Date: 18.1.2024

Session Topic: Sliding Window

Task 2

Question: Longest Even Odd Subarray With Threshold

Solution:

```
class Solution {
    public int longestAlternatingSubarray(int[] nums, int threshold) {
    int max=0;
    for(int i=0;i<nums.length;i++)
    {
        if(nums[i]%2==0 && nums[i]<=threshold)
        {
            int count=1;
            for(int j=i;j<nums.length-1;j++)
            {
                if(nums[j]%2!=nums[j+1]%2 && nums[j+1]<=threshold)
                {
                    count++;
                }
                else
                {
                    break;
                }
            }
            max=Math.max(max,count);
        }

    }

    return max;

    }



}
```

Date: 18.1.2024

Session Topic: Sliding Window

Task 3

Question: Minimum Size Subarray Sum

Solution:

```
class Solution {

    public int minSubArrayLen(int target, int[] nums) {

        int left=0,sum=0,min=Integer.MAX_VALUE;

        for(int right=0;right<nums.length;right++)

        {

            sum+=nums[right];

            while(sum>=target)

            {

                sum-=nums[left];

                min=Math.min(min,right-left+1);

                left++;

            }

        }

        if(min==Integer.MAX_VALUE)
        {

            return 0;

        }

        return min;

    }

}
```

Date: 18.1.2024

Session Topic: Sliding Window

Task 4

Question: Max Consecutive Ones III

Solution:

```
class Solution {

    public int longestOnes(int[] nums, int k) {

       int max=0;

       int right=0,left=0;

       for(int i=0;i<nums.length;i++)

       {

           int count=k;

           for(int j=i;j<nums.length;j++)

           {

              if(nums[j]==1)

              {

                max=Math.max(max,j-i+1);

              }

              else if(nums[j]==0)
              {
                  if(count>=1)
                  {
                      count--;
                      max=Math.max(max,j-i+1);
                  }
                  else
                  {
                      break;
                  }
              }
           }
        }
       return max;
    }
}
```

Date: 18.1.2024

Session Topic: Sliding Window

Task 5

Question: Count Number of Nice Subarrays

Solution:

```
class Solution {
    public int numberOfSubarrays(int[] nums, int k) {
        if(nums == null || nums.length == 0) return 0;
        int max=0;
        for(int i=0;i<nums.length;i++)
        {
            if(nums[i]%2==0)
            {
                nums[i]=0;
            }
            else
            {
                nums[i]=1;
            }
        }
        int sum=0;
        Map<Integer,Integer>map=new HashMap<>();
        for(int i:nums)
        {
            sum+=i;
            if(sum==k)
            {
                max++;
            }
            map.put(sum,map.getOrDefault(sum,0)+1);
            if(map.containsKey(sum-k))
            {
                max+=map.get(sum-k);
            }
        }
        return max;

    }

}
```

Date: 18.1.2024

Session Topic: Sliding Window

Task 6

Question: Is Subsequence

Solution:

```
class Solution {

    public boolean isSubsequence(String s, String t) {

        int i=0;

        for(int j=0;j<t.length() && i<s.length();j++)

        {

            if(t.charAt(j)==s.charAt(i))

            {

                i++;

            }

        }

        return i==s.length();

    }

}
```

Date: 18.1.2024

Session Topic: Sliding Window

Task 7

Question:Sort Colors

Solution:

```
class Solution {
    public void sortColors(int[] nums) {
        Arrays.sort(nums);
    }
}
```

Date: 18.1.2024

Session Topic: Sliding Window

Task 8

Question: . Reverse Words in a String

Solution:

```
class Solution {
    public String reverseWords(String s) {
        s=s.trim();
        String arr[]=s.split(" ");
        StringBuilder sb = new StringBuilder();
        StringBuilder sb1 = new StringBuilder();
        for(int i=arr.length-1;i>=0;i--)
        {
            sb1.append(arr[i]);
            if(sb1.length()!=0)
            {
                sb.append(arr[i]);
                sb.append(" ");
            }
            sb1.setLength(0);
        }
        return sb.toString().trim();
    }
}
```