Welcome ☺

Reattempt 1 not reflected.

3 hrs.

5th Aug 12:01 am → 9 days

Agenda : Hashing 2

① Pair sum = K

② distinct elements in every window of len = K

Q Given N array elements, check if there exists a pair (i, j) such that arr[i] + arr[j] = K & i != j

	0	1	2	3	4	5	6	7	8	9
arr[] :	8	9	1	-2	4	5	11	-6	7	5

	i	j	
K = 11	4	8	arr[4] + arr[8] ⇒ 4 + 7 ⇒ 11
K = 6	2	5	arr[2] + arr[5] ⇒ 1 + 5 ⇒ 6
K = 22	no such pair. return false.		

Ideas

Idea 1   Check every pair sum = K.

$T.C \rightarrow O(N^2)$ $S.C \rightarrow O(1)$

Pseudo code

TODO

Idea 2

	0	1	2	3	4	5	6	7	8	9
arr[ ] :	8	9	1	-2	4	5	11	-6	7	5

Optimisation using hashset

Insert all elements in hashset.

K = 11

a+b = 11

a	b→(K-a)	Yes/No	
8	3	✗	
9	2	✗	
1	10	✗	
-2	13	✗	
4	7	✓	return True.

a+b = 5

a	b	Yes/No	
8	-3	✗	
9	-4	✗	
1	4	✓	return True.

	0	1	2	3	4	5	6	7	8	9
arr[ ] :	8	9	1	-2	4	5	11	-6	7	5

a+b = 22 Edge case.

a	b	Yes/No	
8	14	✗	
9	13	✗	
1	21	✗	
-2	24	✗	
4	18	✗	
5	17	✗	
11	11	✓	return True but i = = j.

Therefore this case not valid.

Idea 3 Optimisation using hashmap <int, int>

↓ arr[i] ↓ freq.

	0	1	2	3	4	5	6	7	8	9
arr[ ] :	8	9	4	-2	4	5	11	-6	7	5

K = 10

① insert all the elements in the map.

<8, 1> <9, 1> <4, 2> <-2, 1> ,

<5, 2> <11, 1> <-6, 1> <7, 1> ~~<5, 1>~~

a + b = 10

a	b (K-a)	Yes / No
8	2	✗
9	1	✗
4	6	✗
-2	12	✗
4	6	✗
5	5	Yes a=b , freq[b] > 1

K = 22

a	b	Yes / No .
8	14	✗
9	13	✗
4	18	✗
-2	24	✗
4	18	✗
5	17	✗
11	11	NO b/c freq of 11 is 1

Pseudo code

```
bool pairSum ( int a[] , int K )
{
    hashmap <int , int > hm ;
    ① Insert all the elements in hashmap
    for( int i = 0 ; i < n ; i++ )
    {
        a = arr[i]          b = K - a
        if( a != b  &&  hm.search(b) == true )
                        return true;
        else if ( a == b  &&  hm[a] > 1 )
                        return true;
    }
    return false;
}
```

Idea 4 Optimisation using hashset

	0	1	2	3	4	5	6	7	8	9
arr [ ] :	8	9	1	-2	4	5	11	7	5	-6

idea ⇒ if we are at index i , insert only [0, i-1] elements in hashset.

K = 11

a	b	H.S	Yes / No.
8	3	{ }	✗
		insert 8	
9	2	{8}	✗
		insert 9	
1	10	{ 8 , 9 }	✗

insert 1

-2   13   {8, 9, 1}   ✗

insert -2

4   7   {8, 9, 1, -2}   ✗

insert 4   ✗

5   6   {8, 9, 1, -2, 4}

insert 5

11   0   {8, 9, 1, -2, 4, 5}   ✗

insert 11

7   4   {8, 9, 1, -2, 4, 5, 11}   ✓ return true

Pseudo code

```
bool    targetSumSet( int arr[] , int K)
{
    int n = arr.length;
    hashset <int>  hs;
    for( int i=0 ;  i<n ;  i++)
    {
        a = arr[i]        b = K-a
        if( hs.search(b) == true)   return true;
        hs.insert (a)
    }
    return false.
}
```

T.C → O(N)

S.C → O(N)

Q2 Given N elements, calculate no. of distinct elements in every subarray of size K

eg: a[] :

0	1	2	3	4	5	6	7	8	9
2	4	3	8	3	9	4	9	4	10

K = 4

subarrays	print
[0, 3]	4
[1, 4]	3
[2, 5]	3
[3, 6]	4
[4, 7]	3
[5, 8]	~~3~~2
[6, 9]	3

Idea 1

→ For every subarray of len = K, insert into hashset and get no. of distinct elements.

$$T.C \Rightarrow \underbrace{(n-K+1)}_{\substack{\#\text{subarray}\\ \text{of size } K.}} * \underbrace{K}_{\text{subarray len}}$$

$$K = N/2 \Rightarrow \left(N - \frac{N}{2} + 1\right) * \left(\frac{N}{2}\right)$$

$$\Rightarrow O(N^2)$$

$$S.C \Rightarrow O(K)$$

Idea 2 Optimisation using hashset.

eg: a[] :

0	1	2	3	4	5	6	7	8	9
2	4	3	8	3	9	4	9	4	10

K=4

[0, 3] ⟶ { 2, 4, 3, 8 } ⇒ 4

	delete	add		
[1, 4]	arr[0]	arr[4]	H.S → { 4, 3, 8 }	⇒ 3
[2, 5]	arr[1]	arr[5]	H.S ⇒ { 3, 8, 9 }	⇒ 3
[3, 6]	arr[2]	arr[6]	H.S ⇒ { ~~8, 9, 4~~ }	⇒ ~~3~~ 4 ✓

→ In hashset, deleting element indirectly deletes all occurences

Idea 3   Optimisation using hashmap.

eg: a[] :

0	1	2	3	4	5	6	7	8	9
2	4	3	8	3	9	4	9	4	10

K=4

arr[0,3] ⟶ { <2,1> , <4,1> <3,1> ,<8,1>} 4

delete   add

since freq = 0, remove from hashmap.

arr[1,4]   arr[0]   arr[4] ⇒ {<2,0>, <4,1> , ~~<3,1>~~ , <8,1>
⇓
<3,2> } 3

arr[2,5]   arr[1]   arr[5] ⇒ { ~~<4,0>~~, <8,1>, <3,2>
<9,1>} 3

Pseudo code

```
void   distinctElements( int arr[], int K )
{
    int n = arr.len
    hashmap < int, int > hm ;
    // insert K elements in hm -> [0, K-1]
    for( int i = 0 ; i<K ; i++)  ---> T.C = O(K)
    {
        if( hm.search( arr[i] ) == true )
                  hm[arr[i]] += 1 ;
        else
              hm.insert( arr[i], 1 ) ;
    }
    print( hm.size() );
    s = 1        e = K
    while ( e < N )  ---> T.C = O(N-K)
    {
        // sub [s, e] , remove s-1 , add e
        hm[arr[s-1]] -= 1 ; // reduced freq by 1
        if ( hm[arr[s-1]] == 0 )
        {
            // remove element from hash map
            hm.remove ( arr[s-1] ) ;
        }
        if ( hm.search ( arr[e] ) == true)
        { // increase freq by 1
             hm[arr[e]] += 1 ;
        }
        else{
             hm.insert ( arr[e], 1);
        }

        s++;
        e++;
        print ( hm.size() );
    }
}
```

final T.C
$K + (N-K)$

T.C ⇒ $O(N)$

S.C ⇒ $O(K)$

~~$O(N)$~~

Doubt session

Contest 1 Reattempt 2,3 ⇒ next week.

Contest 2 Reattempt 2 ⇒ live on 5th August 12:01 am for 9 days.

[2, 3, 1]

Count sort

~~Count~~ Sorted arr

a a a bb d f a a bc

a → 4

b → 2

d → 4

a a a a bb dd dd