Qn Link : https://leetcode.com/problems/maximum-sum-of-distinct-subarrays-with-length-k/description/  
Step 1 : Run the for loop upto k and get the sum and put the element in the map.

Step 2 : After k ietarion , if the size of the map s K , then for now , the sum is max.

Step 3 : run the for loop from K to N , for each ietarion, remove the element at index i- k and reduce it occurrence from the map also , remove it value from the sum.

Step 4 : If an occurrence of the element reaches 0 , then remove it from map.

Step 5 : Add the current element to the sum and to the map, if the size of the map is K , then check whether it sum is greater than the existing max

Step 6: If so update it

class Solution {

    public long maximumSubarraySum(int[] nums, int k) {

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

        int n = nums.length;

        long sum = 0;

        long max = 0;

        int left = 0;

        int right = 0;

        for(int i = 0 ; i < k ; i++){

            sum += nums[i];

            map.put(nums[i] , map.getOrDefault(nums[i] , 0) + 1);

        }

        if(map.size() == k){

            max = sum;

        }

        for(int i = k ; i < n ; i++){

            if(i-k >= 0){

                sum -= nums[i - k];

                map.put(nums[i - k] , map.getOrDefault(nums[i - k] , 0) - 1);

                if(map.get(nums[i - k]) == 0){

                    map.remove(nums[i - k]);

                }

                sum += nums[i];

                map.put(nums[i] , map.getOrDefault(nums[i] , 0) + 1);

                if(map.size() == k){

                 max = Math.max(max , sum);

                }

            }

        }

        return max;

    }

}