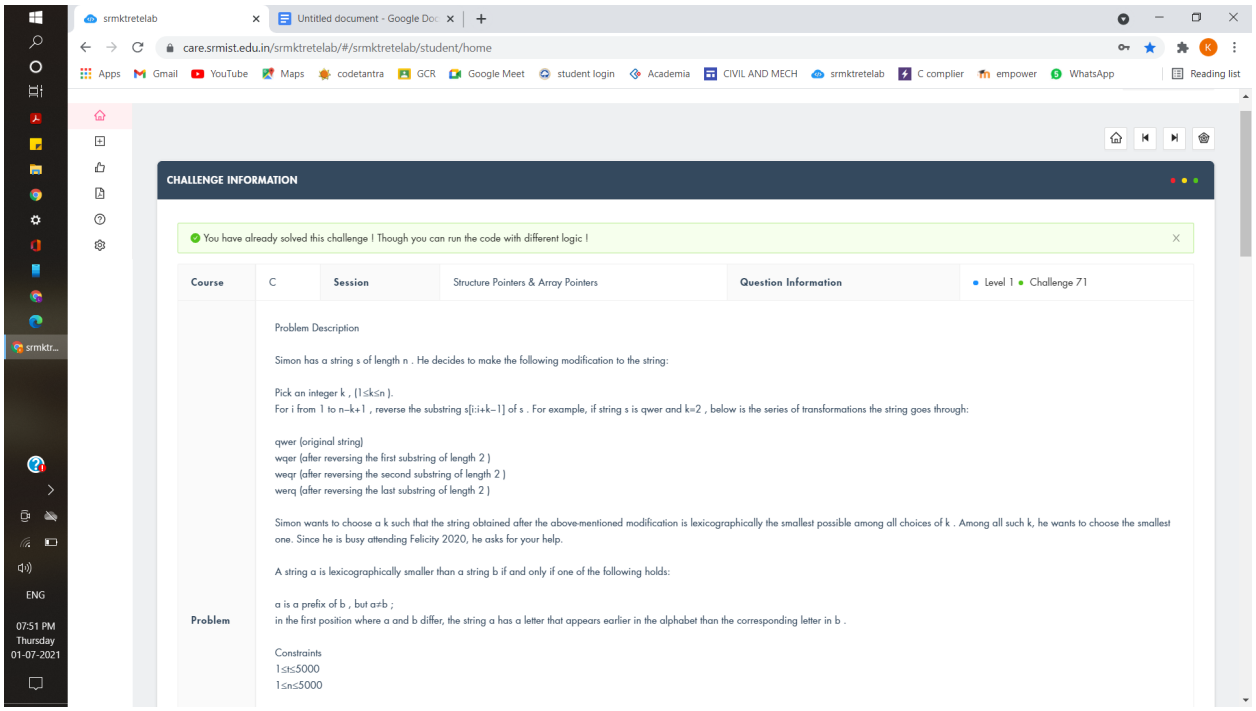
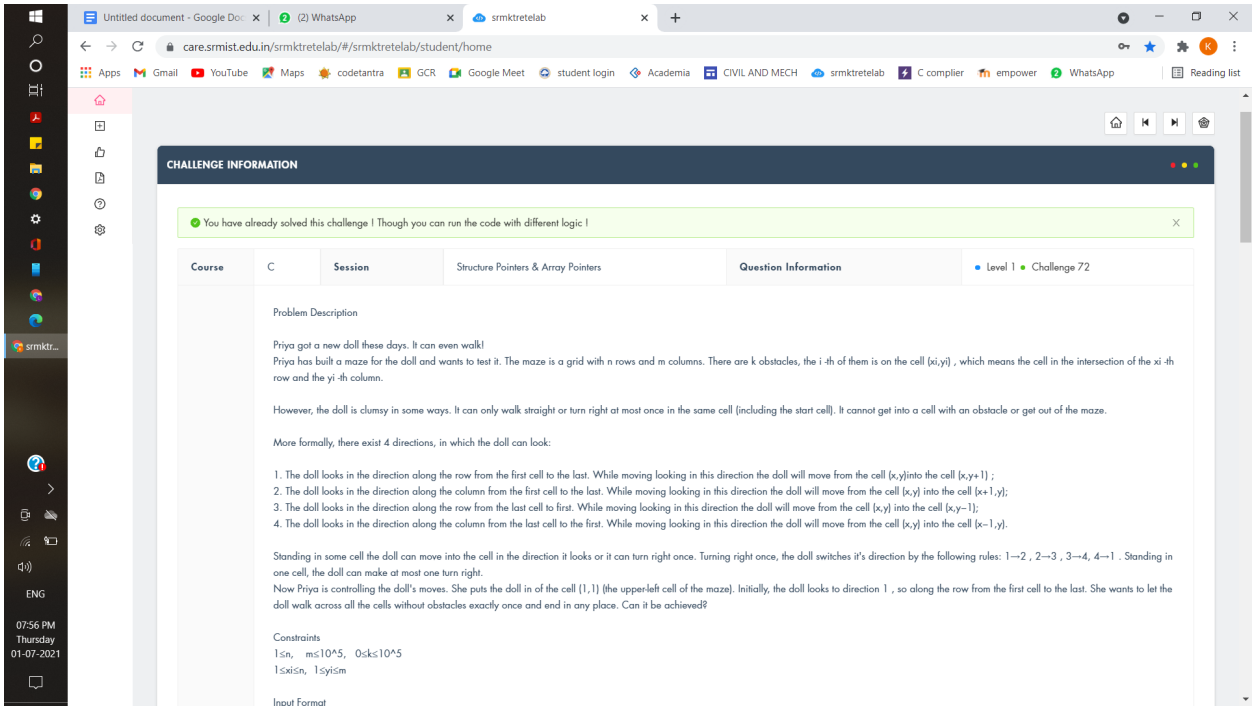
Structure Pointers & Array Pointers LEVEL 1 KV

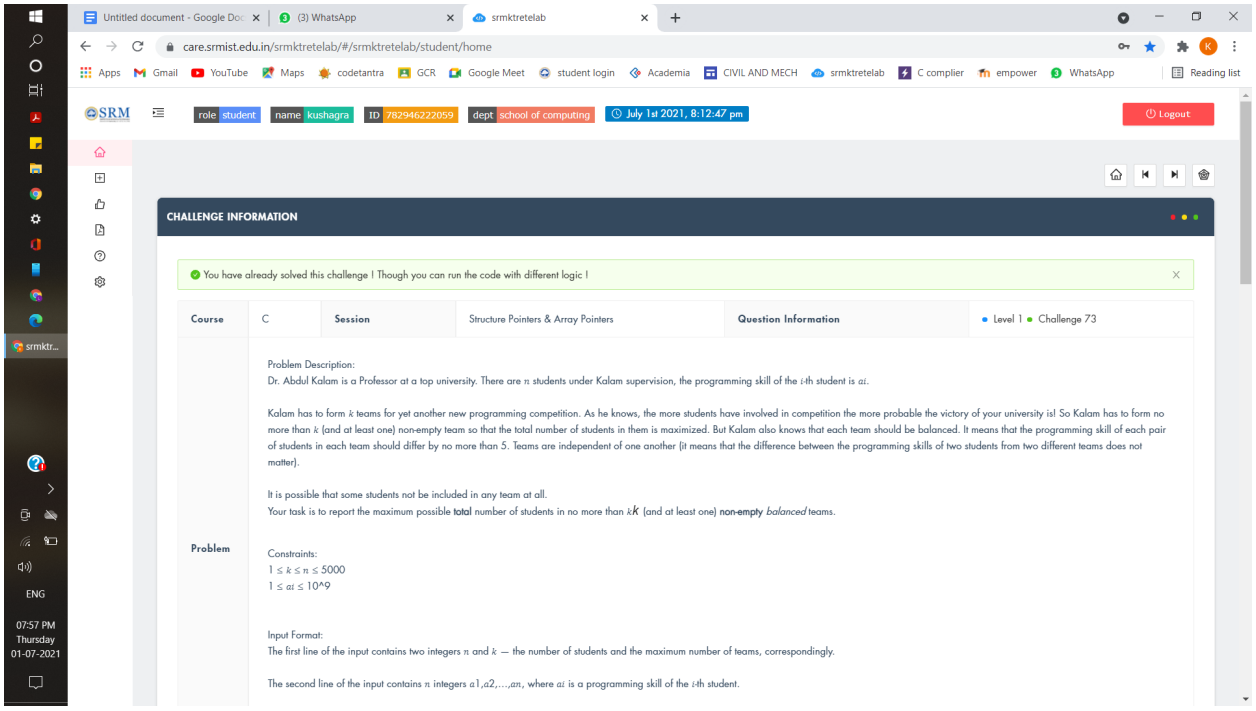



```
#include<stdio.h>
#include<string.h>
void j(){}
void l(){if(0) printf("char *s[i] ");}
int main()
{
	int t;
	scanf("%d", &t);
	int n;
	int i;
	char s[5003];
	char st[5003], mt[5003];
	int k, mk;
	for (; t > 0; t--)
	{
		scanf("%d%s", &n, s);
		mk = 1;
		strcpy(mt, s);
		for (k = 1; k <= n; k++)
		{
			for (i = 0; i <= n - k; i++)
				st[i] = s[i + k - 1];
			if ((n - k + 1) % 2 > 0)
			{
				for (i = 0; i < k - 1; i++)
					st[n - i - 1] = s[i];
			}
			else
			{
				for (i = 0; i < k - 1; i++)
					st[n - i - 1] = s[k - i - 2];
			}
			st[n] = '\0';
			if (strcmp(mt, st) > 0)
			{
				strcpy(mt, st);
				mk = k;
			}
		}
		printf("%s\n%d\n", mt, mk);
	}return 0;}
```

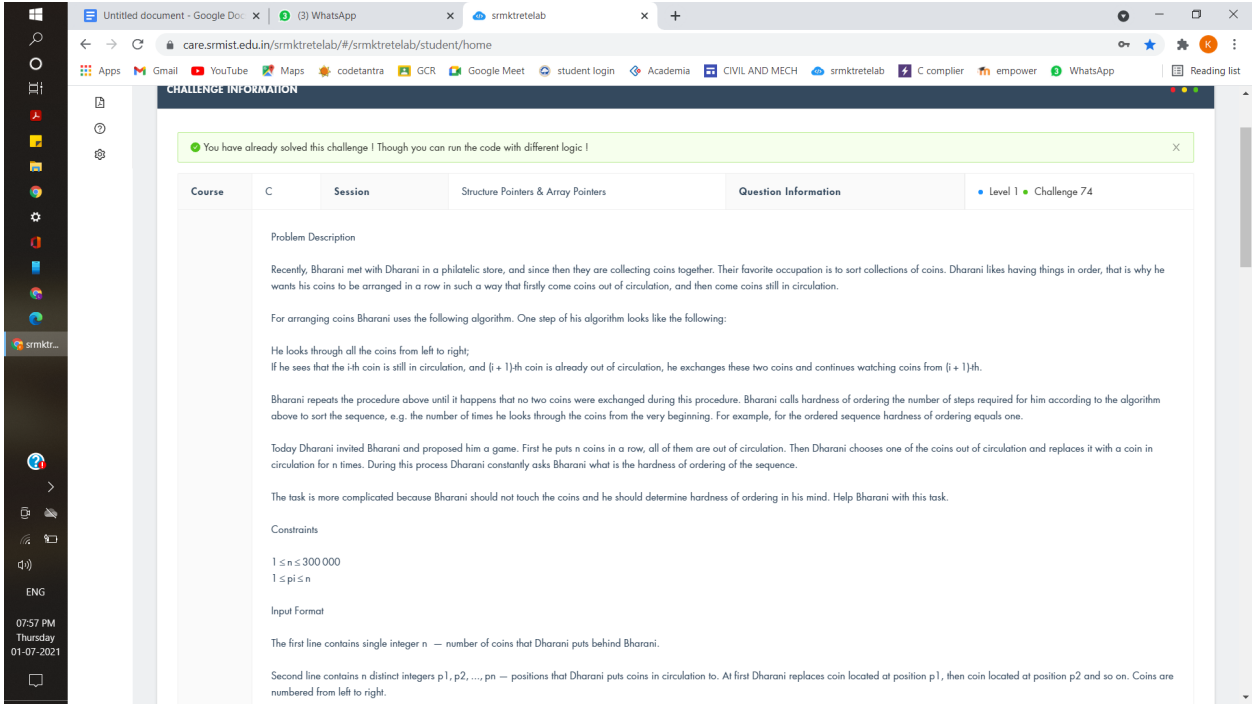



```
#include <stdlib.h>

#define N	100000
#define M	100000
#define K	100000

int min(int a, int b) { return a < b ? a : b; }
int max(int a, int b) { return a > b ? a : b; }

int move(int *aa, int k, int j0, int j1, int incr) {
		int j_, h;

		j_ = -1;
		for (h = 0; h < k; h++) {
				int j = aa[h];

				if (j < j0 || j > j1)
						continue;
				j_ = j_ == -1 ? j : incr ? min(j_, j) : max(j_, j);
		}
		return j_ == -1 ? j1 - j0 + 1 : incr ? j_ - j0 : j1 - j_;
}

int main() {
		static int *aa[N], ka[N], *bb[N], kb[M], ii[K], jj[K];
		int n, m, k, h, i, j, i0, i1, j0, j1, d_;
		long long sum;

		scanf("%d%d%d", &n, &m, &k);
		for (h = 0; h < k; h++) {
				scanf("%d%d", &i, &j), i--, j--;
				ii[h] = i, jj[h] = j;
				ka[i]++, kb[j]++;
		}
		for (i = 0; i < n; i++) {
				aa[i] = malloc(ka[i] * sizeof *aa[i]);
				ka[i] = 0;
		}
		for (j = 0; j < m; j++) {
				bb[j] = malloc(kb[j] * sizeof *bb[j]);
				kb[j] = 0;
		}
		for (h = 0; h < k; h++) {
				i = ii[h], j = jj[h];
				aa[i][ka[i]++] = j;
				bb[j][kb[j]++] = i;
		}
		i0 = 0, i1 = n - 1, j0 = 0, j1 = m - 1, d_ = 1;
		sum = 0;
		while (i0 <= i1 && j0 <= j1) {
				int cnt;

				if (d_ == 1) {
						if ((cnt = move(aa[i0], ka[i0], j0, j1, 1)) == 0)
								break;
						i0++;
						j1 = j0 + cnt - 1;
				} else if (d_ == 2) {
						if ((cnt = move(bb[j1], kb[j1], i0, i1, 1)) == 0)
								break;
						j1--;
						i1 = i0 + cnt - 1;
				} else if (d_ == 3) {
```

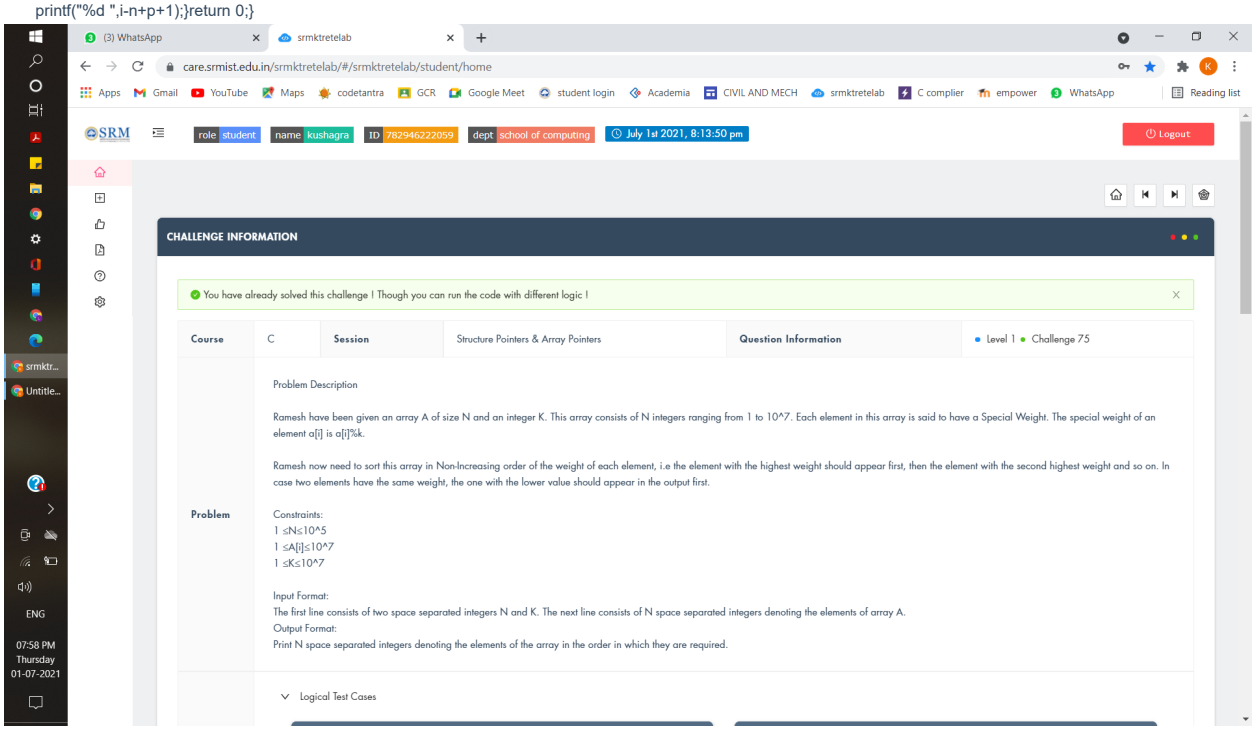
```
                        if ((cnt = move(aa[i1], ka[i1], j0, j1, 0)) == 0)
                                break;
                        i1--;
                        j0 = j1 - cnt + 1;
                } else {
                        if ((cnt = move(bb[j0], kb[j0], i0, i1, 0)) == 0)
                                break;
                        j0++;
                        i0 = i1 - cnt + 1;
                }
                sum += cnt;
                if (d_++ == 4)
                        d_ = 1;
        }
        printf(sum + k == (long long) n * m ? "Yes\n" : "No\n");
        return 0;
}
```




```
#include <stdio.h>
#include <stdlib.h>

#define N       5000

int max(int a, int b) { return a > b ? a : b; }

int compare(const void *a, const void *b) {
        int ia = *(int *) a;
        int ib = *(int *) b;

        return ia - ib;
}

int main() {
        static int aa[N], dp[N + 1][N + 1];
        int n, k, h, i, j;

        scanf("%d%d", &n, &k);
        for (i = 0; i < n; i++)
                scanf("%d", &aa[i]);
        qsort(aa, n, sizeof *aa, compare);
        for (i = 0, j = 1; j <= n; j++) {
                while (aa[i] + 5 < aa[j - 1])
                        i++;
                for (h = 1; h <= k; h++)
                        dp[j][h] = max(dp[j - 1][h], dp[i][h - 1] + j - i);
        }
        printf("%d\n", dp[n][k]);
        return 0;
}
```

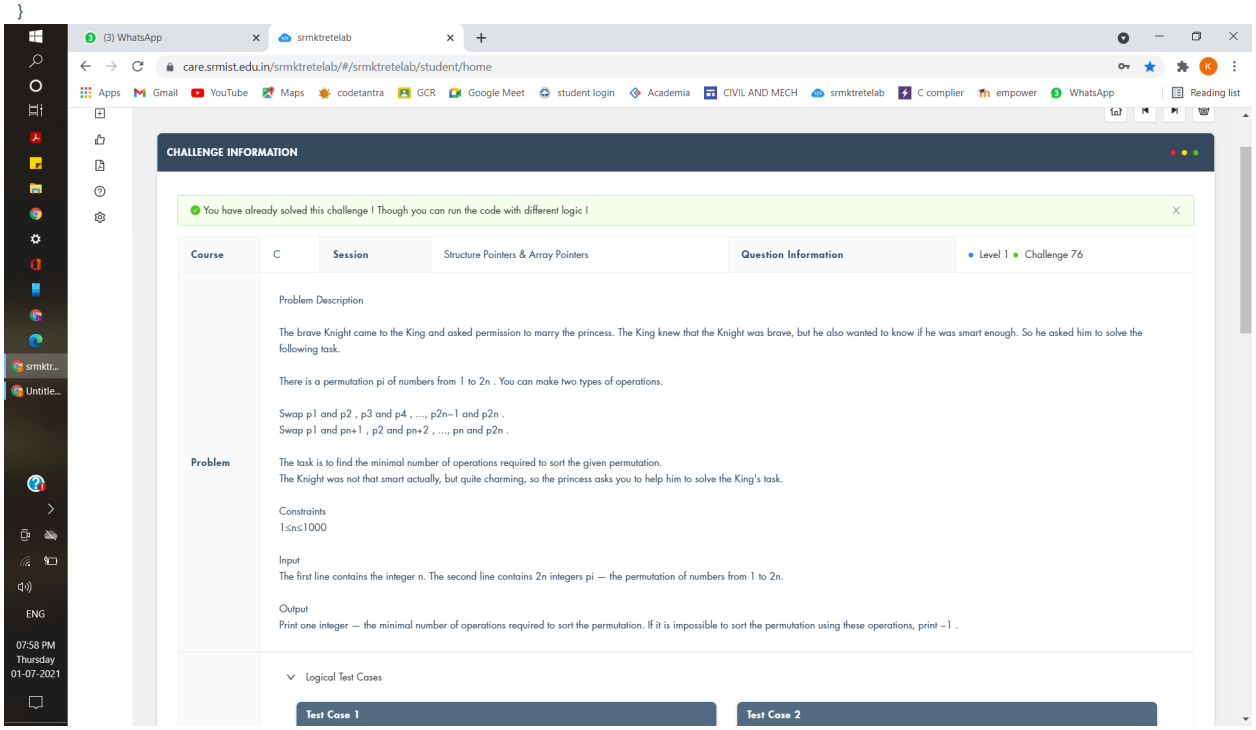



```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   int n,*q,x;
   scanf("%d",&n);
   q=(int*)calloc(n+1, sizeof(int));
   printf("1 ");
   int p=n,i;
   for( i=1;i<=n;i++)
   {
      scanf("%d",&x);
      q[x]=1;
      while(q[p]==1)
         p--;
      printf("%d ",i-n+p+1);}return 0;}
```



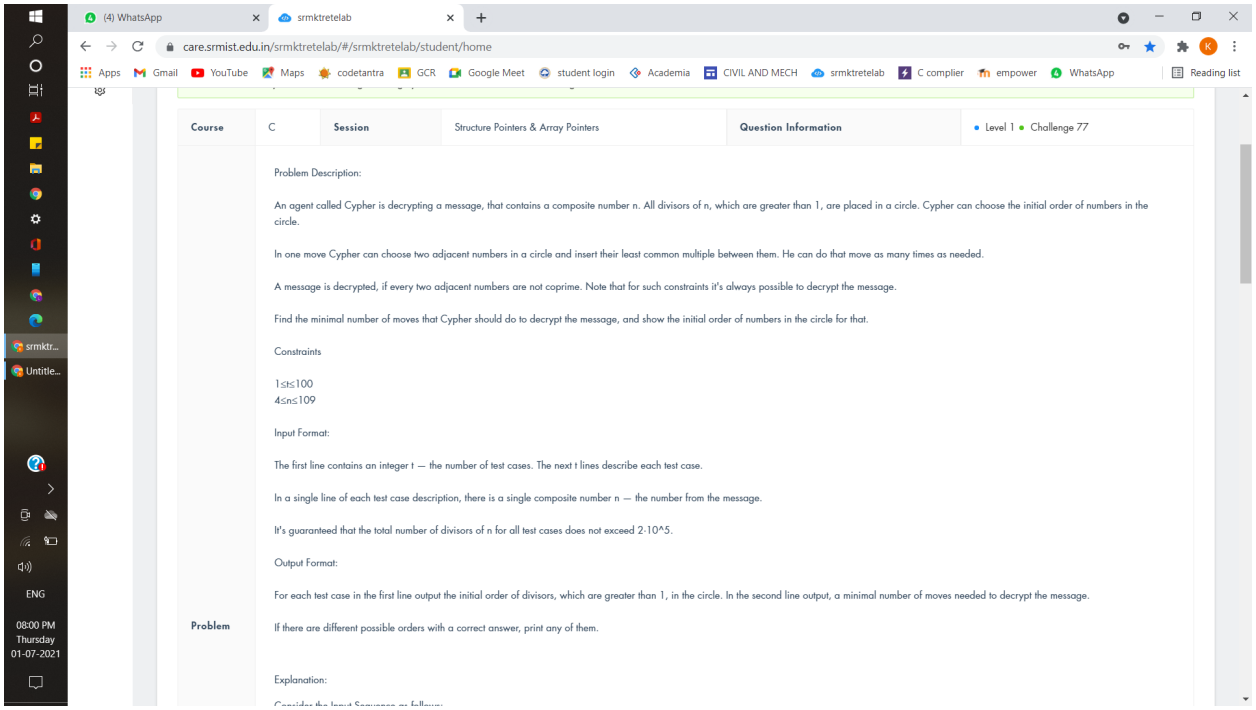

```
#include <stdio.h>
#include <stdlib.h>
   void count(int a[],int n, int k){
                int *f,*temp,i;
                temp=(int*)malloc(n*sizeof(int));
                f=(int*)calloc(k,sizeof(int));
                for(i=0;i<n;i++)
                                f[a[i]%k]++;
                for(i=k-2;i>=0;i--)
                                f[i]=f[i]+f[i+1];
                for(i=n-1;i>=0;i--){
                                temp[f[a[i]%k]-1]=a[i];
                                f[a[i]%k]--;
                }
```

```
        for(i=0;i<n;i++)
                printf("%d ",temp[i]);

}
void sort(int a[],int n,int k,int m){
        int *temp,*f,i;
        f=(int*)calloc(m+1,sizeof(int));
        temp=(int*)malloc(n*sizeof(int));
        for(i=0;i<n;i++)
                f[a[i]]++;
        for(i=1;i<=m;i++)
                f[i]=f[i]+f[i-1];
        for(i=n-1;i>=0;i--){
                temp[f[a[i]]-1]=a[i];
                f[a[i]]--;
        }
        count(temp,n,k);
}
int main()
{
   int n,k,i,*a,max=0;
   scanf("%d %d",&n,&k);
   a=(int*)malloc(n*sizeof(int));
   for(i=0;i<n;i++){
        scanf("%d",&a[i]);
        if(max<a[i])
                max=a[i];
   }
   sort(a,n,k,max);

   return 0;
}
```



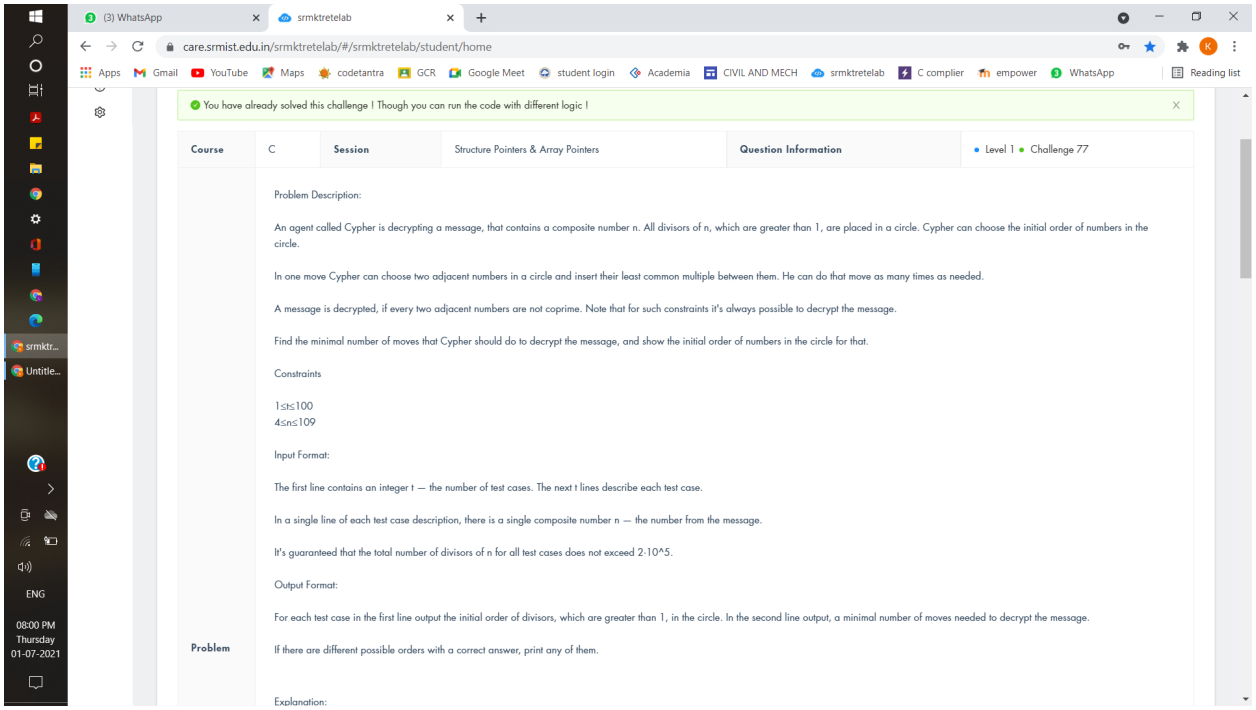

```
#include<stdint.h>
#include<stdio.h>
void option1(int *arr,int n){
  int t=0,i;
  for( i=0;i<n;++i){
    t=arr[2*i];
    arr[2*i]=arr[2*i+1];
    arr[2*i+1]=t;
  }
}
void option2(int *arr,int n){
  int t=0,i;
  for( i=0;i<n;++i){
    t=arr[i];
    arr[i]=arr[i+n];
    arr[i+n]=t;
  }
}

int main()
{
        int n,i,j;
        scanf("%d", &n);
        int arr[2*n], arr_2[2*n];
        for( i=0; i < 2*n; i++)
        {
                scanf(" %d", &arr[i]);
                arr_2[i] = arr[i];
        }
        int t1=-1,t2=-1;
        for(i=0;i<2*n;++i){
```

```
                if(arr[i]!=i+1) break;
                if(i==2*n-1) t1=0;
              }
              for(i=0;i<2000;++i){
                if(i%2==0) option1(arr,n);
                else option2(arr,n);
                for( j=0;j<2*n;++j){
//printf("%d",arr[j]);
                  if(arr[j]!=j+1) break;
                  if(j==2*n-1) t1=i+1;
                }
                        if(t1!=-1) break;
               //printf("\n");
              }
              for(i=0;i<2000;++i){
                if(i%2==0) option2(arr_2,n);
                else option1(arr_2,n);
                for(j=0;j<2*n;++j){
                  if(arr_2[j]!=j+1) break;
                  if(j==2*n-1) t2=i+1;
                }
                        if(t2!=-1) break;
              }
              if(t1<t2) printf("%d\n",t1);
else printf("%d\n",t2);
              return 0;
}
```



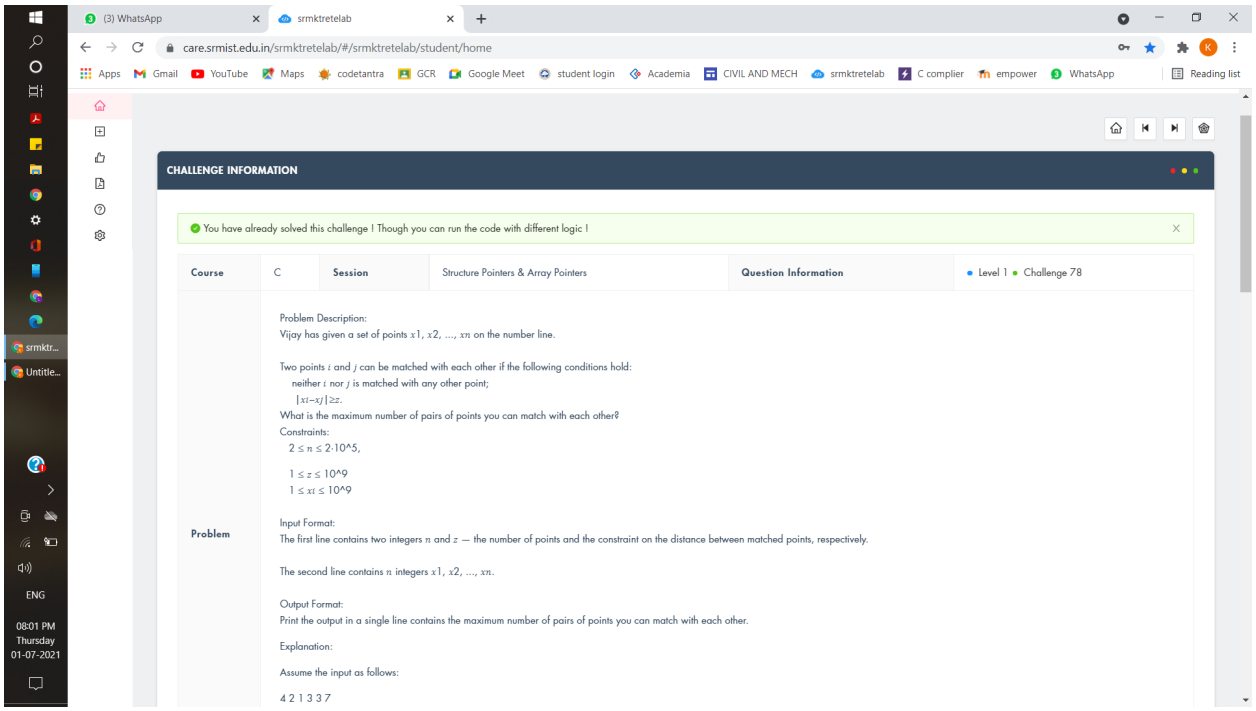

```
#include <stdio.h>
#include <string.h>

#define K     200000

int main() {
        int t;

        scanf("%d", &t);
        while (t--) {
                static int pp[K], dd[K];
                static char used[K];
                int n, n_, kp, kd, p, d, g, h;

                scanf("%d", &n);
                n_ = n;
                kp = 0;
                for (p = 2; p <= n / p; p++)
                        if (n % p == 0) {
                                while (n % p == 0)
                                        n /= p;
                                pp[kp++] = p;
                        }
                if (n > 1)
                        pp[kp++] = n;
                n = n_;
                kd = 0;
                for (d = 2; d <= n / d; d++)
                        if (n % d == 0) {
                                dd[kd++] = d;
                                if (d != n / d)
                                        dd[kd++] = n / d;
                        }
                if (kp == 2 && pp[0] * pp[1] == n) {
                        printf("%d %d %d\n", pp[0], pp[1], n);
                        printf("1\n");
```

```
                continue;
        }
        memset(used, 0, kd * sizeof *used);
        for (g = 0; g + 1 < kp; g++) {
                int d = pp[g] * pp[g + 1];

                for (h = 0; h < kd; h++)
                        if (dd[h] == d) {
                                used[h] = 1;
                                break;
                        }
        }
        for (g = 0; g < kp; g++) {
                p = pp[g];
                for (h = 0; h < kd; h++)
                        if (!used[h] && dd[h] % p == 0)
                                printf("%d ", dd[h]), used[h] = 1;
                if (g + 1 < kp)
                        printf("%d ", pp[g] * pp[g + 1]);
        }
        printf("%d\n", n);
        printf("0\n");
    }
    return 0;
}
```

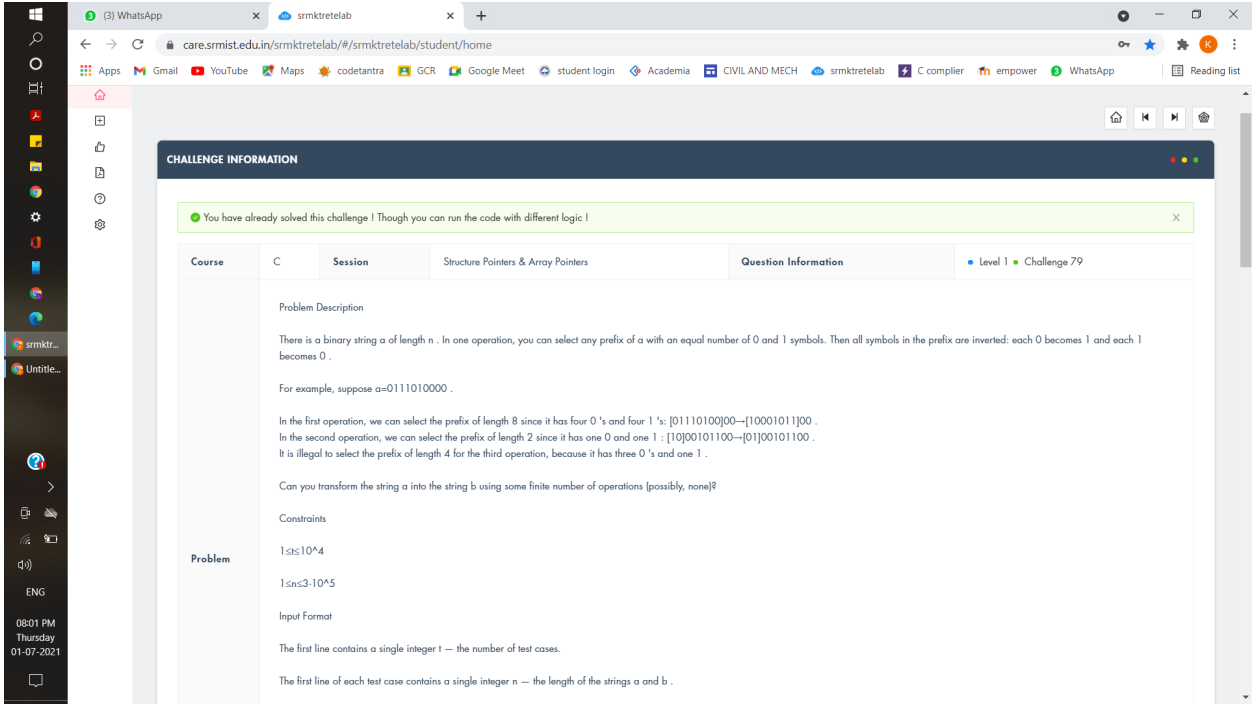



```
#include <stdio.h>
#include <string.h>

#define K     200000

int main() {
        int t;

        scanf("%d", &t);
        while (t--) {
                static int pp[K], dd[K];
                static char used[K];
                int n, n_, kp, kd, p, d, g, h;

                scanf("%d", &n);
                n_ = n;
                kp = 0;
                for (p = 2; p <= n / p; p++)
                        if (n % p == 0) {
                                while (n % p == 0)
                                        n /= p;
                                pp[kp++] = p;
                        }
                if (n > 1)
                        pp[kp++] = n;
                n = n_;
                kd = 0;
                for (d = 2; d <= n / d; d++)
                        if (n % d == 0) {
                                dd[kd++] = d;
                                if (d != n / d)
                                        dd[kd++] = n / d;
                        }
                if (kp == 2 && pp[0] * pp[1] == n) {
                        printf("%d %d %d\n", pp[0], pp[1], n);
                        printf("1\n");
                        continue;
                }
```

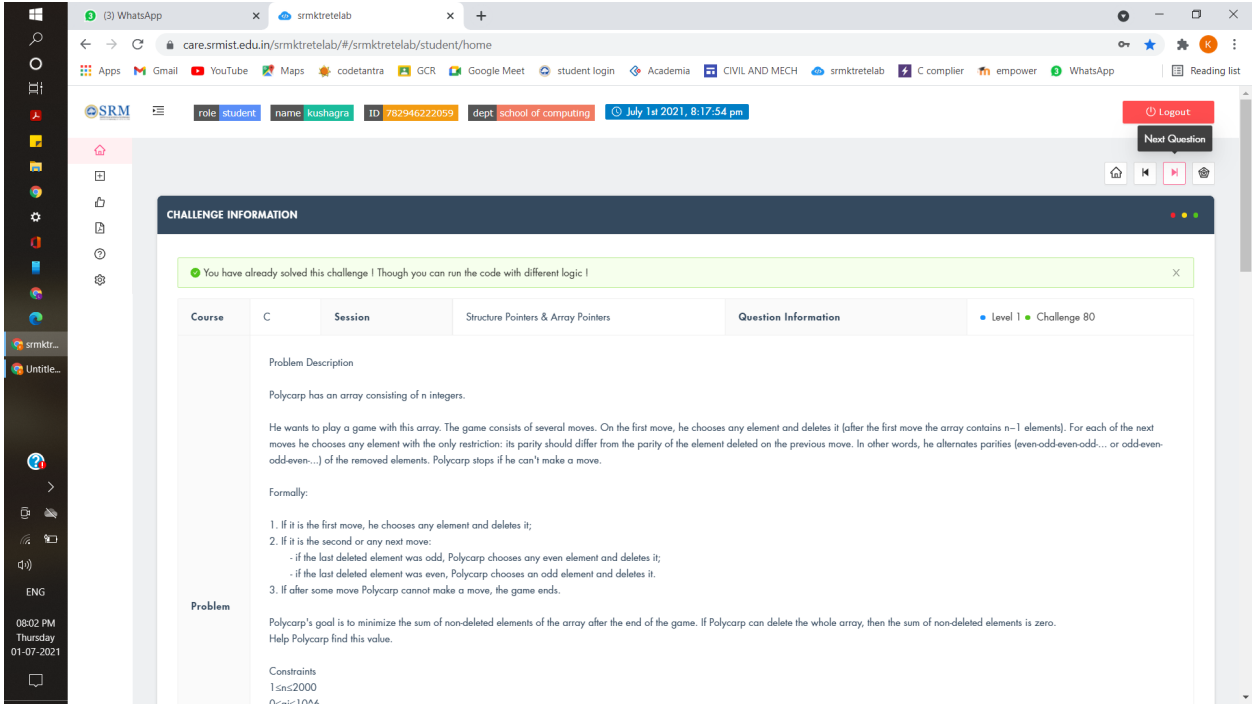
```
                memset(used, 0, kd * sizeof *used);
                for (g = 0; g + 1 < kp; g++) {
                        int d = pp[g] * pp[g + 1];

                        for (h = 0; h < kd; h++)
                                if (dd[h] == d) {
                                        used[h] = 1;
                                        break;
                                }
                }
                for (g = 0; g < kp; g++) {
                        p = pp[g];
                        for (h = 0; h < kd; h++)
                                if (!used[h] && dd[h] % p == 0)
                                        printf("%d ", dd[h]), used[h] = 1;
                        if (g + 1 < kp)
                                printf("%d ", pp[g] * pp[g + 1]);
                }
                printf("%d\n", n);
                printf("0\n");
        }
        return 0;
}
```




```
#include<stdio.h>
#include<stdlib.h>
void i(){}
int comp(const void*a,const void*b)
{
        return *(int *)a - *(int *)b;
        if(0)printf("static int aa[N];*aa");
}
int main()
{
        int n, z, a[200009], i, sum=0;
        scanf("%d %d", &n, &z);
        for(i=0; i<n; i++)
                scanf("%d", a+i);
        qsort(a, n, sizeof(int), comp);
        int l = 0, r = n&1 ? (n>>1)+1 : n>>1;
        for(i=0; i<n; i++)
        while(r < n)
        {
                if(a[r]-a[l] >= z)
                        sum++, l++;
                r++;
        }
        printf("%d", sum);
        return 0;
}
```




```
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main() {
	int n_cases, n, balance, diff;
	char s1[300001], s2[300001], *c1, *c2;
	bool any_same, any_different;
	scanf("%d", &n_cases);
	while (n_cases--) {
		scanf("%d", &n);
		scanf("%s\n%s", s1, s2);
		c1 = s1;
		c2 = s2;
		any_same = false;
		any_different = false;
		balance = 0;
		diff = 0;
		while (*c1) {
			any_same = any_same || *c1 == *c2;
			any_different = any_different || *c1 != *c2;
			if (any_same && any_different) break;
			balance += *c2 == '1' ? 1 : -1;
			diff += *c1 - *c2;
			if (balance == 0) {
				any_same = false;
				any_different = false;
			}
			c1++;
			c2++;
		}
		printf(((any_same && any_different) || diff != 0) ? "NO\n" : "YES\n");
	}
	return 0;
}
```




```
#include <stdio.h>
#include <stdlib.h>
int cmp(const void *a, const void *b) {
		return *(int*)a - *(int*)b;
}
int main() {
		int o[2000], ol = 0, e[2000], el = 0, n, t;
		scanf("%d", &n);
		while(n--) {
			scanf("%d", &t);
			if(t % 2)
				o[ol++] = t;
			else
				e[el++] = t;
		}
		qsort(o, ol, sizeof(int), cmp);
		qsort(e, el, sizeof(int), cmp);
		while(ol && el) {
			ol--;
			el--;
		}
		t = 0;
		if(ol) {
			ol--;
			while(ol)
				t += o[--ol];
		} else if(el) {
			el--;
			while(el)
				t += e[--el];}
		printf("%d", t);
		return 0;}
```