Agenda

1) Distinct numbers in window

2) No. of distinct 2D points

3) Class object as key

Q.1 Given an array, calculate no. of distinct elements in every subarray of size K.

	0	1	2	3	4	5	6	7
A:	2	4	3	8	3	9	4	9

K=4

0 to 3 → 4

1 to 4 → 3

2 to 5 → 3

3 to 6 → 4

4 to 7 → 3

subarray of len K in n length array = n-K+1

idea: use sliding window technique with hashset.

	0	1	2	3	4	5	6	7
A:	2	4	3	8	3	9	4	9

K=4

last window ans

↓

remove the impact of A[s-1] and

odd the impact A[e]

	0	1	2	3	4	5	6	7
A :	2	4	3	8	3	9	4	9

k = 4

s	e	remove	add	HashSet	ans
0	3			2 4 3 8	4
1	4	A[0] (2)	A[4] (3)	4 3 8	3
2	5	A[1] (4)	A[5] (9)	3 8 9	3
3	6	A[2] (3)	A[6] (4)	4 8 9	3 ✗

	s			e
0	1	2	3	4
~~3~~	2	5	3	7

k = 4

3 is wrong
ans. should be 4 } 1 to 4 window

~~3~~ 2
5 7

Let's try hashmap

	s			e
0	1	2	3	4
~~3~~	2	5	3	7

3 → ~~2~~ 1
2 → 1
5 → 1
7 → 1

A :	0	1	2	3	4	5	6	7
	2	4	3	8	3	9	4	9

K = 4

s	e	remove	add	HashMap	ans
0	3			2 → 1 4 → 1 3 → 1 8 → 1	4
1	4	A[0] = 2	A[4] = 3	4 → 1 3 → 2 8 → 1	3
2	5	A[1] = 4	A[5] = 9	3 → 2 8 → 1 9 → 1	3
3	6	A[2] = 3	A[6] = 4	3 → 1 8 → 1 9 → 1 4 → 1	4
4	7	A[3] = 8	A[7] = 9	3 → 1 9 → 2 4 → 1	3

```
void solve (int [] A, int k) {
    1) calculate ans of 1st window (0 to k-1)

    // apply sliding window technique on rest of windows

     s=1, e=k;

     while(e<n) {
     |      // remove the impact of A[s-1] in map
     |
     |      // add the impact of A[e] in map
     |
     |      sopln(map.size());
     |
     |      s++; e++;
     }
}
```

TC: O(n)

SC: O(n)

code: IDE

Q-2 Given a 2D array denoting points on a 2D plane. (x,y)
Return total no. of distinct points in the array.

```
A = { {5,6},
      {2,8},
      {-1,-1},
      {2,-3},
      {2,8},
      {7,7},
      {2,8},
      {2,-3}
    };
```

Distinct points : 5

(5,6) (2,8) (-1,-1)

(2,-3) (7,7)

i^{th} coordinate : $x = A[i][0]$

$y = A[i][1]$

all Coordinate

```
{ x -> A[i][0]
  y -> A[i][1]

  String str = x + "#" + y;
  hs.add(str);
```

$$A = \begin{array}{c|c|c} & 0 & 1 \\ \hline 0 & 2 & 8 \\ 1 & 5 & 3 \\ 2 & 8 & 2 \\ 3 & 2 & 8 \\ 4 & -1 & 3 \end{array}$$

i	x	y	str
0	2	8	"2#8"
1	5	3	"5#3"
2	8	2	"8#2"
3	2	8	"2#8"
4	-1	3	"-1#3"

"2#8"
"5#3"
"8#2"
"-1#3"

hs

ans: hs.size()

code: IDE

TC: O(n)

SC: O(n)

Object as Key in Hashing

int

→ Every distinct key has a hashcode, hashcode of a key is used in the backend implementation of hashset / hashmap.