Roll no. 123103005

Que 1: you are given an empty board with k consecutive cells,at any moment each cell can display one character you want the board to display a string s with length n>k, since board is not larg enough have to display string in n-k+1 steps

calculate power rwequired to display the whole string power required=no.of differing characters in two consecutive display

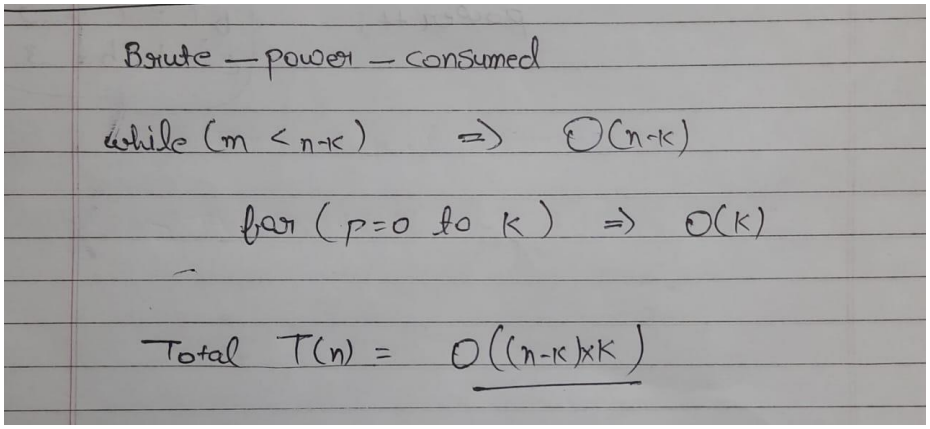
Ans:

Code:

(i)brute force:

```
int Brute_power_consumed(char str[],int n,int k){
        int power=0;
        int m=0;

        while(m<n-k){  // --->O(n-k)
            int i=m;
            int j=m+k;

            for(int p=0;p<k;p++){  // --->O(k)
                if(str[i]!=str[j]){
                power++;
            }
            i++,j++;
            }

            m++;
        }
        return power;    //-->O((n-k)*k)
}
```

Analysis:

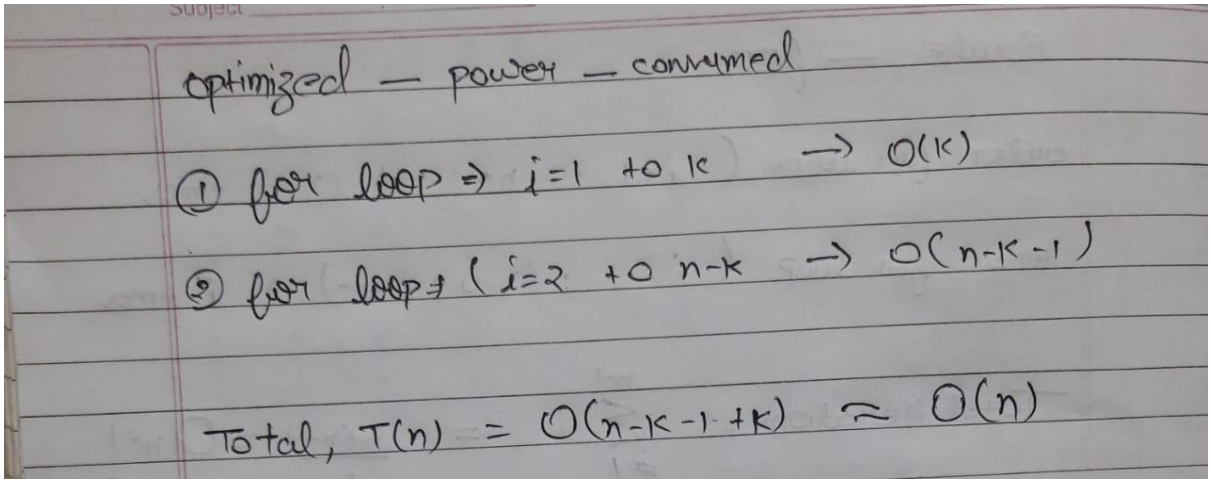


(ii)optimized approach:

Code:

```
int optimized_power_consumed(char str[], int k) {
    int n = strlen(str);
    int power = 0, totalPower = 0;

    for (int i = 1; i <= k; i++) {
        if (str[i] != str[i - 1]) {
            power++;
        }
    }

    totalPower = power;

    for (int i = 2; i <= n - k; i++) {
        if (str[i-1] != str[i - 2]) {
            power--;
        }
        if (str[i + k - 1] != str[i + k - 2]) {
            power++;
        }
        totalPower += power;
    }

    return totalPower;
}
```

Analysis:



Test cases:

String	k(size of board)	power consumed
abcbaccba	3	15
kush	6	0
3005	4	0
Verylargestring	2	26

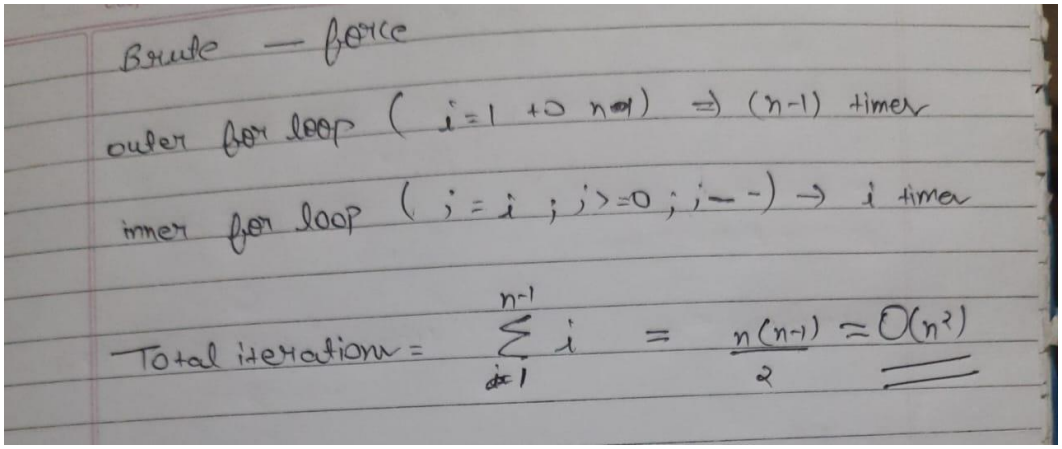
Que2: you are given an array a[i] of stock price on each day find span s[i] for each a[i] the span s[i] of the stock price on a given day i is defined as the maximum number of consecutive days just before the given day for which price of stock on that day is less than or equal to price on the given day,(current day also included).

Ans:

(i)brute force

```
void brute_stock(int A[],int n,int S[]){
        for(int i=1;i<n;i++){   //--->O(n-1)
            for(int j=i;j>=0;j--){  // --->O(i)
                if(A[j]<=A[i]){
                    S[i]++;
                }
                else break;
            }
        }
}
```

Analysis:

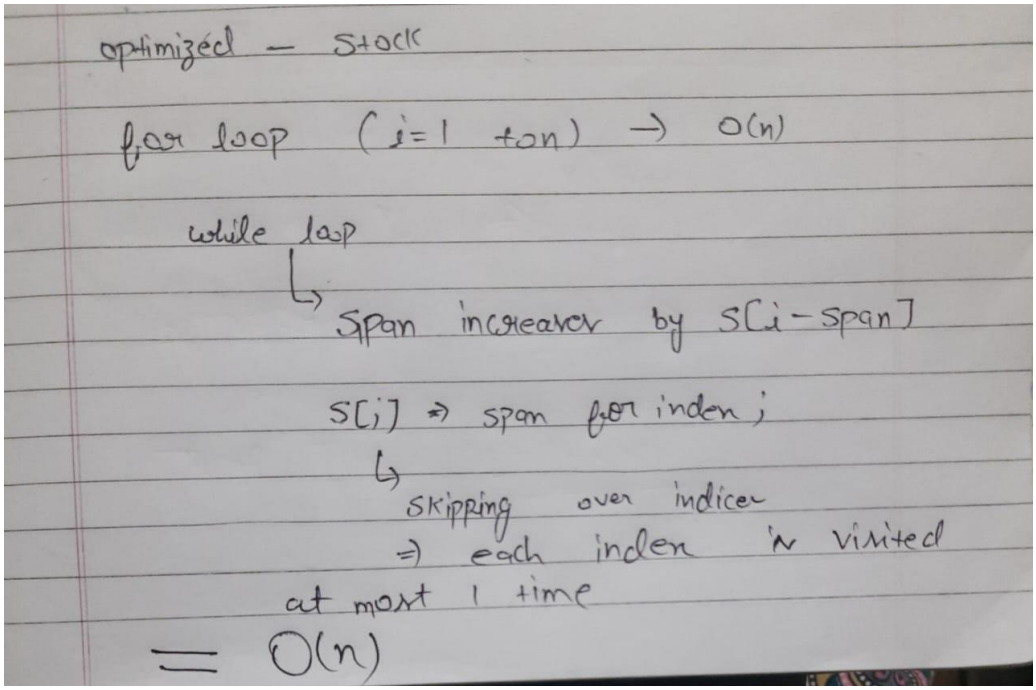


(ii)optimized approach

Code:

```
void optimized_stock(int A[], int n, int S[]) {
    S[0] = 1;

    for (int i = 1; i < n; i++) {
        int span = 1;
        while ((i - span) >= 0 && A[i] >= A[i - span]) {
            span += S[i - span];
        }
        S[i] = span;
    }
}
```

Analysis:



Test cases:

N	A[]	S[](output)
5	40,45,12,35,80	1,2,1,2,5
1	50	1
2	50,50	1,2
3	40,30,20	1,1,1