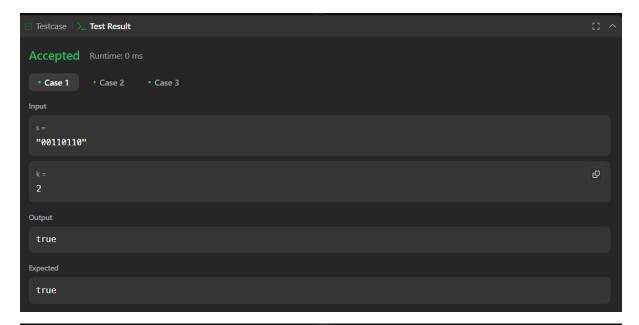
ADVANCED CODING-II **RAVIPATI GANESH** VU21CSEN0300277 CSE-AIML

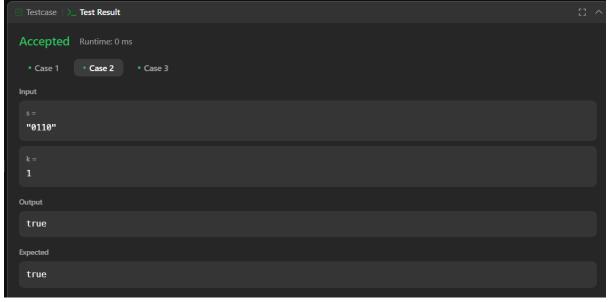
1. Check If a String Contains All Binary Codes of Size K

```
class Solution {
   public boolean hasAllCodes(String s, int k) {
        Set<String> codes = new HashSet<>();
        int total = 1 << k;

        for (int i=0; i+k<=s.length(); i++) {
            codes.add(s.substring(i, i+k));
            if (codes.size() == total) return true;
        }

        return false;
   }
}</pre>
```



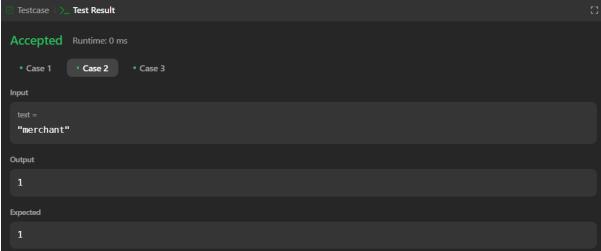




2.Longest Chunked Palindrome Decomposition

```
class Solution {
    public int longestDecomposition(String text) {
        int n = text.length();
        int k = 0, totalLength = 0;
        int str1Start = 0, str1End = 0;
        int str2Start = n-1, str2End = n;
        while (str1End < str2Start) {</pre>
            if (text.substring(str1Start, str1End +
1).equals(text.substring(str2Start, str2End))) {
                totalLength += (str2End - str2Start) * 2;
                k++;
                str1Start = str1End + 1;
                str2End = str2Start;
            str1End++;
            str2Start--;
        if (totalLength < n) return (k * 2) + 1;</pre>
```







3. Constrained Subsequence Sum

```
class Solution {
   public int constrainedSubsetSum(int[] nums, int k) {
      int res = Integer.MIN_VALUE;
      Queue<int[]> maxHeap = new PriorityQueue<>>((a,b) -> Integer.compare(b[1],
a[1]));
```



```
Testcase > Test Result

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

nums = [10,-2,-10,-5,20]

k = 2

Output

23

Expected

23
```

4. Max Value of Equation

```
class Solution {
    public int findMaxValueOfEquation(int[][] points, int k) {
        int ans=Integer.MIN_VALUE;
        int i=0;
        int f=1;
        while(i < points.length) {</pre>
            if(f<i+1)
                f=i+1;
            for (int j = f; j <= points.length-1; j++) {</pre>
                if(points[j][0]>(points[i][0]+k))
                    break;
                if((points[i][1]+points[j][1]+points[j][0]-points[i][0])>ans){
                    ans=points[i][1]+points[j][0]-points[i][0];
                    f=j-1;
            i++;
        return ans;
```



