Q1. /* count distinct elements */

a = [5, 10, 15, 5, 4, 5]

ans = 4

→ Brute force sol^{n}.

```
int countDistElement ( int a[]) {
        int count = 0;
    for(int i = 0;  i<=count ; i++) {
          ~~if (a[i]~~
```

→ Optimised approach.

```
int countDisElement(int [] 'a) {
    set <Integer> set = new HashSet <>();
    for (int element : a) {
          set.add (element) }.
    }
   return  set.size ();
}
```

Q2 /* union of two arrays */

```
a[] = {5, 10, 15, 5}
b[] = {10, 15, 4, 5}

ans = 4
```

```
⇒ int union (int a[], int b[]){
      Set<Integer> set = new HashSet<>(

      for(int x : a){
          set.add(x);
      }
      for(int x: b){
          set.add(x);
      }
      return set.size();
  }
```

Q3. Intersection of two arrays */

```
a[] = {5, 10, 15, 5, 10}
b[] = {15, 10, 4}
ans = 2
```

```
int intersect (int a[], int b[]) {
    Set<Integer> set = new HashSet
    int count = 0;
    for (int x : a) {
        set.add(x);
    }
    for (int x : b) {
        if (set.contains(x)) {
            count++;
            set.remove(x);
        }
    }
    return count;
}
```

(Hashmap)

Q1: find the subArray with the given sum

a = [10, 15, -5, 15, -10, 5]

sum = 5

~~20~~

10 → 0

25 → 1

20 → 2

35 → 3

→ start = 3

end = 4

code

```
void subArraySum(int a[], int sum){
    int currSum = 0;
    int start = 0;
    int end = -1;

    HashMap<Integer, Integer> map = new
                        HashMap<>();

    for(int i=0; i<n; i++){
        currSum += a[i];
        if(currSum-sum==0){
            start=0; end=i;
            break;
        }
        if(map.containsKey(currSum-sum)){
            start = map.get(currSum -sum)+1;
            end = i; break;
        }
        map.put(currSum, i);
    }

    if(end == -1){
        Sysout("Not found");
    }
    else{
        Sysout(Start+" "+end);
    }
```

Q5 count the distinct elements in every window of size k.

a = [1, 2, 2, 1, 3, 1, 1, 3]

k = 4

output: 2
3
3
2
2

code:

```
void countDistinctElement (int a[], int k) {
    Map<Integer, Integeu> = new HashMap<>();

    for(int i=0; i<k; i++){

        map.put (a[i], map.getOrDefault
                        (a[i], 0) + 1);
    }
}
    ~~return Map.size()~~;
    Sysout(map.size()
    for (int i=k; i<a.length; i++){
        if (~~a[i~~ map.get(a[i-k] == 1)){
                map.remove(a[i-k]);
        }

        else{

            map.put(a[i], map.get(a[i-k]-1));
        }
        map.put(a[i], map.getOrDefault(a[i],+1
    Sysout(map.size
```