NAME : ABHISHEK SHARMA

CS 2nd YEAR

SECTION : “I”

ROLL NO.: 01

ENROLLMENT NO.: 12019009001127

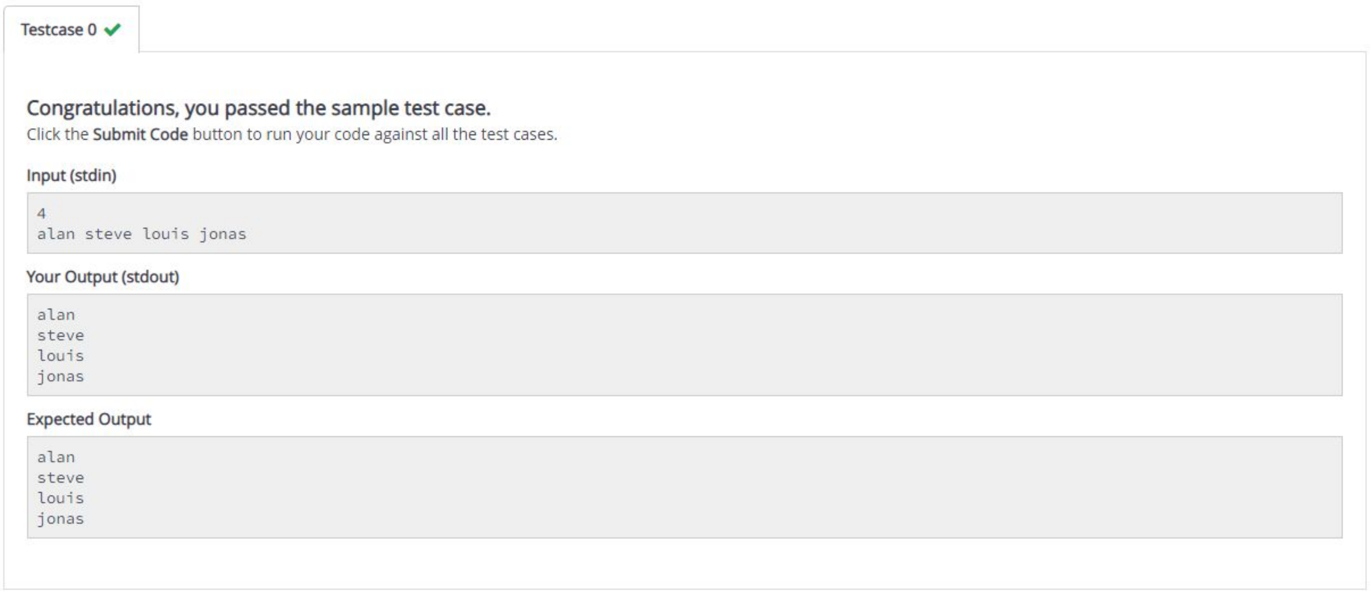
DATA STRUCTURE LAB ASSIGNMENTS [WEEK 2]

DATE : 13.07.2020

Q1. Take an 1D array of size n. It contains names of students in a class. Print the names in separate lines.
CODE :

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

int main() {

    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int n,i;
    scanf("%d",&n);
    char a[n][20];
    for(i=0;i<n;i++)
        scanf("%s ",a[i]);
    for(i=0;i<n;i++)
        printf("%s\n",a[i]);
    return 0;
}
```

OUTPUT :


Testcase 0 ✓

Congratulations, you passed the sample test case.
Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
4
alan steve louis jonas
```

Your Output (stdout)

```
alan
steve
louis
jonas
```

Expected Output

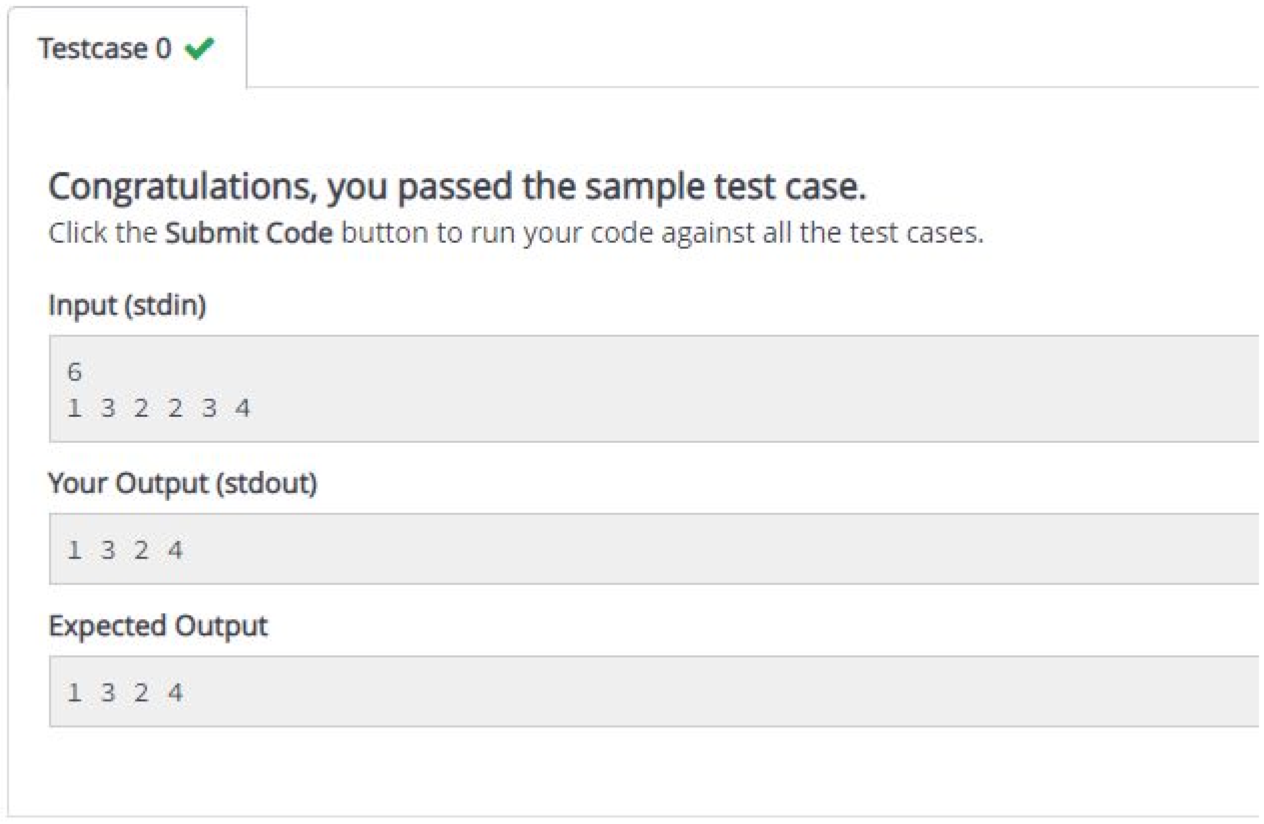
```
alan
steve
louis
jonas
```

Q2. Delete all the duplicate elements from an 1D array. Size of the array is user input.
CODE :

```
#include<stdio.h>

int main()
{
    int a[20], i, j, k, n;


    scanf("%d", &n);
    for(i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }

    for(i = 0; i < n; i++)
    {
        for(j = i+1; j < n; )
        {
            if(a[j] == a[i])
            {
                for(k = j; k < n; k++)
                {
                    a[k] = a[k+1];
                }
                n--;
            }
            else
            {
                j++;
            }
        }
    }

    for(i = 0; i < n; i++)
    {
        printf("%d ", a[i]);
    }

}
```

P.T.O.

OUTPUT :



Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
6
1 3 2 2 3 4
```

Your Output (stdout)

```
1 3 2 4
```

Expected Output

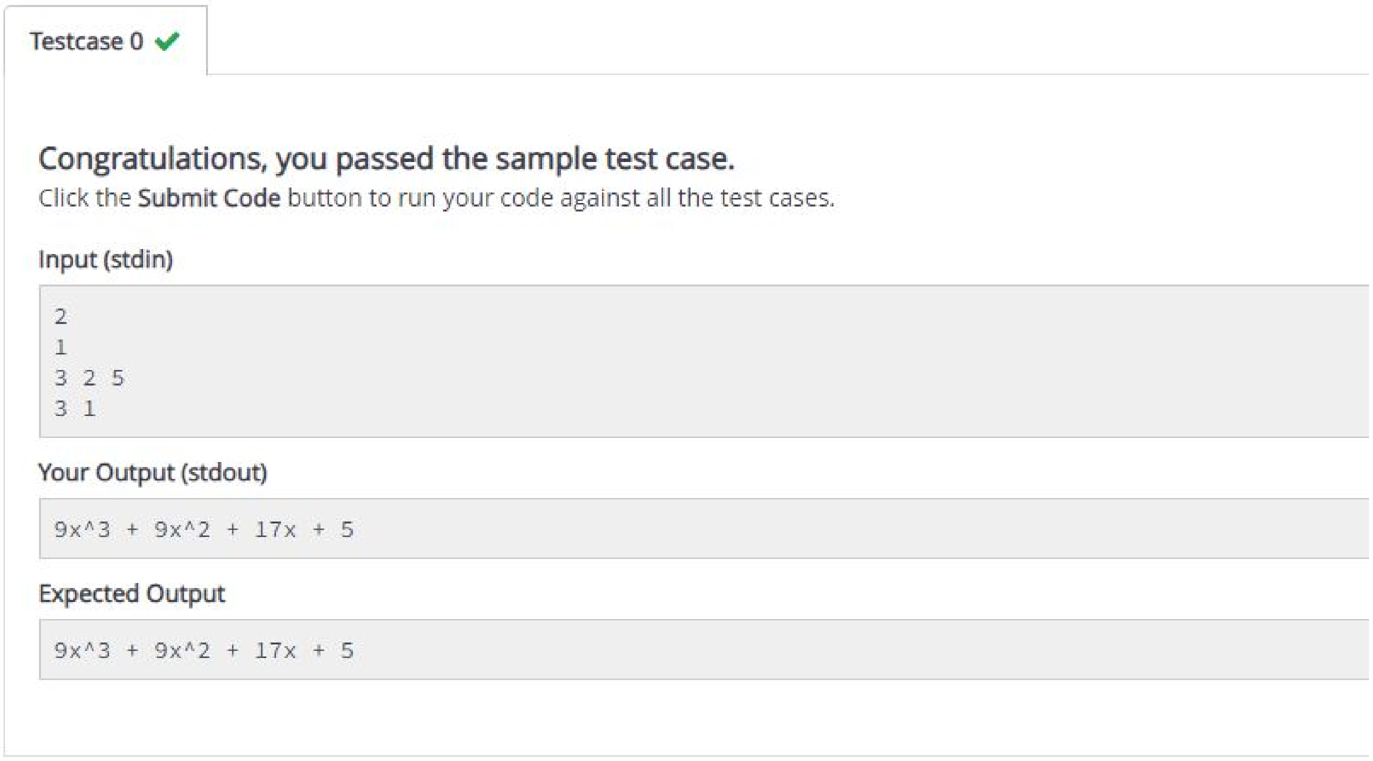
```
1 3 2 4
```

Q3. Perform multiplication of two polynomial using arrays
CODE :

```
#include <stdio.h>

#define MAX 10

struct term
{
    int coeff ;
    int exp ;
} ;

struct poly
{
    struct term t [10] ;
    int noofterms ;
} ;

void initpoly ( struct poly *) ;
void polyappend ( struct poly *, int, int ) ;
struct poly polyadd ( struct poly, struct poly ) ;
struct poly polymul ( struct poly, struct poly ) ;
void display ( struct poly ) ;

int main( )
```

```
{
    struct poly p1, p2, p3 ;

    initpoly ( &p1 ) ;
    initpoly ( &p2 ) ;
    initpoly ( &p3 ) ;

    polyappend ( &p1,3,2 ) ;
    polyappend ( &p1,2,1 ) ;
    polyappend ( &p1,5,0 ) ;

    polyappend ( &p2,3,1 ) ;
    polyappend ( &p2,1,0 ) ;

    p3 = polymul (p1,p2) ;
    display (p3) ;

    return 0; ;
}

/* initializes elements of struct poly */
void initpoly ( struct poly *p )
{
    int i ;
    p -> noofterms = 0 ;
    for ( i = 0 ; i < MAX ; i++ )
    {
        p -> t[i].coeff = 0 ;
        p -> t[i].exp = 0 ;
    }
}

/* adds the term of polynomial to the array t */
void polyappend ( struct poly *p, int c, int e )
{
    p -> t[p -> noofterms].coeff = c ;
    p -> t[p -> noofterms].exp =  e ;
    ( p -> noofterms ) ++ ;
}

/* displays the polynomial equation */
void display ( struct poly p )
{
    int flag = 0, i ;
    for ( i = 0 ; i < p.noofterms ; i++ )
    {
        if ( p.t[i].exp != 0 )
            if(p.t[i].exp ==1)
            printf ( "%dx + ", p.t[i].coeff ) ;
        else
            printf ( "%dx^%d + ", p.t[i].coeff, p.t[i].exp ) ;
        else
        {
            printf ( "%d", p.t[i].coeff ) ;
            flag = 1 ;
        }


    }
    if ( !flag )
        printf ( "\b\b  " ) ;

}
/* adds two polynomials p1 and p2 */
struct poly polyadd ( struct poly p1, struct poly p2 )
{
    int i, j, c ;
    struct poly p3 ;
    initpoly ( &p3 ) ;

    if ( p1.noofterms > p2.noofterms )
        c = p1.noofterms ;
    else
        c = p2.noofterms ;

    for ( i = 0, j = 0 ; i <= c ; p3.noofterms++ )
    {
        if ( p1.t[i].coeff == 0 && p2.t[j].coeff == 0 )
            break ;
        if ( p1.t[i].exp >= p2.t[j].exp )
        {
            if ( p1.t[i].exp == p2.t[j].exp )
            {
                p3.t[p3.noofterms].coeff = p1.t[i].coeff + p2.t[j].coeff ;
                p3.t[p3.noofterms].exp = p1.t[i].exp ;
                i++ ;
                j++ ;
            }
            else
            {
                p3.t[p3.noofterms].coeff = p1.t[i].coeff ;
                p3.t[p3.noofterms].exp = p1.t[i].exp ;
                i++ ;
            }
        }
        else
        {
            p3.t[p3.noofterms].coeff = p2.t[j].coeff ;
            p3.t[p3.noofterms].exp = p2.t[j].exp ;
            j++ ;
        }
    }
    return p3 ;
}

/* multiplies two polynomials p1 and p2 */
struct poly polymul ( struct poly p1, struct poly p2 )
{
    int coeff, exp ;
    struct poly temp, p3 ;

    initpoly ( &temp ) ;
    initpoly ( &p3 ) ;

    if ( p1.noofterms != 0 && p2.noofterms != 0 )
    {
        int i ;
        for ( i = 0 ; i < p1.noofterms ; i++ )
        {
            int j ;

            struct poly p ;
            initpoly ( &p ) ;

            for ( j = 0 ; j < p2.noofterms ; j++ )
            {
                coeff = p1.t[i].coeff * p2.t[j].coeff ;
                exp = p1.t[i].exp + p2.t[j].exp ;
                polyappend ( &p, coeff, exp ) ;
            }

            if ( i != 0 )
            {
                p3 = polyadd ( temp, p ) ;
                temp = p3  ;
            }
            else
                temp = p ;
        }
    }
    return p3 ;
}
```

OUTPUT :



Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
2
1
3 2 5
3 1
```

Your Output (stdout)

```
9x^3 + 9x^2 + 17x + 5
```

Expected Output

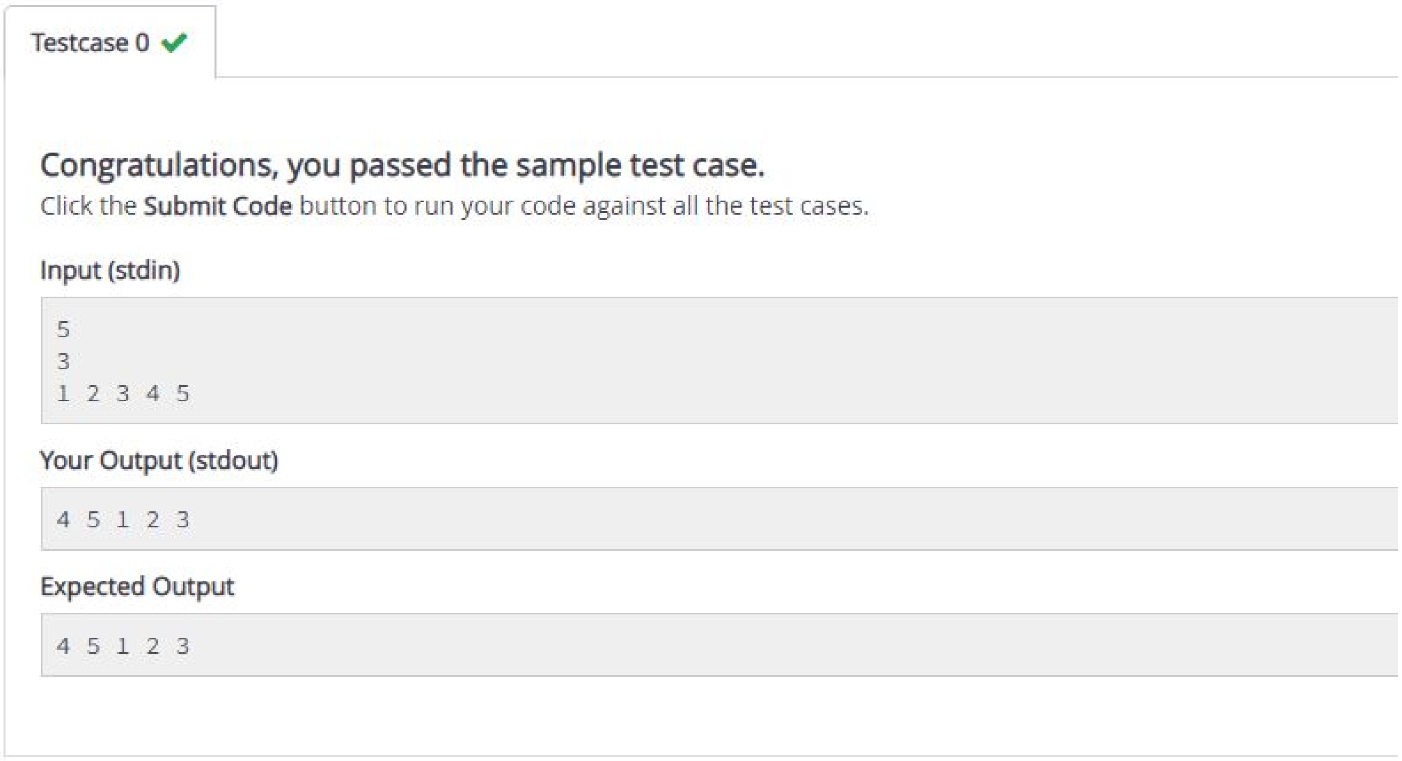
```
9x^3 + 9x^2 + 17x + 5
```

Q4. Perform right shift operation by n on an 1D array of size m. For example say, A[5] = {1,2,3,4,5}. If we are right shifting the elements by 3 then the output will be {4,5,1,2,3}

CODE :

```
#include <stdio.h>

int main()
{

      int n;
    scanf ("%d",&n);
    int k;
    scanf ("%d",&k);
 int arr[n];
  // taking input and storing it in an array
  for(int i = 0; i < n; ++i) {
     scanf("%d", &arr[i]);
  }


    //Calculate length of array arr
    int length = sizeof(arr)/sizeof(arr[0]);
    //Rotate the given array by n times toward left
    for(int i = 0; i < k; i++){
        int j, first;
        //Stores the first element of the array
        first = arr[0];

        for(j = 0; j < length-1; j++){
            //Shift element of array by one
            arr[j] = arr[j+1];
        }
        //First element of array will be added to the end
        arr[j] = first;
    }



    for(int i = 0; i < length; i++){
        printf("%d ", arr[i]);
    }
    return 0;
}
```

OUTPUT :



Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
5
3
1 2 3 4 5
```

Your Output (stdout)

```
4 5 1 2 3
```

Expected Output

```
4 5 1 2 3
```

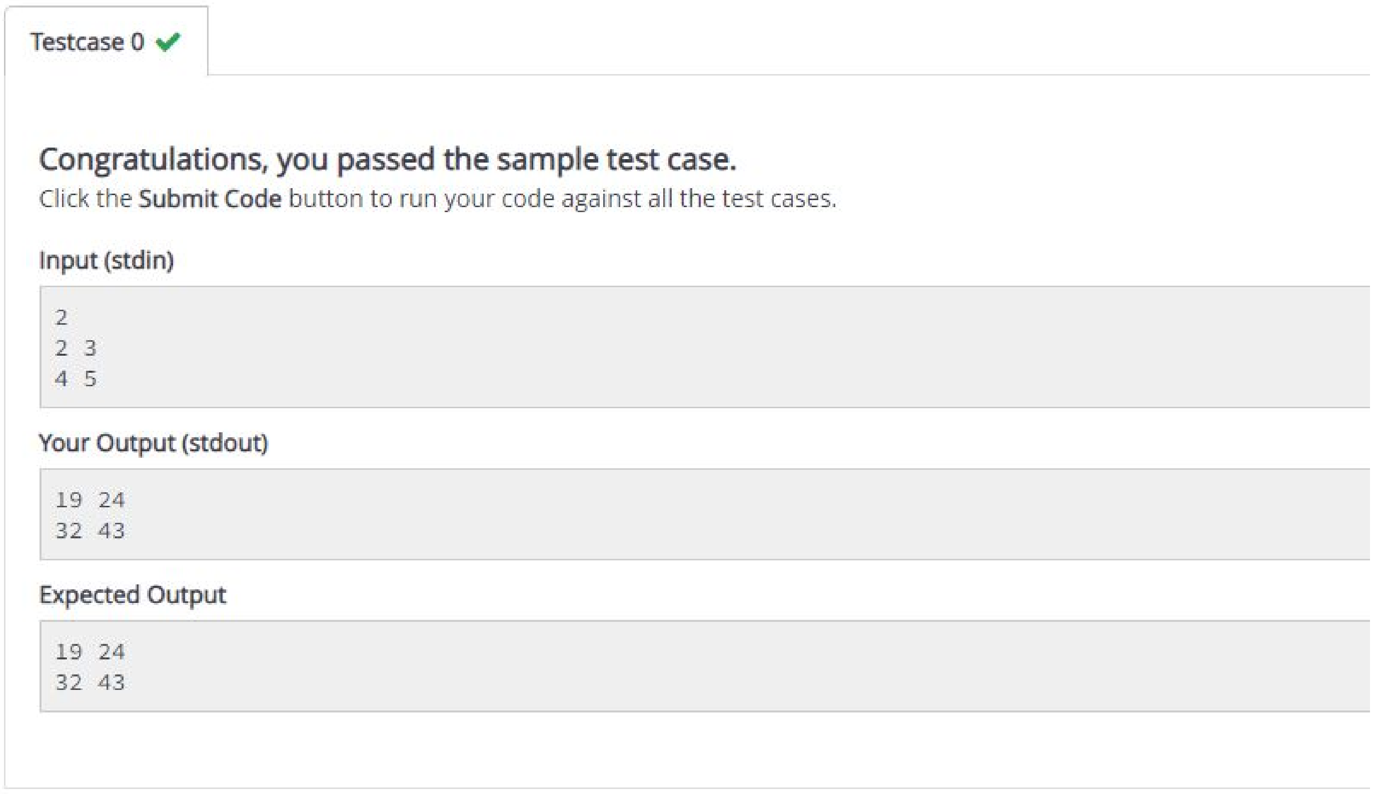
Q5. A is a square matrix of dimension nxn. ‘n’ is user input. Calculate the value of the following equation A^2+A+I

CODE :

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
int main() {

    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    int i,j,k,n,A[100][100],C[100][100];
    scanf("%d",&n);
    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
            scanf("%d",&A[i][j]);
    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
        {
            C[i][j]=0;
            for(k=0;k<n;k++)
                C[i][j]+=A[i][k]*A[k][j];
        }
    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
        {
            if(i==j)
            {
                C[i][j]+=A[i][j]+1;
            }
            else
            {
                C[i][j]+=A[i][j];
            }
        }
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
            printf("%d ",C[i][j]);
        printf("\n");
    }
    return 0;
}
```

P.T.O.

OUTPUT :


Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
2
2 3
4 5
```

Your Output (stdout)

```
19 24
32 43
```

Expected Output

```
19 24
32 43
```

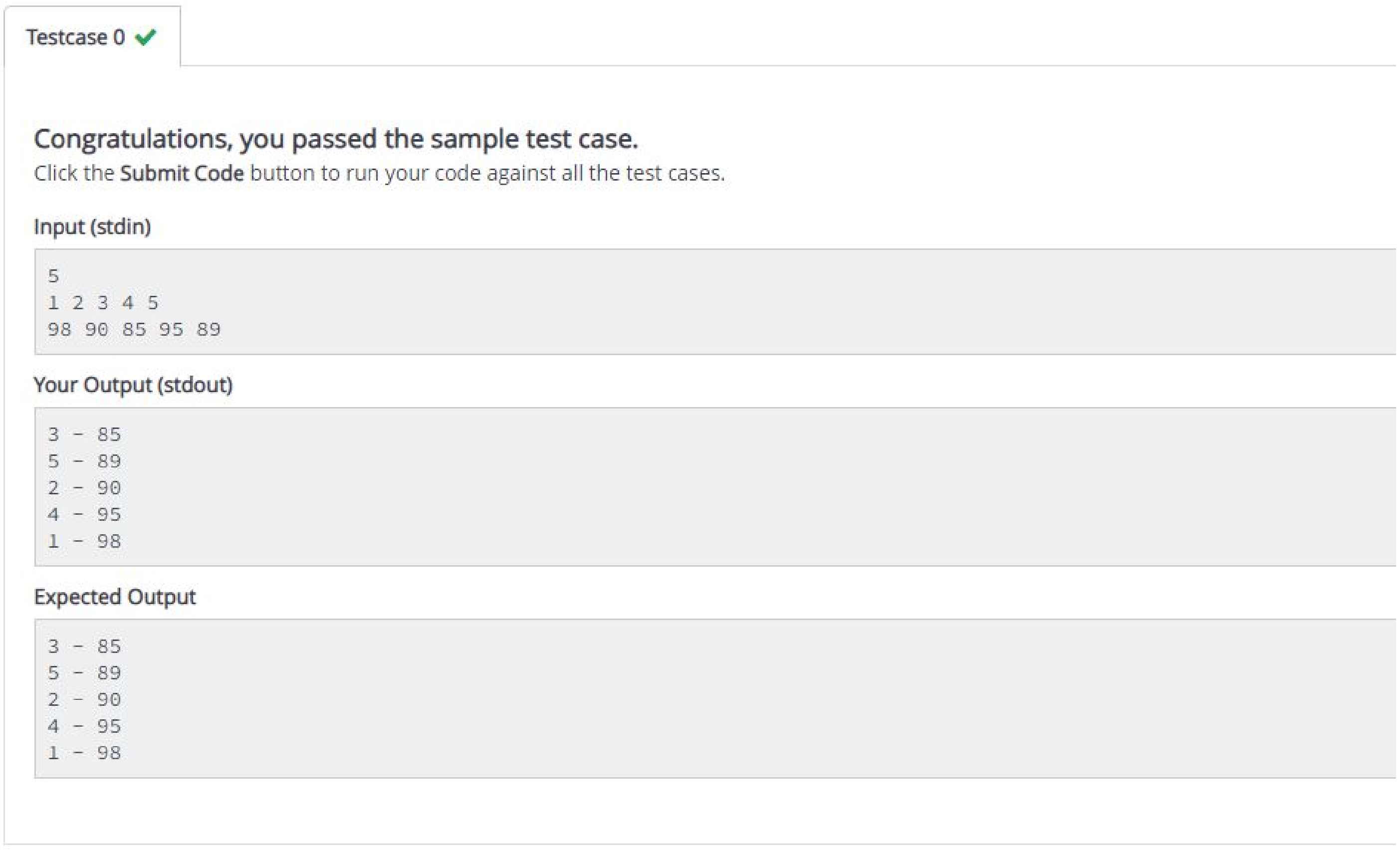
Q6. Take a 2D array of size 2Xn. 1st row contains the roll numbers and 2nd row contains corresponding marks in exam. 'n' is the number of students here. Now sort the marks in ascending order along with the roll numbers and.

CODE :

```
#include<stdio.h>

struct student
{
   int roll_no,marks;
}stud[100],t;

int main()
{
   int i,j,n;

   scanf("%d",&n);

   for(i=0;i<n;i++)
   {
      scanf("%d",&stud[i].roll_no);
   }
    for(i=0;i<n;i++)
   {
      scanf("%d",&stud[i].marks);
   }
```

```
    for(i=0;i<n;i++)
    {
        for(j=0;j<n-1;j++)
        {
            if(stud[j].marks>stud[j+1].marks)
            {
                t=stud[j];
                stud[j]=stud[j+1];
                stud[j+1]=t;
            }
        }
    }

    for(i=0;i<n;i++)
    {
        printf("%d - %d\n",stud[i].roll_no,stud[i].marks);
    }
}
```

OUTPUT :



Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
5
1 2 3 4 5
98 90 85 95 89
```

Your Output (stdout)

```
3 - 85
5 - 89
2 - 90
4 - 95
1 - 98
```

Expected Output

```
3 - 85
5 - 89
2 - 90
4 - 95
1 - 98
```

Q7. You have been given an array of n positive numbers. Now print the same array of n length but here every index will contain the sum of all the elements in the array except the element on that index. For example a[i] = a[0]+a[1]+....+a[i-1]+a[i+1]+…+a[n-1].

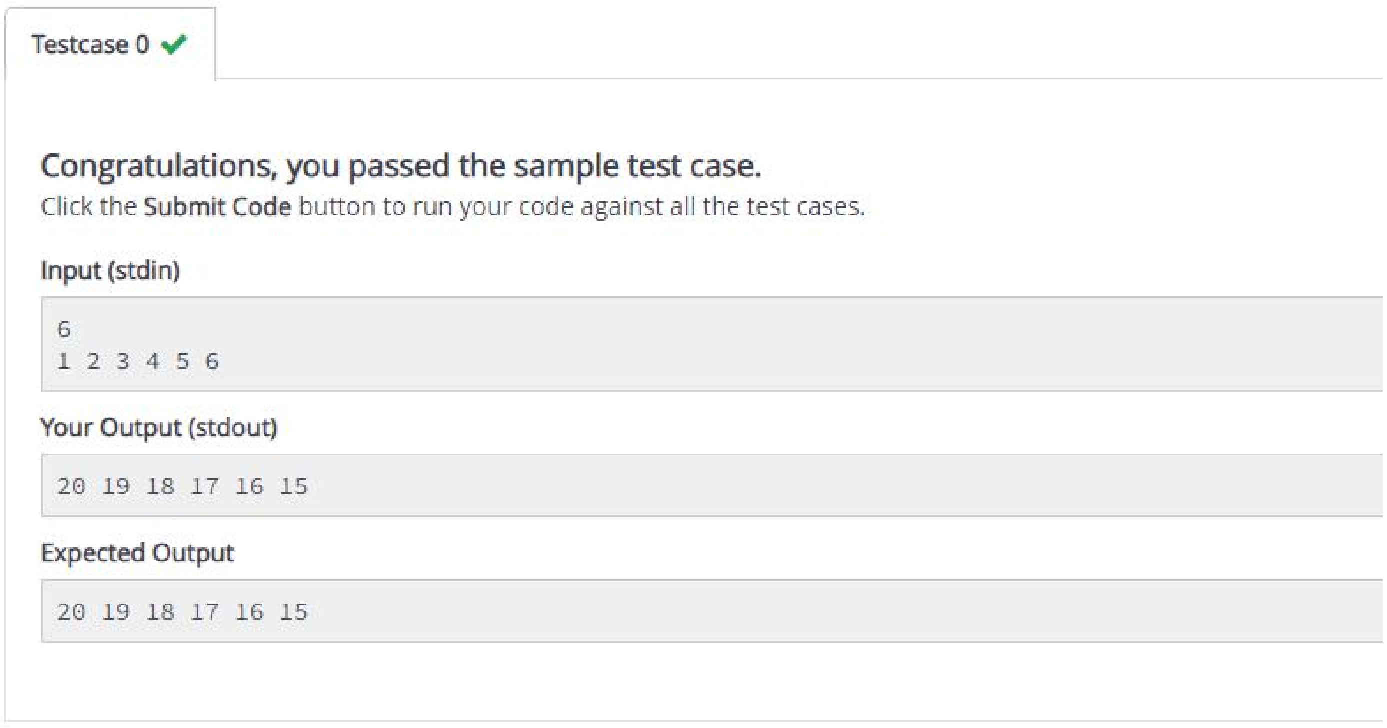
CODE :

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

```
void productArray(int arr[], int n)
{
    if (n == 1) {
        printf("0");
        return;
    }
    int* left = (int*)malloc(
        sizeof(int) * n);
    int* right = (int*)malloc(
        sizeof(int) * n);
    int* prod = (int*)malloc(
        sizeof(int) * n);
    int i, j;
    left[0] = 1;
    right[n - 1] = 1;
    for (i = 1; i < n; i++)
        left[i] = arr[i - 1] + left[i - 1];
    for (j = n - 2; j >= 0; j--)
        right[j] = arr[j + 1] + right[j + 1];
    for (i = 0; i < n; i++)
        prod[i] = ((left[i] + right[i])-2);
    for (i = 0; i < n; i++)
        printf("%d ", prod[i]);
    return;
}
int main()
{

    int z;
    scanf ("%d",&z);
    int arr[z];
    for (int y=0;y<z;y++)
        scanf ("%d",&arr[y]);
    int n = sizeof(arr) / sizeof(arr[0]);
    productArray(arr, n);
    getchar();
}
```

P.T.O.

OUTPUT :


Testcase 0 ✓

Congratulations, you passed the sample test case.
Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
6
1 2 3 4 5 6
```

Your Output (stdout)

```
20 19 18 17 16 15
```

Expected Output

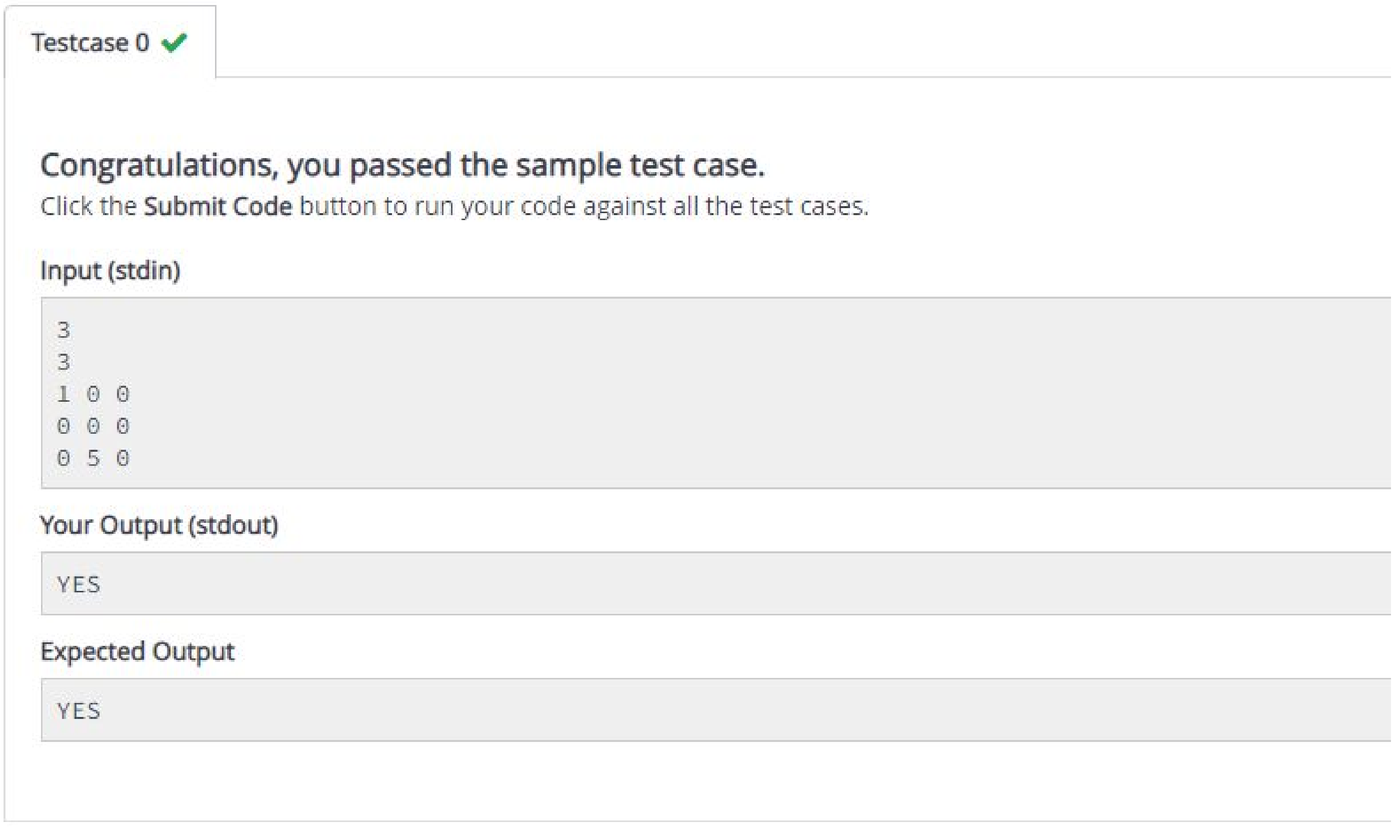
```
20 19 18 17 16 15
```

Q8. Check whether a matrix of nXm size is sparse or not
CODE :

```
#include <stdio.h>

int main ()
{
   int matrix[10][10];
   int i, j, m, n;
   int sparse_counter = 0;


   scanf("%d %d", &m, &n);

   for (i = 0; i < m; ++i)
   {
      for (j = 0; j < n; ++j)
      {
         scanf("%d", &matrix[i][j]);
         if (matrix[i][j] == 0)
         {
            ++sparse_counter;
         }
      }
   }
   if (sparse_counter > ((m * n) / 2))
   {
      printf("YES\n");
```

```
    }
    else
        printf("NO\n");

}
```

OUTPUT :



Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
3
3
1 0 0
0 0 0
0 5 0
```

Your Output (stdout)

```
YES
```

Expected Output

```
YES
```

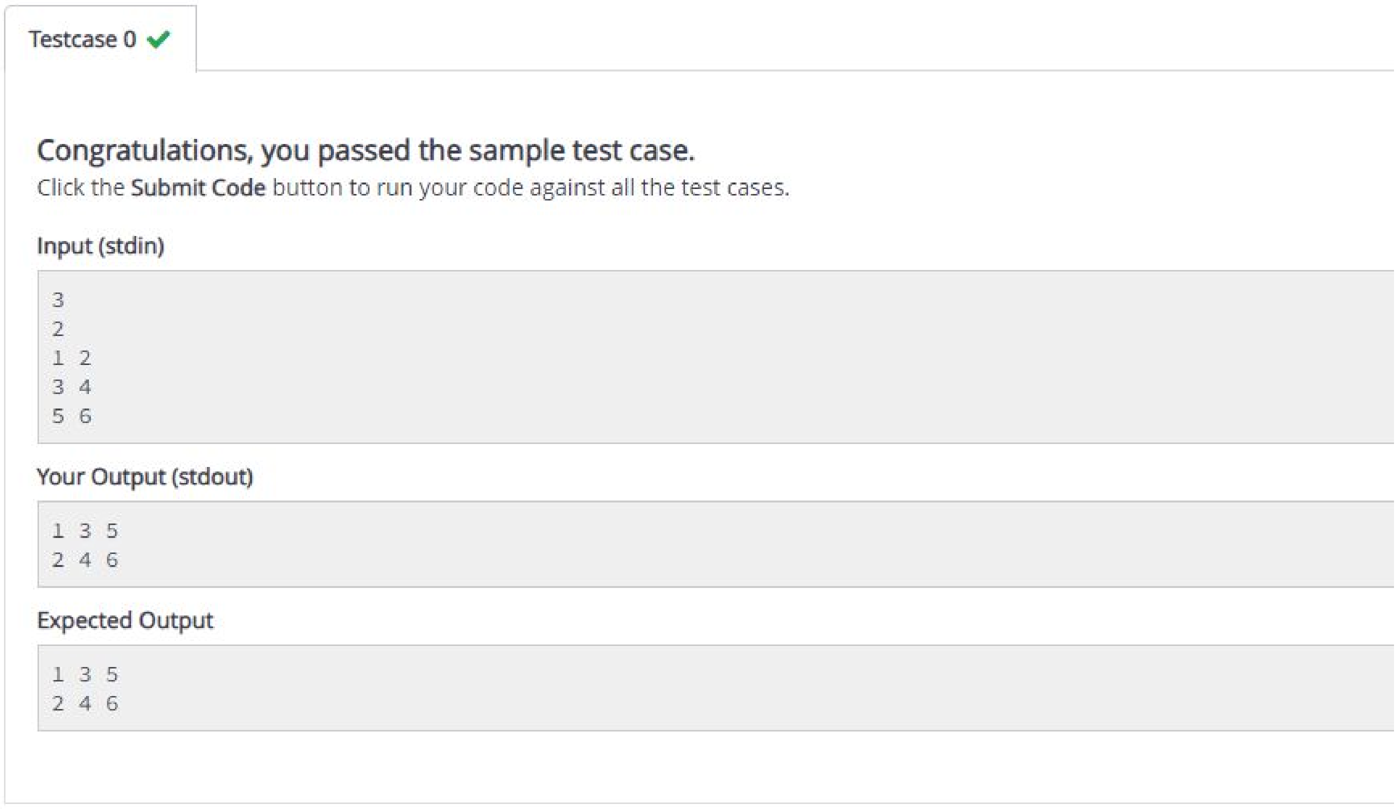
Q9. Determine transpose of a given nXm matrix

CODE :

```
#include <stdio.h>
int main() {
    int a[10][10], transpose[10][10], r, c, i, j;

    scanf("%d %d", &r, &c);



    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {

            scanf("%d", &a[i][j]);
        }




    // Finding the transpose of matrix a
    for (i = 0; i < r; ++i)
```

```
        for (j = 0; j < c; ++j) {
            transpose[j][i] = a[i][j];
        }


    for (i = 0; i < c; ++i)
        for (j = 0; j < r; ++j) {
            printf("%d ", transpose[i][j]);
            if (j == r - 1)
                printf("\n");
        }
    return 0;
}
```

OUTPUT :



Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
3
2
1 2
3 4
5 6
```

Your Output (stdout)

```
1 3 5
2 4 6
```

Expected Output

```
1 3 5
2 4 6
```

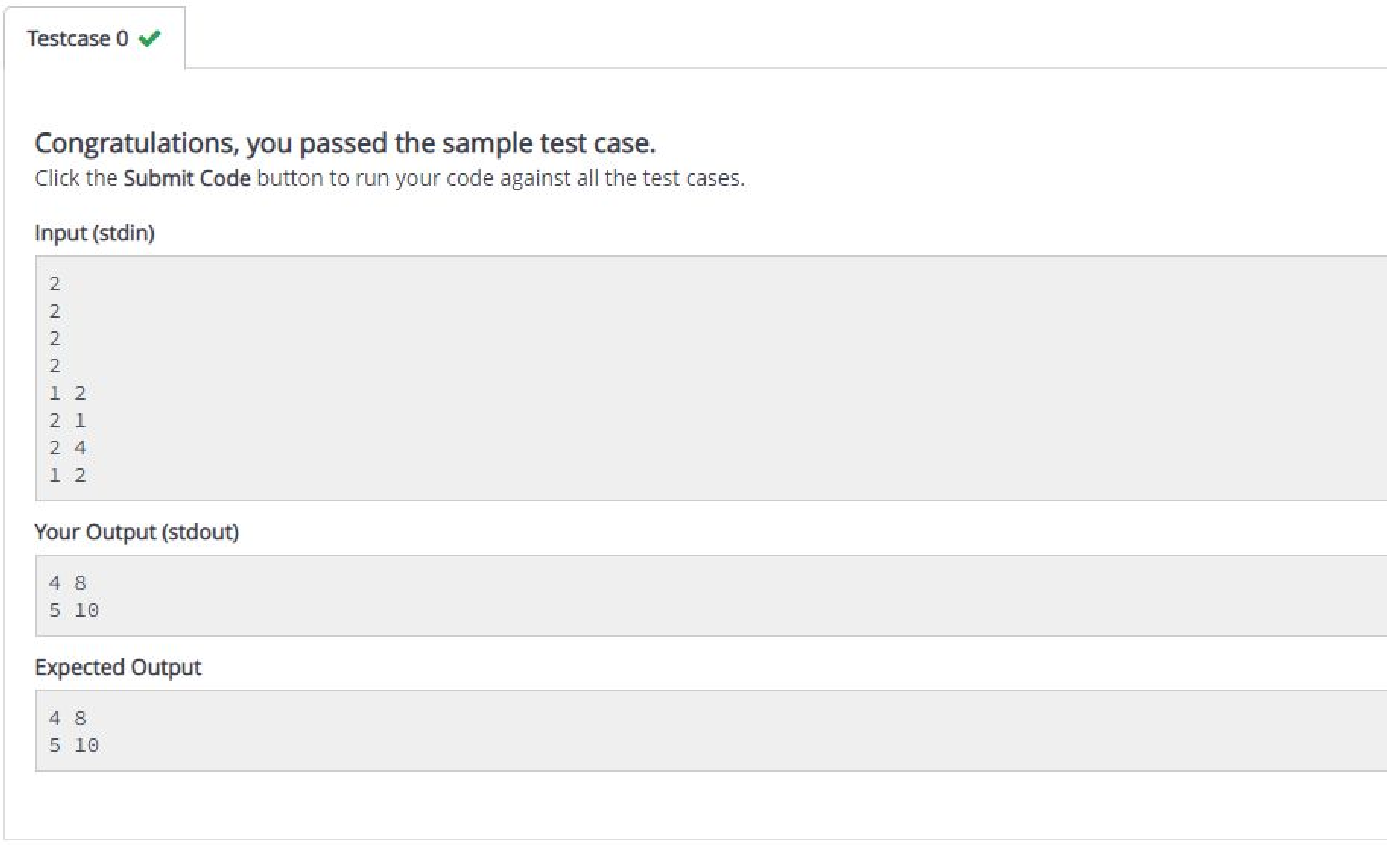
Q10. Perform multiplication of two matrices using pointer to array

CODE :

```
#include<stdio.h>
int main()
{
int fst_mat[100][100],snd_mat[100][100],mul_mat[100][100],r,c,m1,n1,m2,n2,k,sum=0;
label:
scanf("%d",&n1);
scanf("%d",&m1);
scanf("%d",&n2);
scanf("%d",&m2);
while(m1!=n2)
{
printf("MATRIX MULTIPLICATION IS NOT POSSIBLE . . . . . . . . . . TRY AGAIN ");
goto label;
}
for(r=1;r<=n1;r++)
{
for(c=1;c<=m1;c++)
{
scanf("%d", &fst_mat[r][c]);

}

}
for(r=1;r<=n2;r++)
{
for(c=1;c<=m2;c++)
{
scanf("%d",&snd_mat[r][c]);

}

}
for(r=1;r<=n1;r++)
{
for(c=1;c<=m2;c++)
{
for(k=1;k<=n2;k++)
{
   sum=sum+(fst_mat[r][k]*snd_mat[k][c]);
}
mul_mat[r][c]=sum;
sum=0;
}
}
for(r=1;r<=n1;r++)
{
   for(c=1;c<=m2;c++)
   {
      printf("%d ",mul_mat[r][c]);
   }
   printf("\n");
}
}
```

P.T.O.

OUTPUT :



Testcase 0 ✓

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
2
2
2
2
1 2
2 1
2 4
1 2
```

Your Output (stdout)

```
4 8
5 10
```

Expected Output

```
4 8
5 10
```