutive Sequence :-

5	4	3	2	1

1
2
3
4
5

Approach:-

① Traverse whole array put it into the hashset.

② Then again check that
if curr element previous ele
is present or not
if present then
we do nothing.

③ But if not present then we
search for next element if
it it is present then we
Search until it will not exhaust.

5	4	3	2	1

1
2
3
4
5

1 → 2 ✓
2 → 3 ✓
3 → 4 ✓
4 → 5 ✓
5 → 6 ✗

5 → 4 ✗ - do Nothing
4 → 3 ✗
3 → 2 ✗
2 → 1 ✗
1 → 0 ✓ Do something

get maxlength &
print it.

PAGE No.
DATE / /

③ Largest subarray with sum 0 :-

1, -1, 3, 2, -2, -8, 1, 7, 10, 23

Approach :-

① First declaring two variables sum & max = 0;

② Also unordered_map.

③ traversing the array. and add it to the index with adding it to sum.

④ If at some point we got sum as 0.

then we ~~g~~ can subtract that curr index & hash set presented element index because of that ele we get sum as 0

⑤ And we update max

⑥ & repeat this steps until array get whole traversed.

Time complexity = O(N logN)
space complexity = O(N)

PAGE No.
DATE / /

* Count subarrays with Xor as k.

[4, 2, 2, 6, 4]

Xor = 0 cnt = 0.

$\boxed{Y = XR \wedge K}$

Approach:

① Traversing the array First
we do the Xor operation
& whatever we get we put into
unordered map.

② when we get 6 that time
we put it into set
& check $Y = XR \wedge K$
if it is get 0.
then we increase cnt.

③ Continue this process until Array end.

Time complexity: O(NlogN)
Space complexity: O(N)