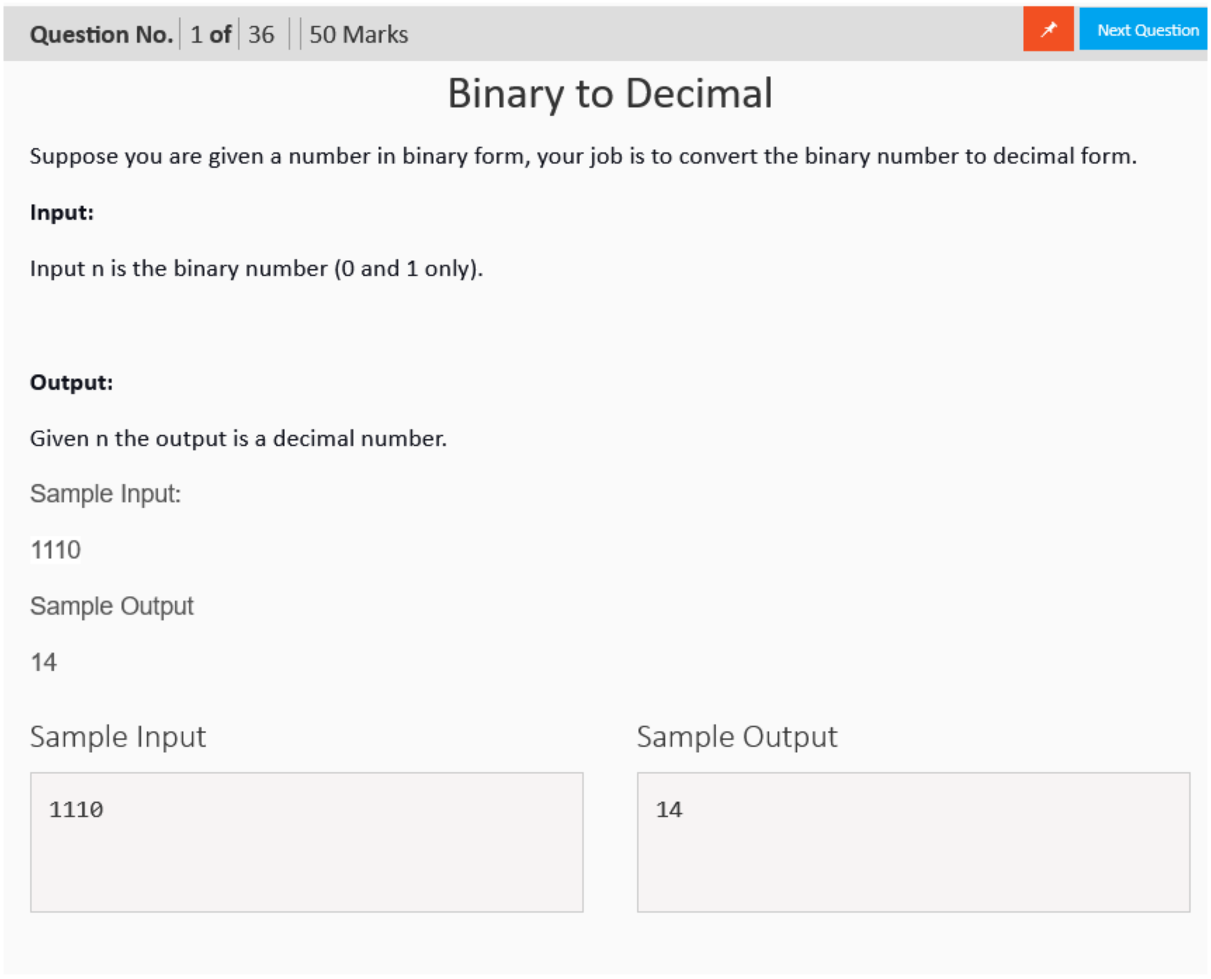


Binary to Decimal

Suppose you are given a number in binary form, your job is to convert the binary number to decimal form.

Input:

Input n is the binary number (0 and 1 only).

Output:

Given n the output is a decimal number.

Sample Input:

1110

Sample Output

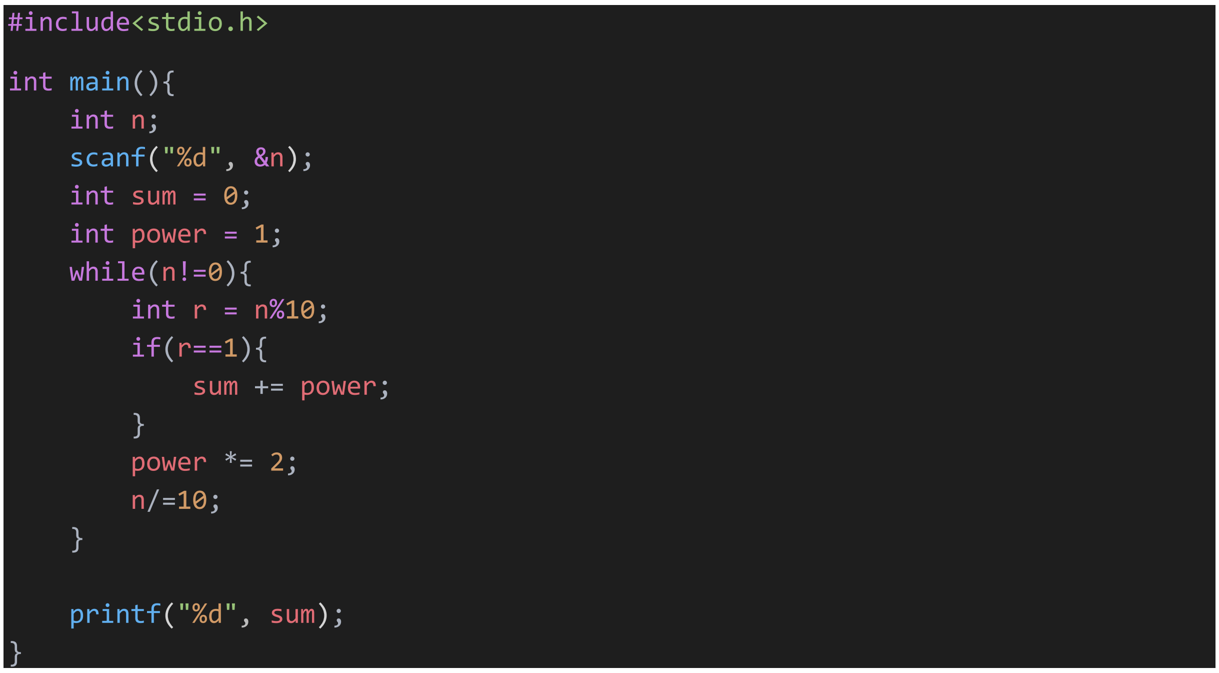
14

Sample Input

```
1110
```

Sample Output

```
14
```



```c
#include<stdio.h>

int main(){
    int n;
    scanf("%d", &n);
    int sum = 0;
    int power = 1;
    while(n!=0){
        int r = n%10;
        if(r==1){
            sum += power;
        }
        power *= 2;
        n/=10;
    }

    printf("%d", sum);
}
```

Check Array Rotation

You have been given an integer array (ARR) of size N. It has been sorted (in increasing order) and then rotated by some number 'K' in the right-hand direction.

Your task is to write a function that returns the value of 'K', which means, the index from which the array/list has been rotated.

Input format :

The first line contains an integer 'N' representing the size of the array/list.

The second line contains 'N' single space-separated integers representing the elements in the array.

Output Format :

Print the value of 'K' or the index from which the array has been rotated.

The output will be printed in a separate line.

Constraints :

0 <= N <= 10^5

Sample Input 1:

6

5 6 1 2 3 4

Sample Output 1:

2

Sample Input 2:

5

3 6 8 9 10

Sample Output 2:

0

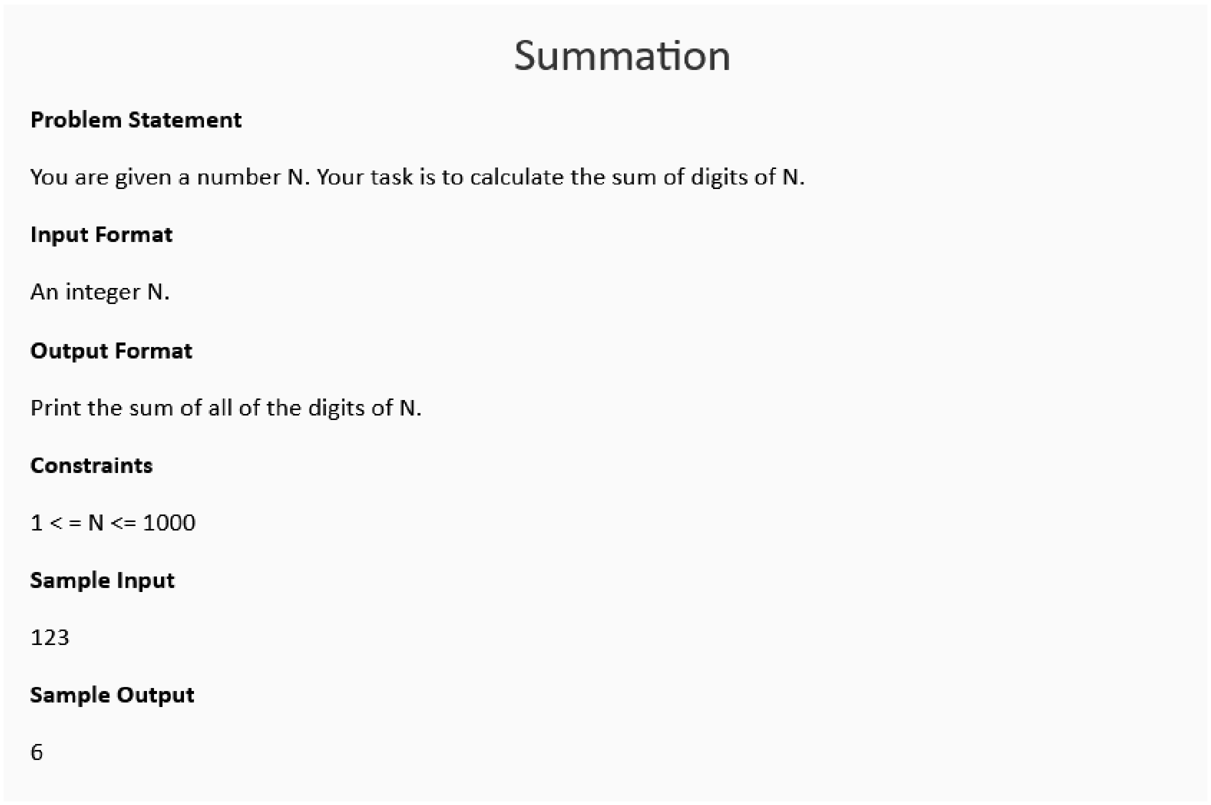
Sample Input

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

Sample Output

```
2
```

```c
#include<stdio.h>

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

    int count = 1;
    for(i=1; i<n; i++){
        if(arr[i] < arr[i-1]) break;
        count++;
    }
    if(count == n) count = 0;

    printf("%d", count);
}
```



Summation

Problem Statement

You are given a number N. Your task is to calculate the sum of digits of N.

Input Format

An integer N.

Output Format

Print the sum of all of the digits of N.

Constraints

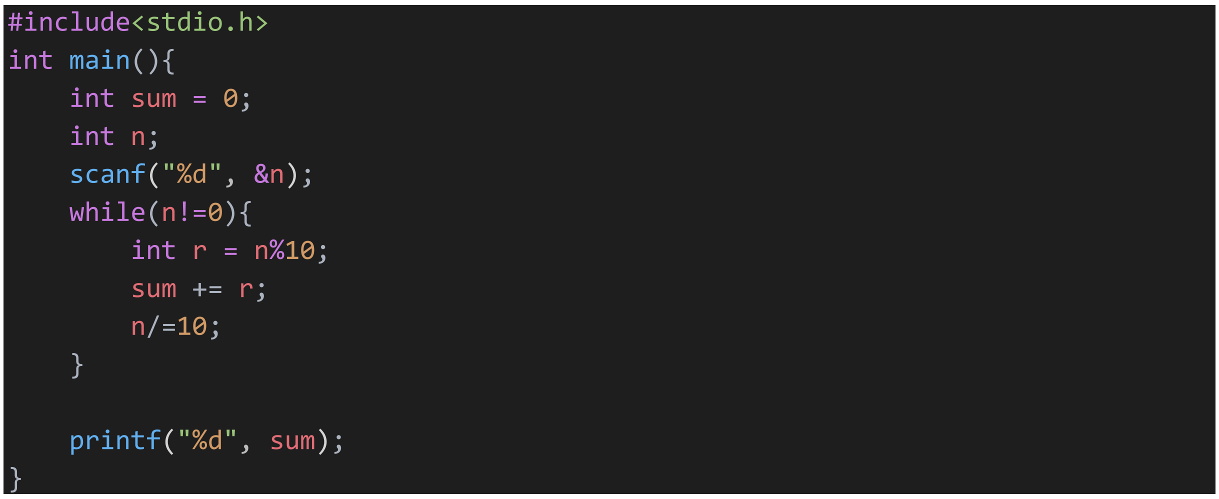
1 < = N <= 1000

Sample Input

123

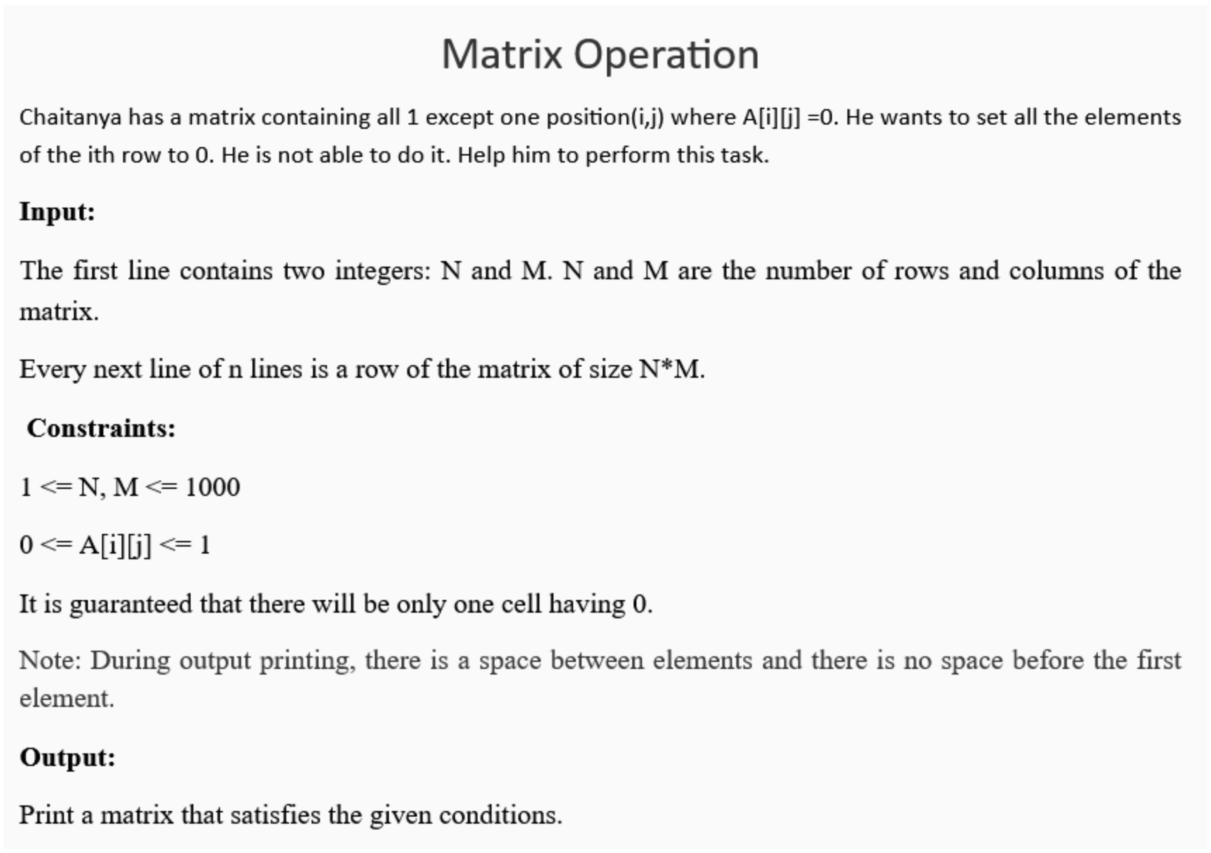
Sample Output

6



```c
#include<stdio.h>
int main(){
    int sum = 0;
    int n;
    scanf("%d", &n);
    while(n!=0){
        int r = n%10;
        sum += r;
        n/=10;
    }

    printf("%d", sum);
}
```

4.



Matrix Operation

Chaitanya has a matrix containing all 1 except one position(i,j) where A[i][j] =0. He wants to set all the elements of the ith row to 0. He is not able to do it. Help him to perform this task.

Input:

The first line contains two integers: N and M. N and M are the number of rows and columns of the matrix.

Every next line of n lines is a row of the matrix of size N*M.

Constraints:

1 <= N, M <= 1000

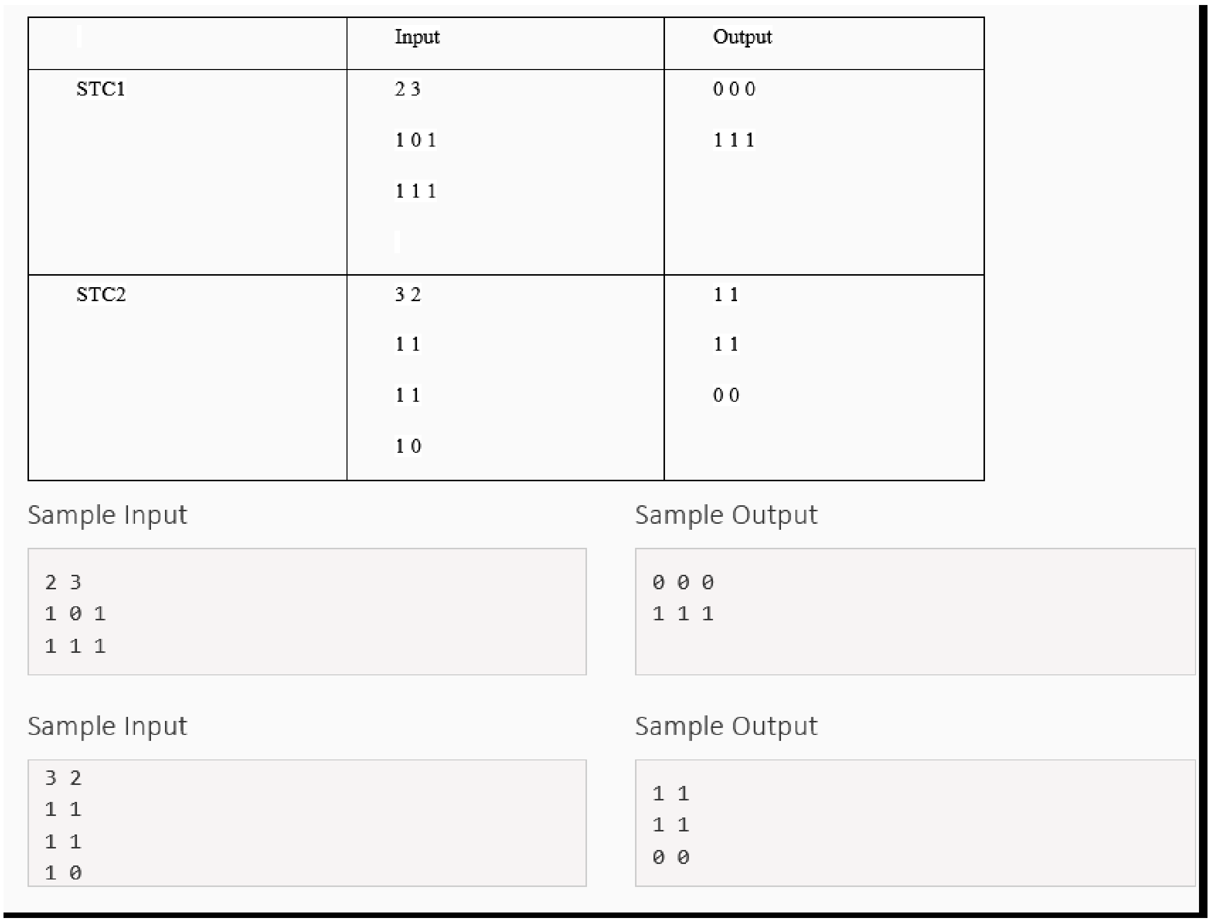
0 <= A[i][j] <= 1

It is guaranteed that there will be only one cell having 0.

Note: During output printing, there is a space between elements and there is no space before the first element.

Output:

Print a matrix that satisfies the given conditions.



	Input	Output

STC1	2 3
	1 0 1
	1 1 1
		0 0 0
		1 1 1

STC2	3 2
	1 1
	1 1
	1 0
		1 1
		1 1
		0 0

Sample Input

```
2 3
1 0 1
1 1 1
```

Sample Output

```
0 0 0
1 1 1
```

Sample Input

```
3 2
1 1
1 1
1 0
```

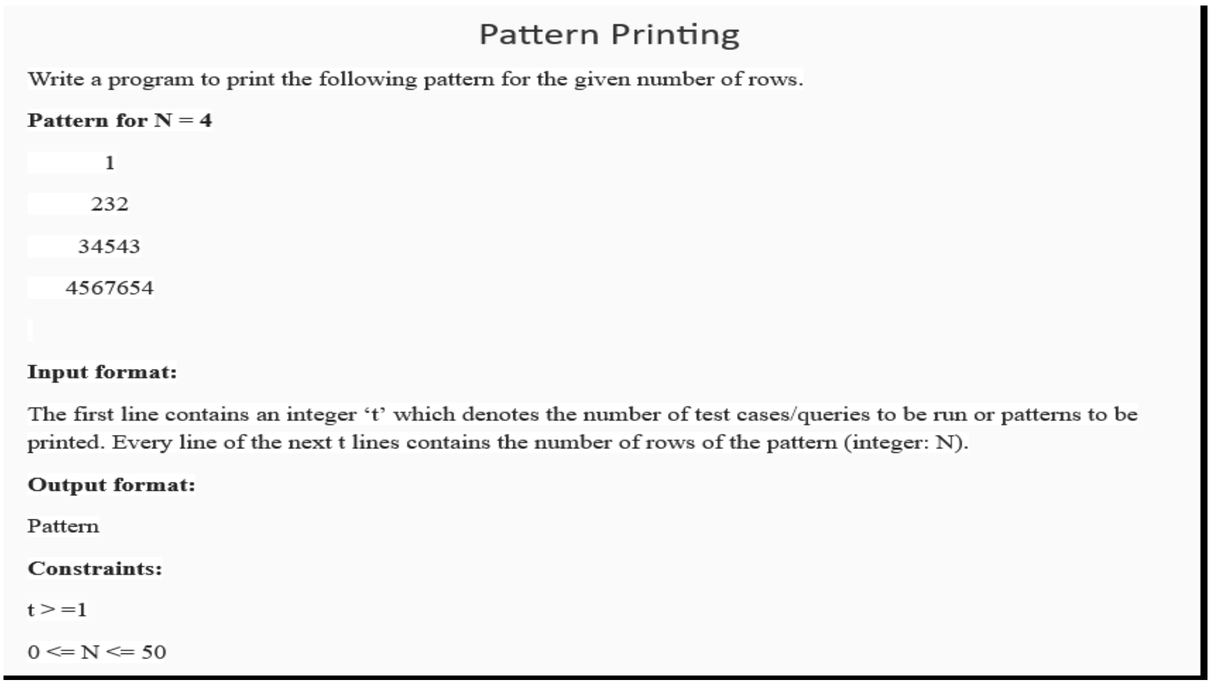
Sample Output

```
1 1
1 1
0 0
```

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

```c
int main(){
    int n, m;
    scanf("%d" ,&n);
    scanf("%d" ,&m);

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

    for(i=0; i<n; i++){
        for(j=0; j<m; j++){
            if(arr[i][j] == 0){
                for(j=0; j<m; j++){
                    arr[i][j] = 0;
                }
                break;
            }
        }

    }

    for(i=0; i<n; i++){
        for(j=0; j<m; j++){
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }


}
```

5.



Pattern Printing

Write a program to print the following pattern for the given number of rows.

Pattern for N = 4

```
      1
     232
    34543
   4567654
```

Input format:

The first line contains an integer 't' which denotes the number of test cases/queries to be run or patterns to be printed. Every line of the next t lines contains the number of rows of the pattern (integer: N).

Output format:

Pattern

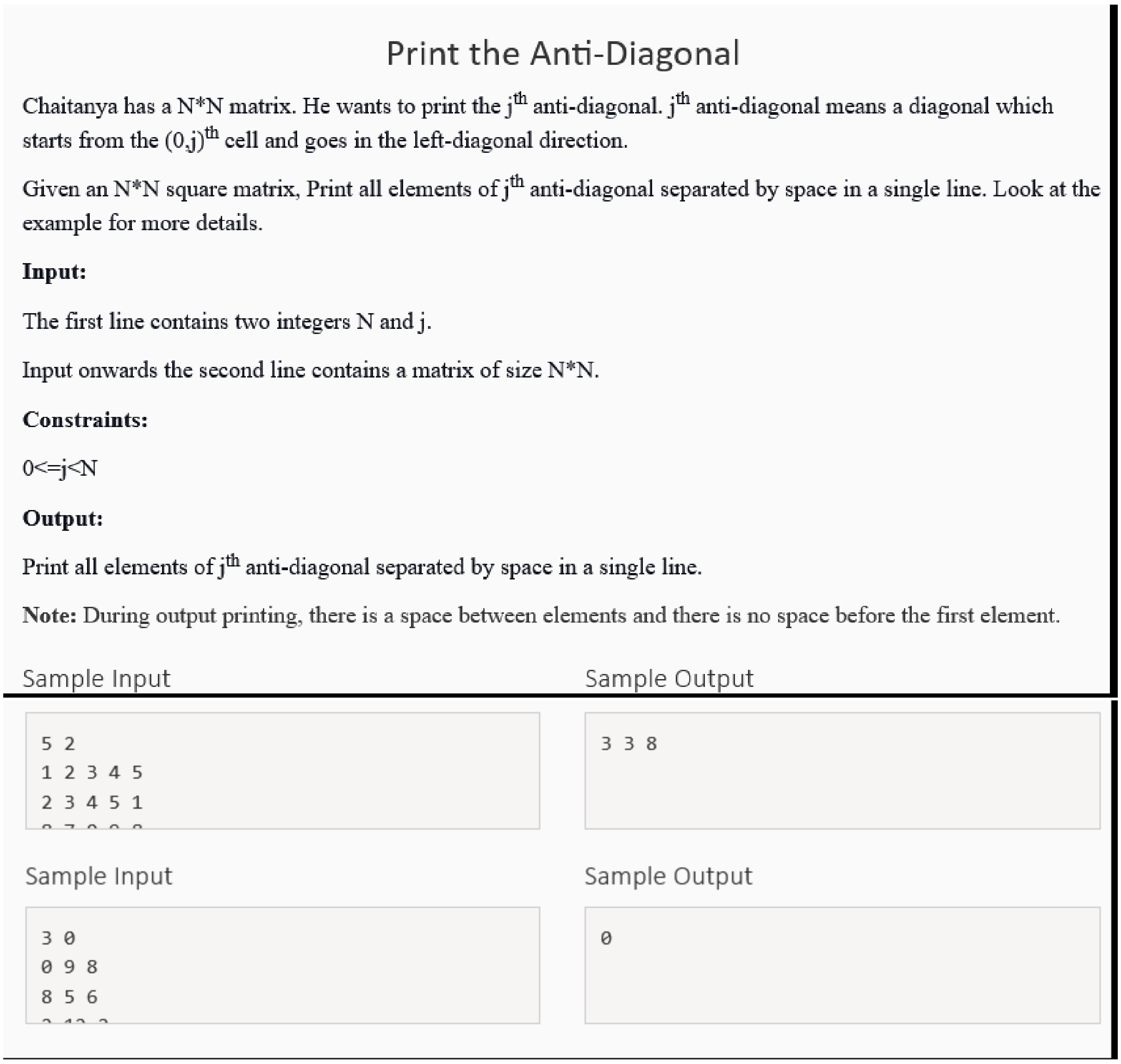
Constraints:

t >=1

0 <= N <= 50

```c
#include <stdio.h>

int main() {
    int t;
    scanf("%d", &t);
    while(t--){
        int n, i, j, k;
        scanf("%d", &n);
        for(i=1; i<=n; i++){
            for(j=1; j<=n-i; j++){
                printf(" ");
            }
            k = i;
            for(j=1; j<=i; j++){
                printf("%d", k);
                k++;
            }
            k -= 2;
            for(j=1; j<i; j++){
                printf("%d", k);
                k--;
            }
            printf("\n");
        }
    }
}
```

6.



Print the Anti-Diagonal

Chaitanya has a N*N matrix. He wants to print the jth anti-diagonal. jth anti-diagonal means a diagonal which starts from the (0,j)th cell and goes in the left-diagonal direction.

Given an N*N square matrix, Print all elements of jth anti-diagonal separated by space in a single line. Look at the example for more details.

Input:

The first line contains two integers N and j.

Input onwards the second line contains a matrix of size N*N.

Constraints:

0<=j<N

Output:

Print all elements of jth anti-diagonal separated by space in a single line.

Note: During output printing, there is a space between elements and there is no space before the first element.

Sample Input

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

Sample Output

```
3 3 8
```

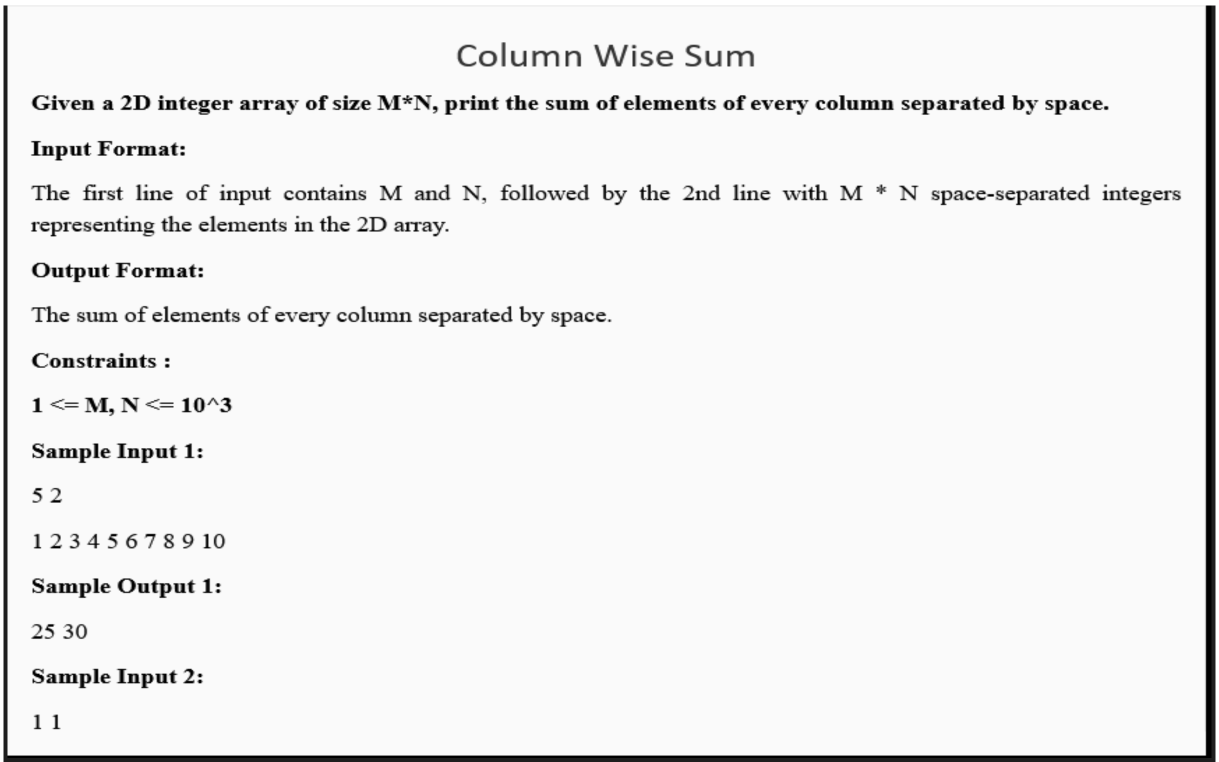
Sample Input

```
3 0
0 9 8
8 5 6
```

Sample Output

```
0
```

```c
#include<stdio.h>
int main(){
    int n;
    scanf("%d", &n);
    int m;
    scanf("%d", &m);


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

    i = 0;
    j = m;
    m++;
    while(m!=0){
        printf("%d ", arr[i][j]);
        m--;
        j--;
        i++;
    }
}
```



Column Wise Sum

Given a 2D integer array of size M*N, print the sum of elements of every column separated by space.

Input Format:

The first line of input contains M and N, followed by the 2nd line with M * N space-separated integers representing the elements in the 2D array.

Output Format:

The sum of elements of every column separated by space.

Constraints :

1 <= M, N <= 10^3

Sample Input 1:

5 2

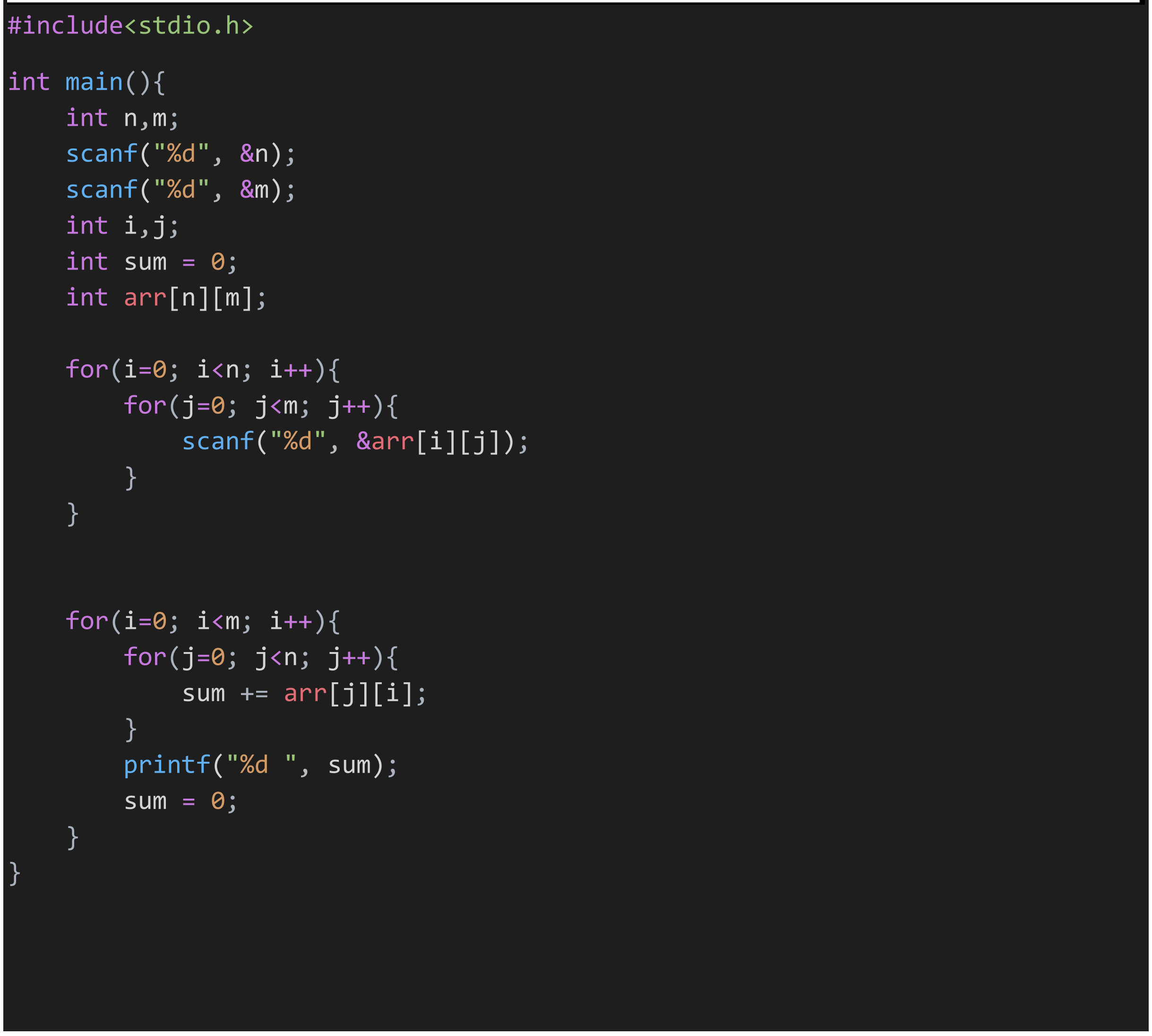
1 2 3 4 5 6 7 8 9 10

Sample Output 1:

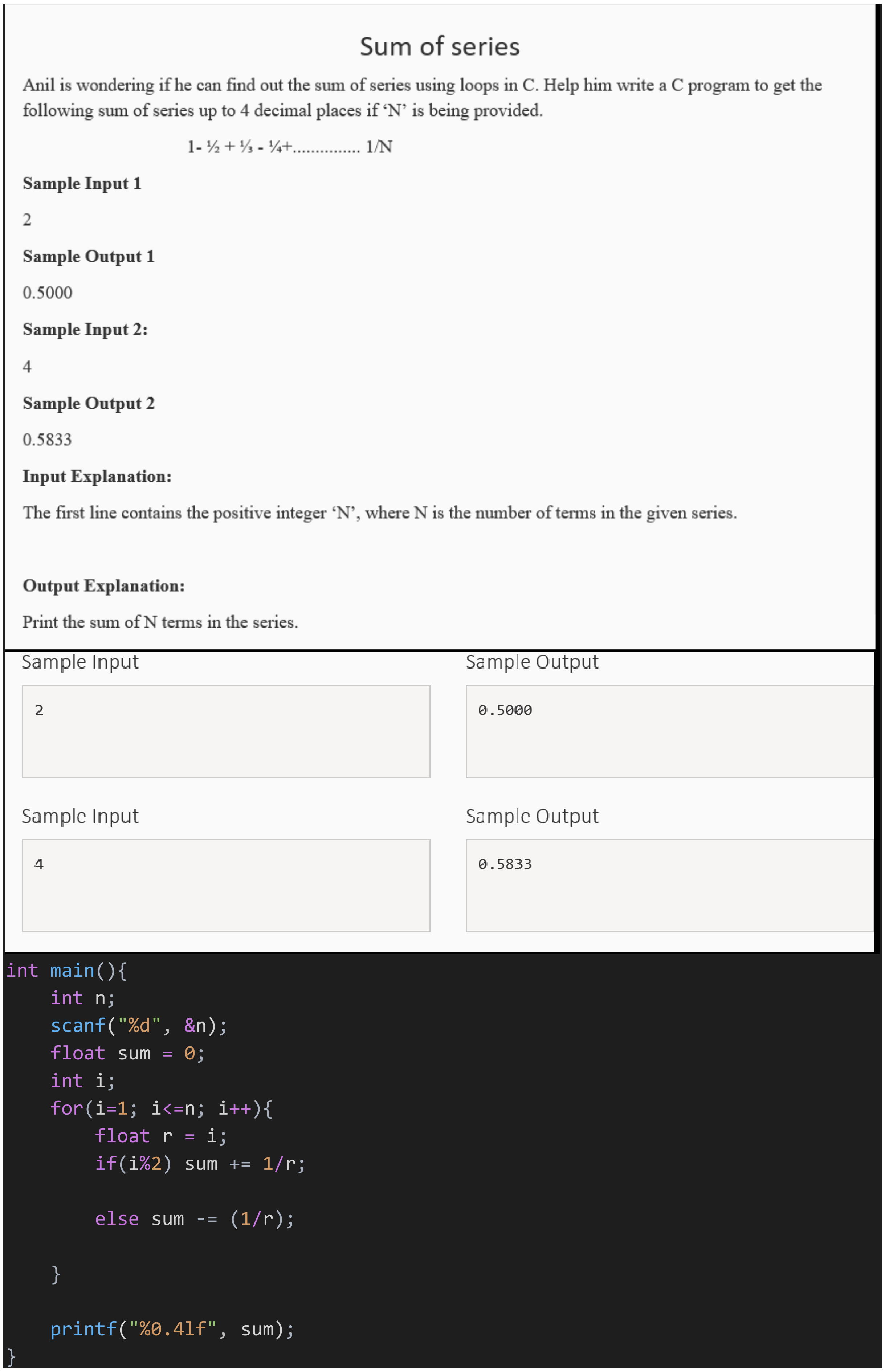
25 30

Sample Input 2:

1 1



```c
#include<stdio.h>

int main(){
    int n,m;
    scanf("%d", &n);
    scanf("%d", &m);
    int i,j;
    int sum = 0;
    int arr[n][m];

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


    for(i=0; i<m; i++){
        for(j=0; j<n; j++){
            sum += arr[j][i];
        }
        printf("%d ", sum);
        sum = 0;
    }
}
```



Sum of series

Anil is wondering if he can find out the sum of series using loops in C. Help him write a C program to get the following sum of series up to 4 decimal places if 'N' is being provided.

1- ½ + ⅓ - ¼+............... 1/N

Sample Input 1

2

Sample Output 1

0.5000

Sample Input 2:

4

Sample Output 2

0.5833

Input Explanation:

The first line contains the positive integer 'N', where N is the number of terms in the given series.

Output Explanation:

Print the sum of N terms in the series.

Sample Input

```
2
```

Sample Output

```
0.5000
```

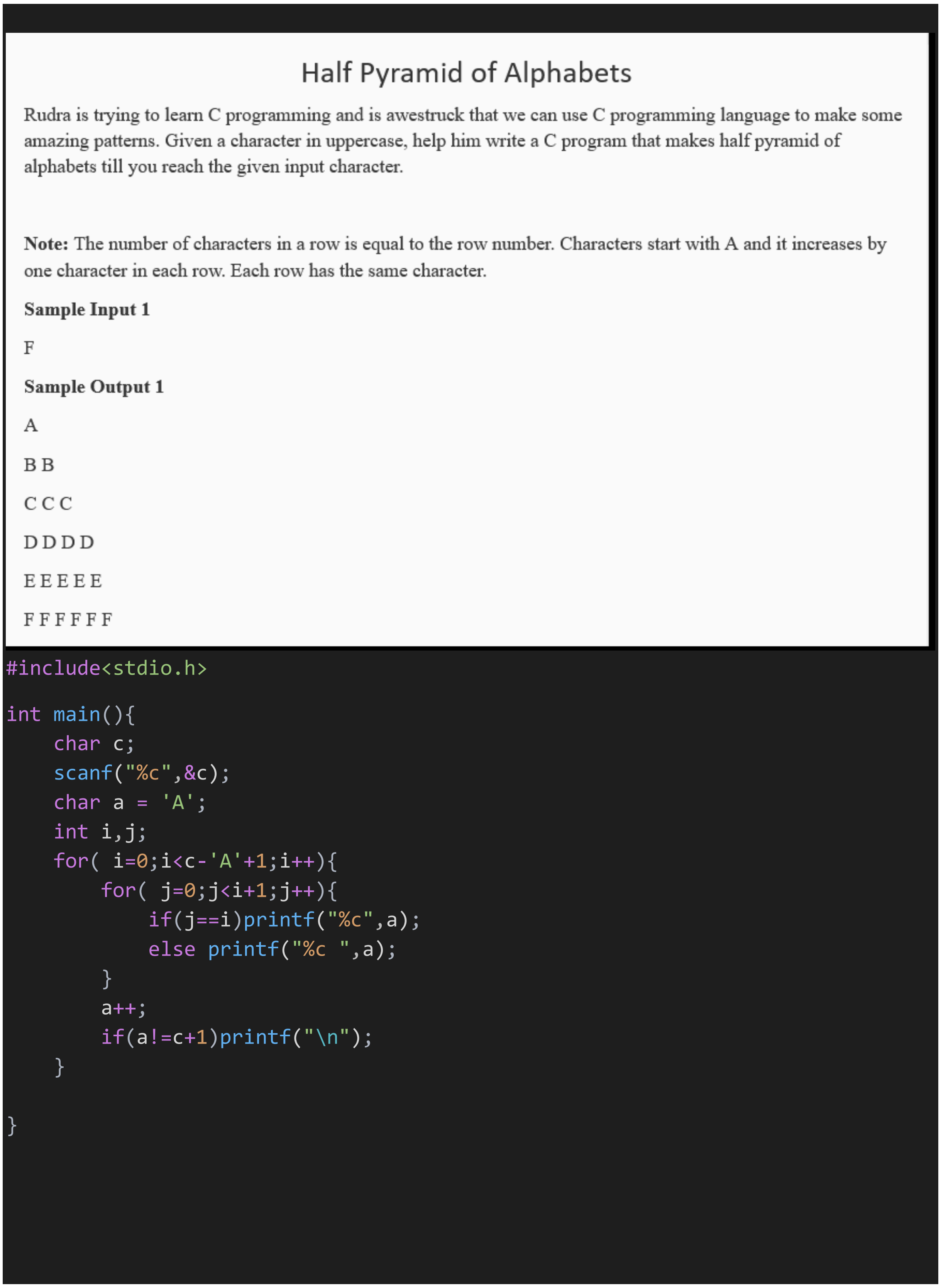
Sample Input

```
4
```

Sample Output

```
0.5833
```

```c
int main(){
    int n;
    scanf("%d", &n);
    float sum = 0;
    int i;
    for(i=1; i<=n; i++){
        float r = i;
        if(i%2) sum += 1/r;

        else sum -= (1/r);

    }

    printf("%0.4lf", sum);
}
```



Half Pyramid of Alphabets

Rudra is trying to learn C programming and is awestruck that we can use C programming language to make some amazing patterns. Given a character in uppercase, help him write a C program that makes half pyramid of alphabets till you reach the given input character.

Note: The number of characters in a row is equal to the row number. Characters start with A and it increases by one character in each row. Each row has the same character.

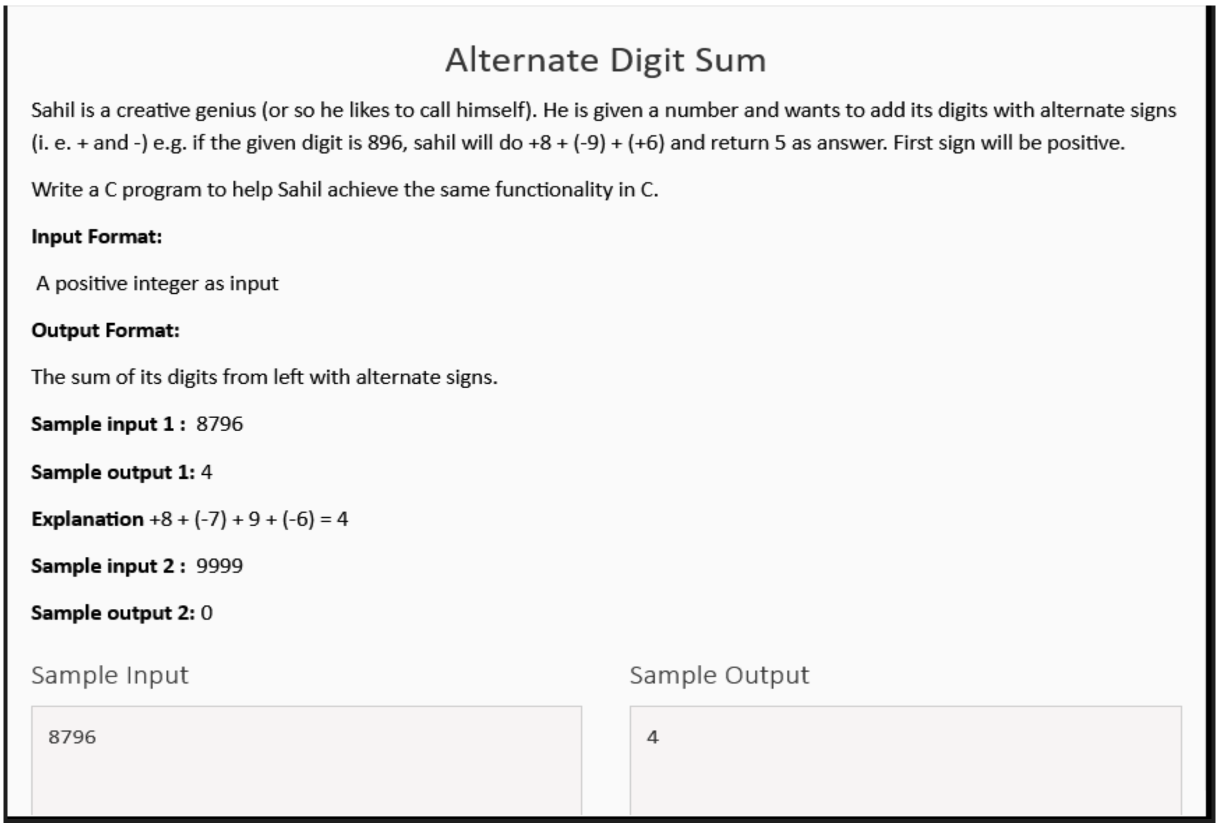
Sample Input 1

```
F
```

Sample Output 1

```
A

B B

C C C

D D D D

E E E E E

F F F F F F
```

```c
#include<stdio.h>

int main(){
    char c;
    scanf("%c",&c);
    char a = 'A';
    int i,j;
    for( i=0;i<c-'A'+1;i++){
        for( j=0;j<i+1;j++){
            if(j==i)printf("%c",a);
            else printf("%c ",a);
        }
        a++;
        if(a!=c+1)printf("\n");
    }

}
```



Alternate Digit Sum

Sahil is a creative genius (or so he likes to call himself). He is given a number and wants to add its digits with alternate signs (i. e. + and -) e.g. if the given digit is 896, sahil will do +8 + (-9) + (+6) and return 5 as answer. First sign will be positive.

Write a C program to help Sahil achieve the same functionality in C.

Input Format:

A positive integer as input

Output Format:

The sum of its digits from left with alternate signs.

Sample input 1 : 8796

Sample output 1: 4

Explanation +8 + (-7) + 9 + (-6) = 4

Sample input 2 : 9999

Sample output 2: 0

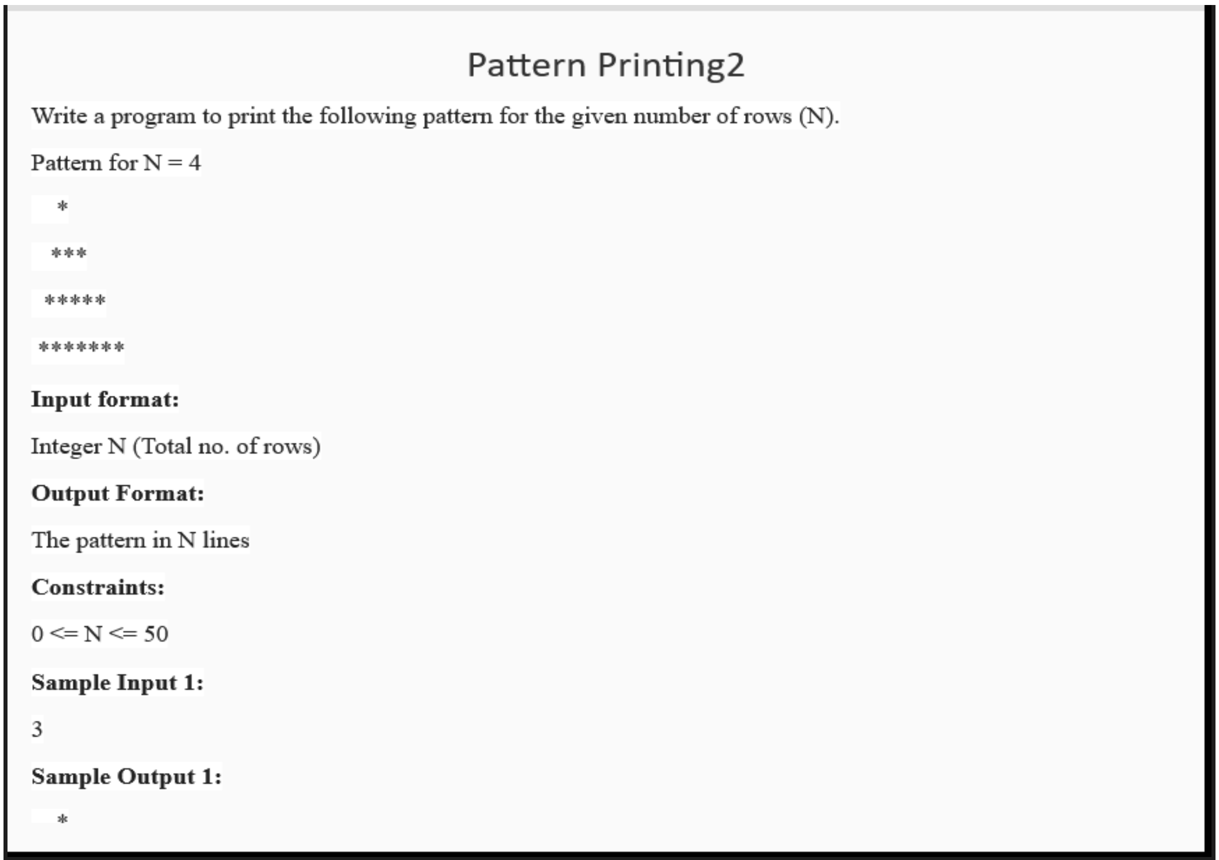
Sample Input

```
8796
```

Sample Output

```
4
```

```c
#include <stdio.h>

#include <stdlib.h>

int main() {
    int num, sum = 0, sign = 1;

    scanf("%d", &num);
    int rev = 0;
    while(num!=0){
        int rem = num%10;
        rev = rev * 10 + rem;
        num /= 10;
    }
    // printf("%d ", rev);
    num = rev;

    while (num != 0) {
        int digit = num % 10;
        sum += sign * digit;
        sign = -sign;
        num /= 10;
    }

    printf("%d", sum);

    return 0;
}
```



Pattern Printing2

Write a program to print the following pattern for the given number of rows (N).

Pattern for N = 4

```
   *
  ***
 *****
*******
```

Input format:

Integer N (Total no. of rows)

Output Format:

The pattern in N lines

Constraints:

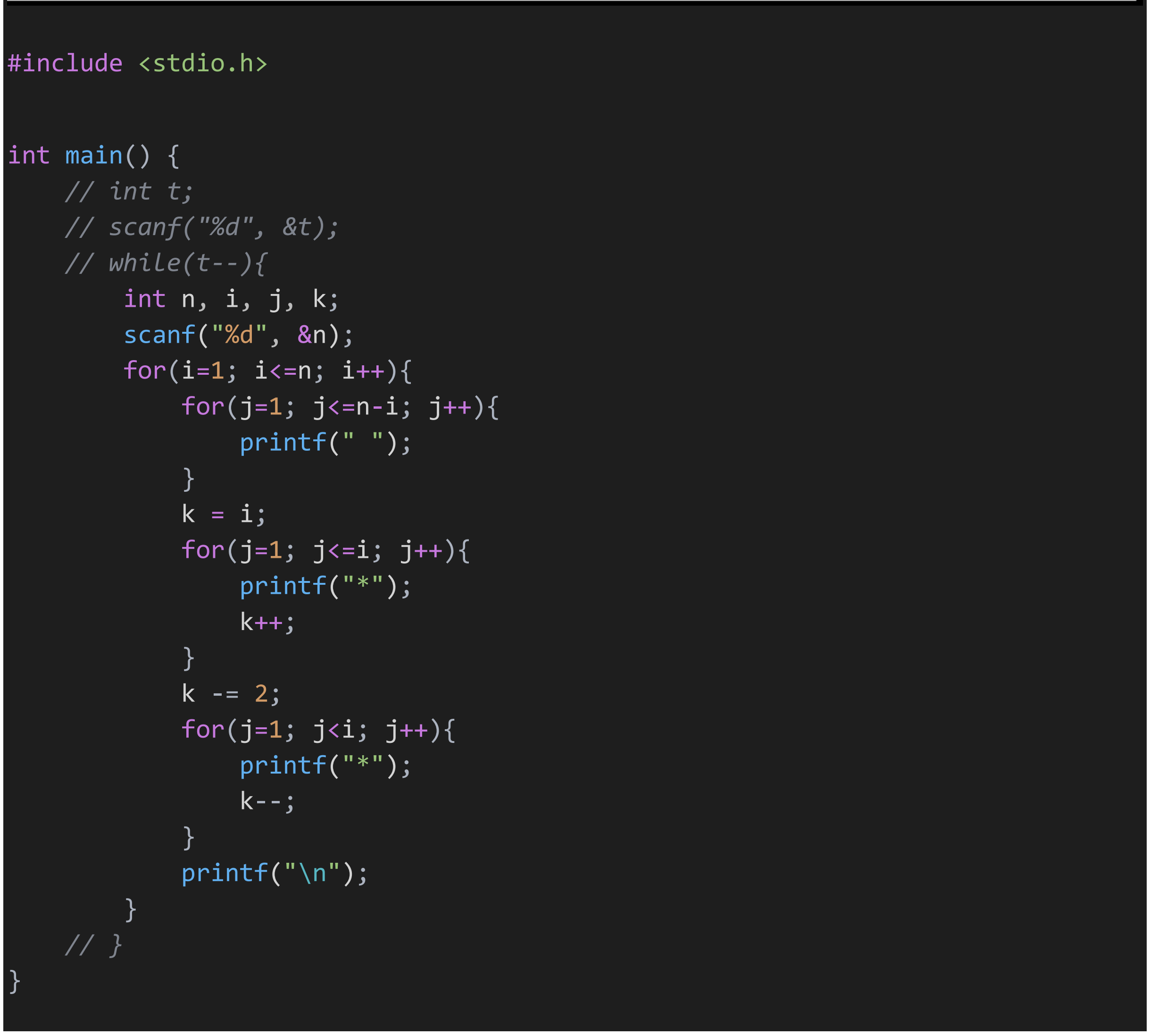
0 <= N <= 50

Sample Input 1:

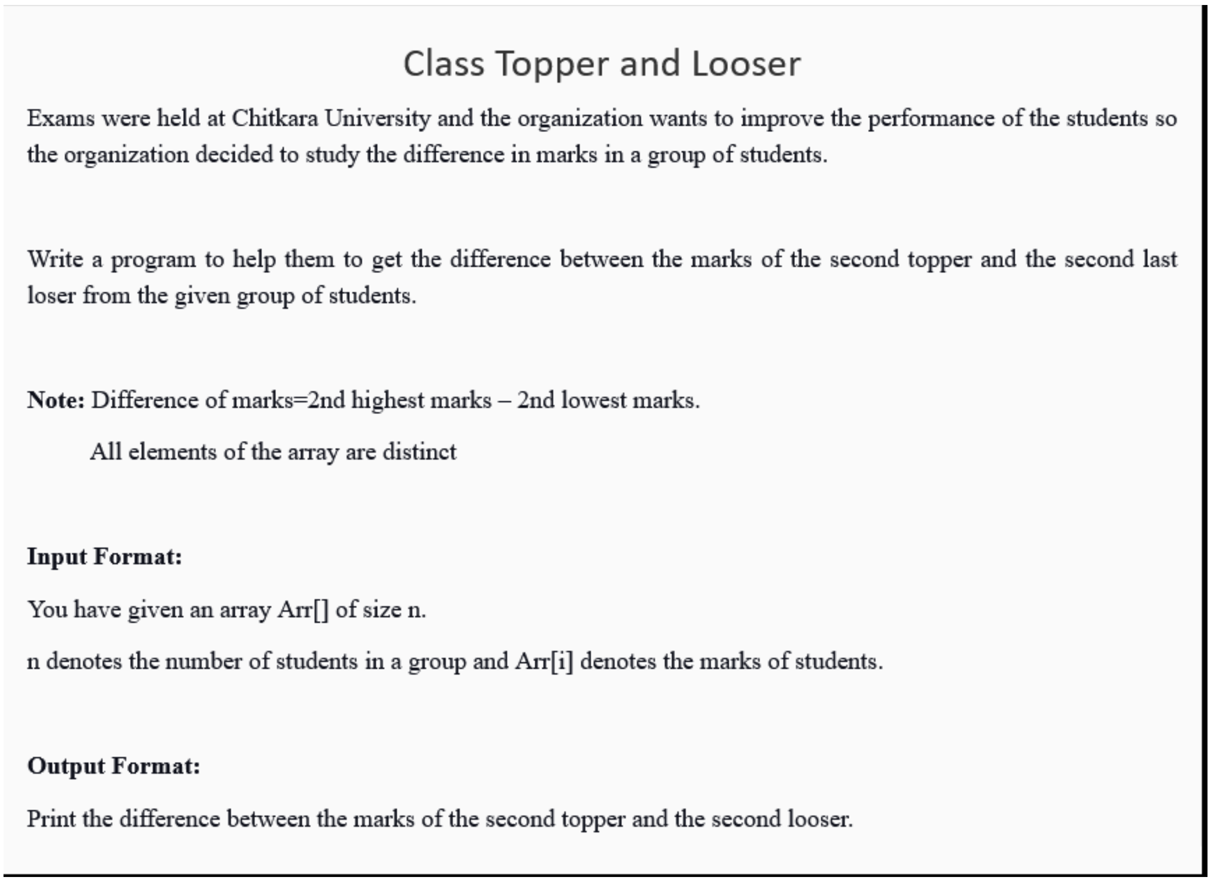
3

Sample Output 1:

```
  *
```



```c
#include <stdio.h>

int main() {
    // int t;
    // scanf("%d", &t);
    // while(t--){
        int n, i, j, k;
        scanf("%d", &n);
        for(i=1; i<=n; i++){
            for(j=1; j<=n-i; j++){
                printf(" ");
            }
            k = i;
            for(j=1; j<=i; j++){
                printf("*");
                k++;
            }
            k -= 2;
            for(j=1; j<i; j++){
                printf("*");
                k--;
            }
            printf("\n");
        }
    // }
}
```



Class Topper and Looser

Exams were held at Chitkara University and the organization wants to improve the performance of the students so the organization decided to study the difference in marks in a group of students.

Write a program to help them to get the difference between the marks of the second topper and the second last loser from the given group of students.

Note: Difference of marks=2nd highest marks – 2nd lowest marks.

All elements of the array are distinct

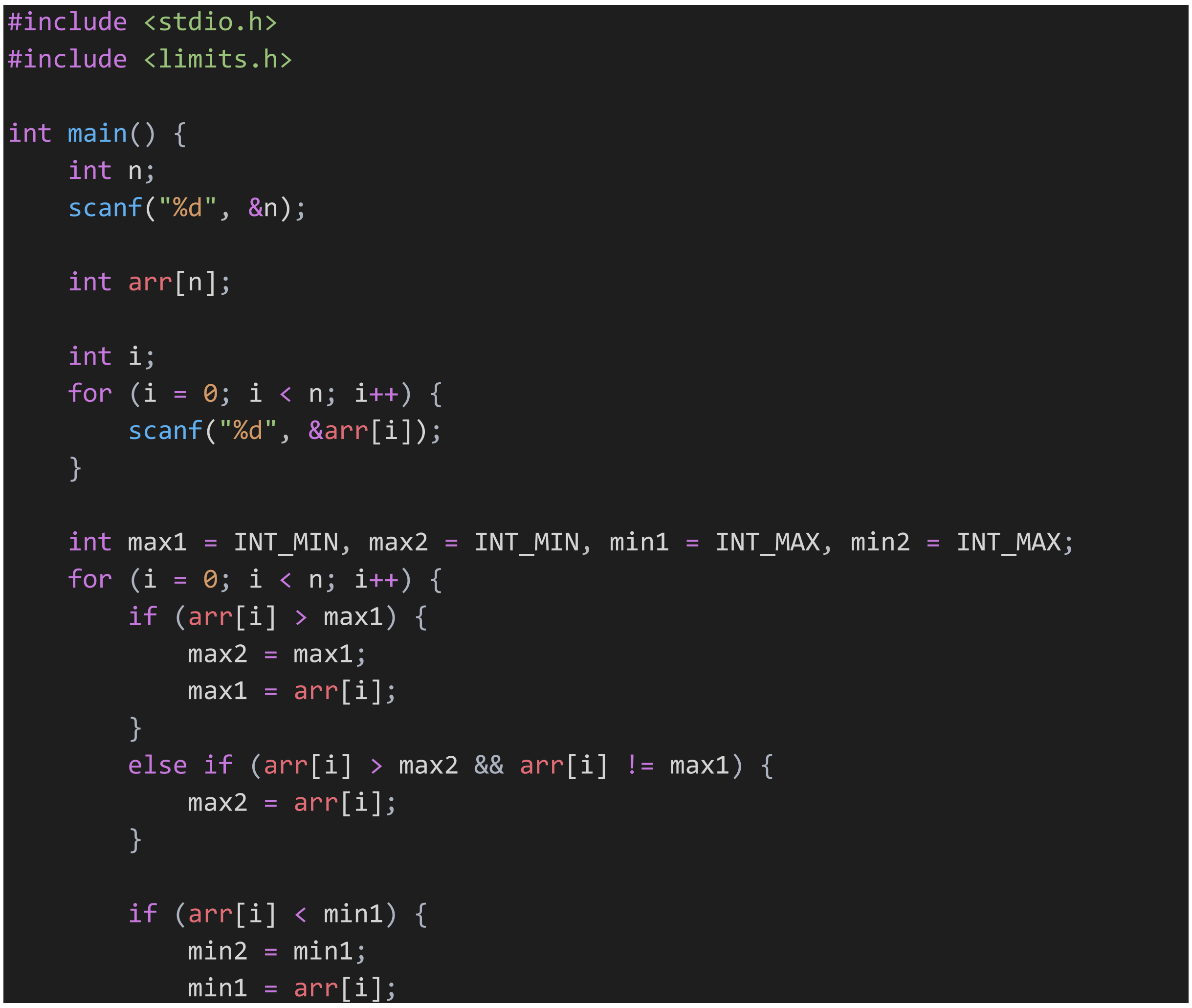
Input Format:

You have given an array Arr[] of size n.

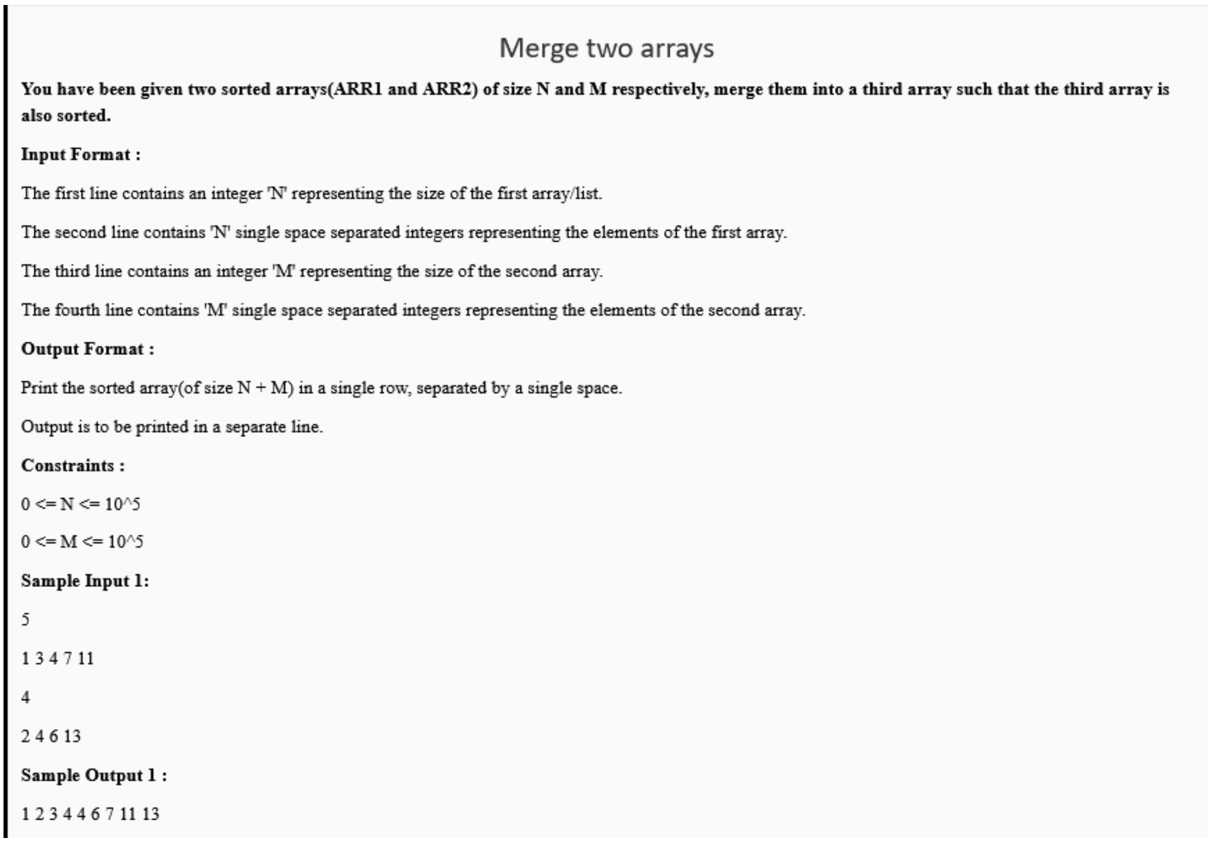
n denotes the number of students in a group and Arr[i] denotes the marks of students.

Output Format:

Print the difference between the marks of the second topper and the second looser.



```c
#include <stdio.h>
#include <limits.h>

int main() {
    int n;
    scanf("%d", &n);

    int arr[n];

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

    int max1 = INT_MIN, max2 = INT_MIN, min1 = INT_MAX, min2 = INT_MAX;
    for (i = 0; i < n; i++) {
        if (arr[i] > max1) {
            max2 = max1;
            max1 = arr[i];
        }
        else if (arr[i] > max2 && arr[i] != max1) {
            max2 = arr[i];
        }

        if (arr[i] < min1) {
            min2 = min1;
            min1 = arr[i];
        }
        else if (arr[i] < min2 && arr[i] != min1) {
            min2 = arr[i];
        }
    }

    printf("%d", max2 - min2);
    // printf("%d", min2);

    return 0;
}
```



Merge two arrays

You have been given two sorted arrays(ARR1 and ARR2) of size N and M respectively, merge them into a third array such that the third array is also sorted.

Input Format :

The first line contains an integer 'N' representing the size of the first array/list.

The second line contains 'N' single space separated integers representing the elements of the first array.

The third line contains an integer 'M' representing the size of the second array.

The fourth line contains 'M' single space separated integers representing the elements of the second array.

Output Format :

Print the sorted array(of size N + M) in a single row, separated by a single space.

Output is to be printed in a separate line.

Constraints :

0 <= N <= 10^5

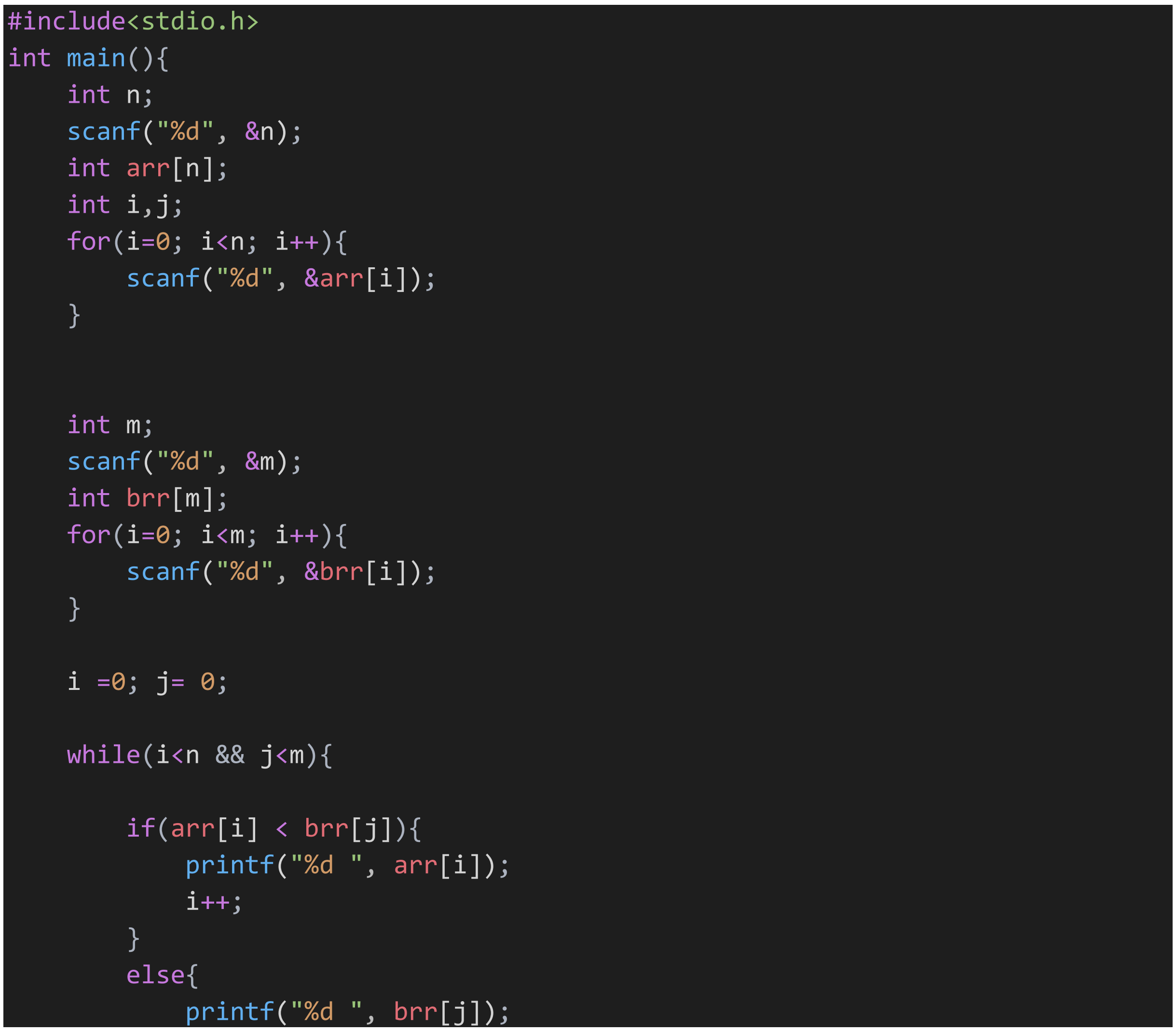
0 <= M <= 10^5

Sample Input 1:

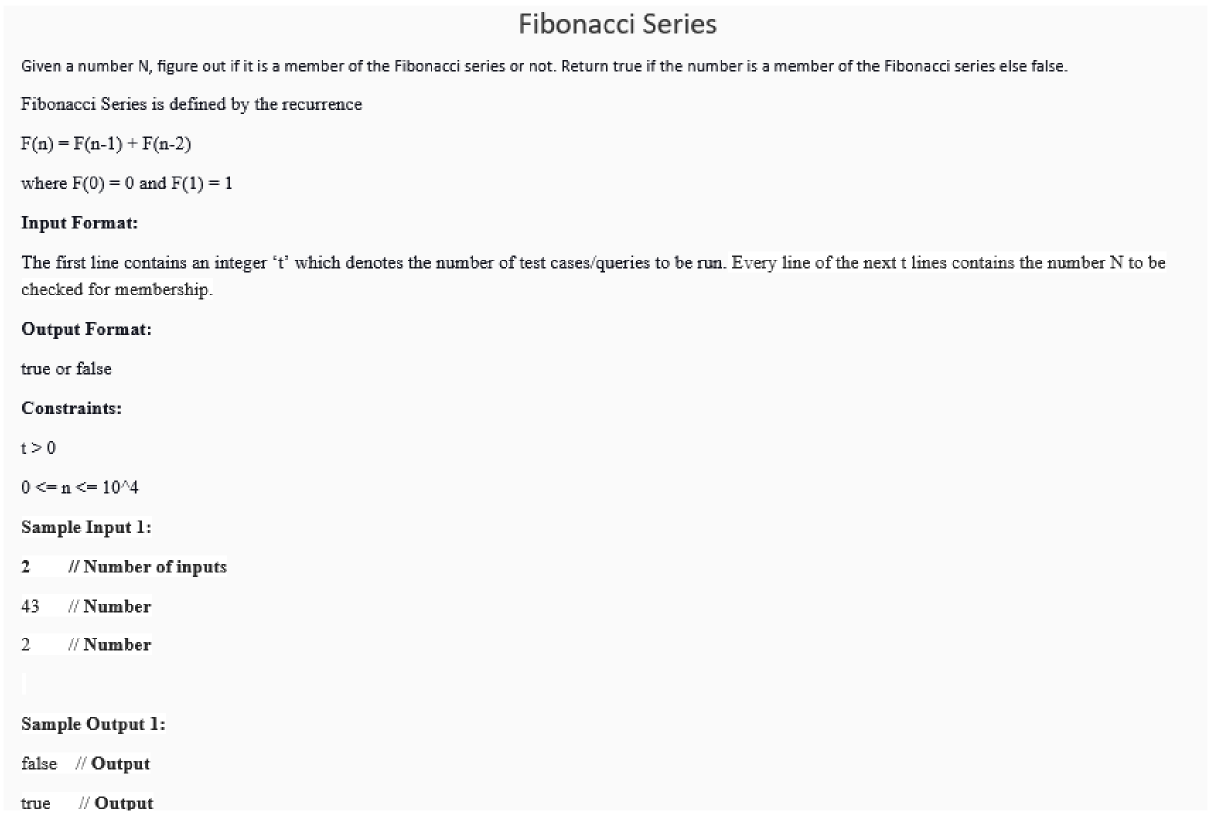
```
5
1 3 4 7 11
4
2 4 6 13
```

Sample Output 1 :

```
1 2 3 4 4 6 7 11 13
```



```c
#include<stdio.h>
int main(){
    int n;
    scanf("%d", &n);
    int arr[n];
    int i,j;
    for(i=0; i<n; i++){
        scanf("%d", &arr[i]);
    }


    int m;
    scanf("%d", &m);
    int brr[m];
    for(i=0; i<m; i++){
        scanf("%d", &brr[i]);
    }

    i =0; j= 0;

    while(i<n && j<m){

        if(arr[i] < brr[j]){
            printf("%d ", arr[i]);
            i++;
        }
        else{
            printf("%d ", brr[j]);
            j++;
        }
    }

    while(i<n){
        printf("%d ", arr[i]);
        i++;
    }

    while(j<m){
        printf("%d ", brr[j]);
        j++;
    }
}
```



Fibonacci Series

Given a number N, figure out if it is a member of the Fibonacci series or not. Return true if the number is a member of the Fibonacci series else false.

Fibonacci Series is defined by the recurrence

F(n) = F(n-1) + F(n-2)

where F(0) = 0 and F(1) = 1

Input Format:

The first line contains an integer 't' which denotes the number of test cases/queries to be run. Every line of the next t lines contains the number N to be checked for membership.

Output Format:

true or false

Constraints:

t > 0

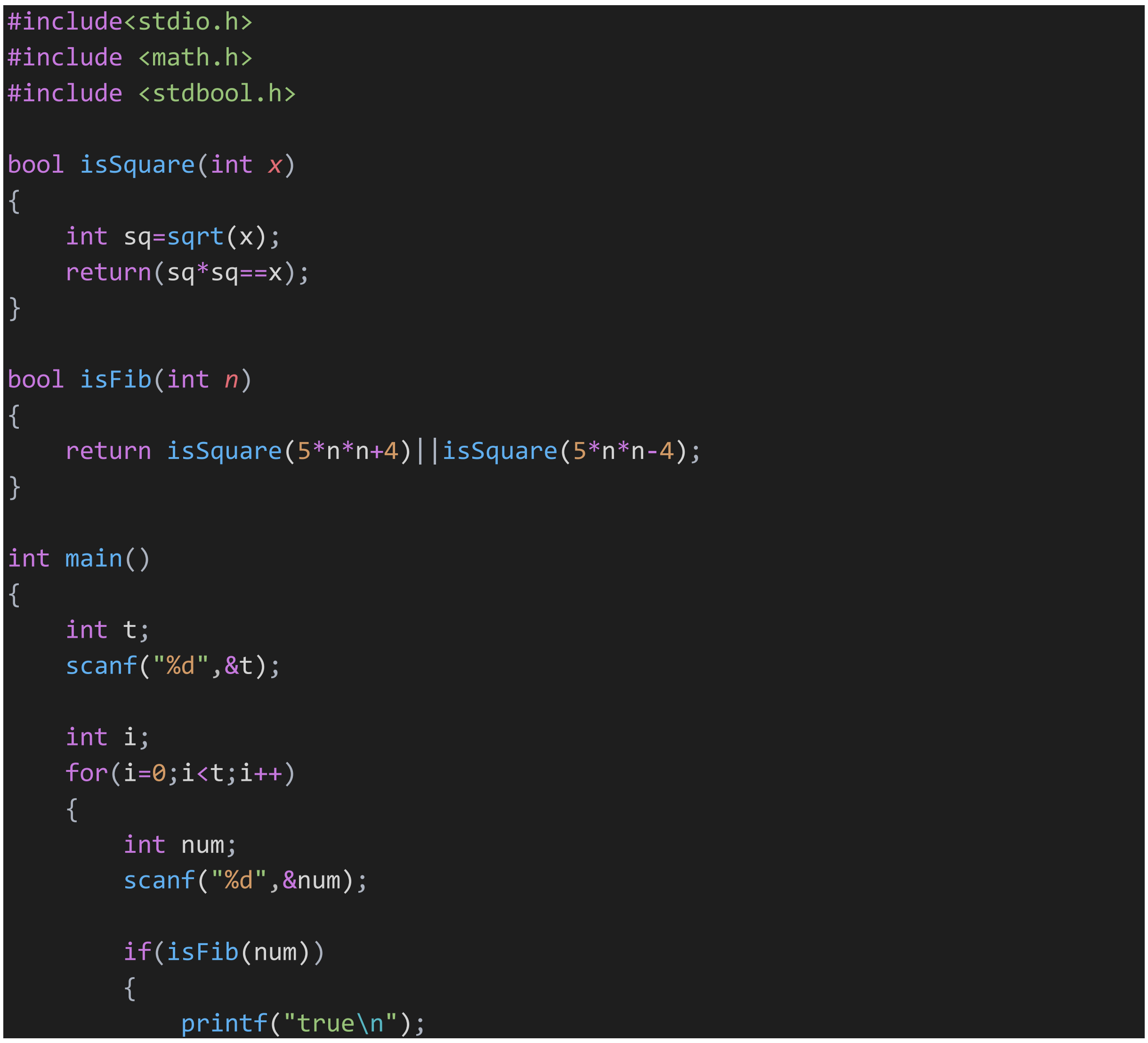
0 <= n <= 10^4

Sample Input 1:

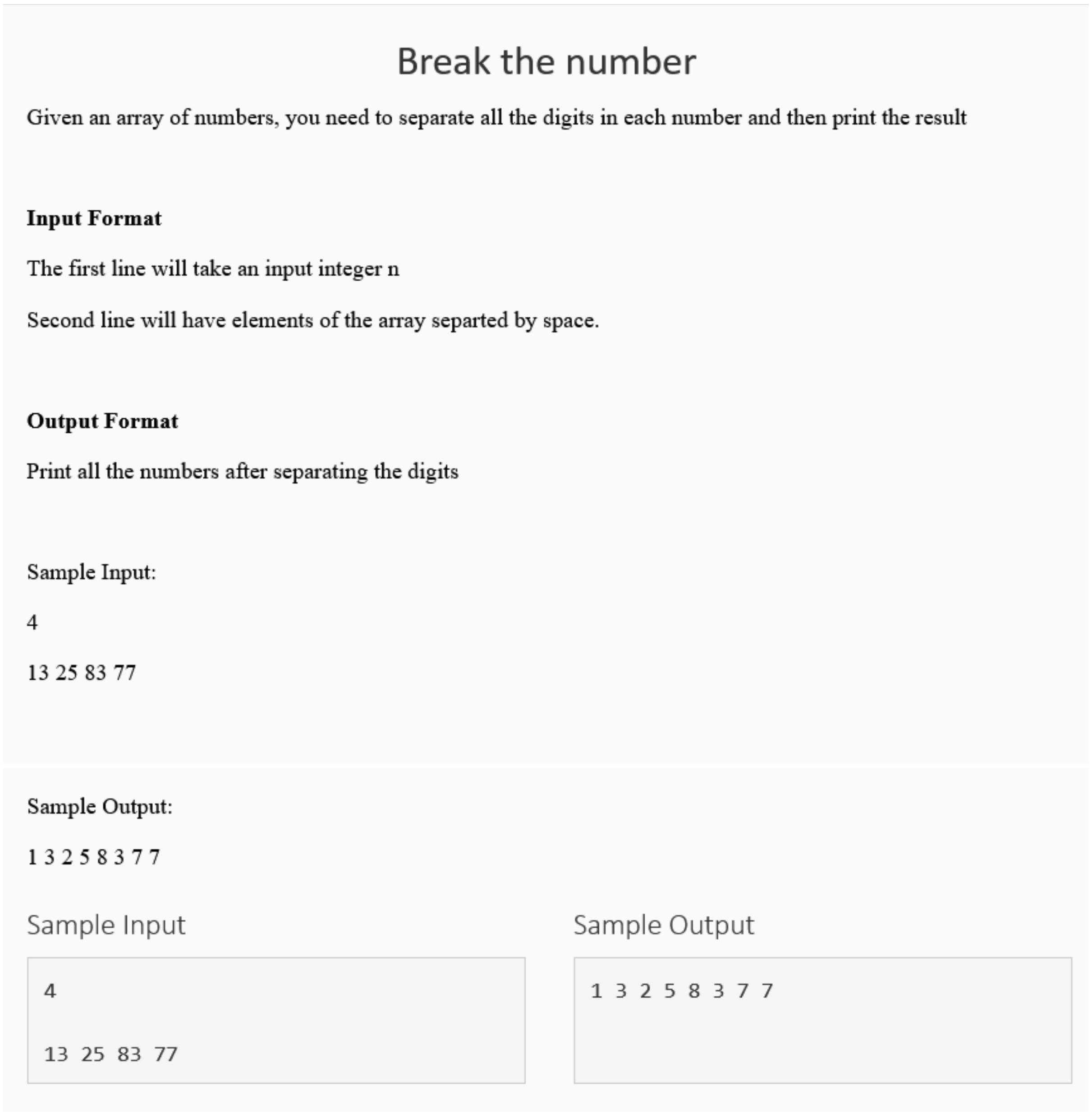
```
2      // Number of inputs
43     // Number
2      // Number
```

Sample Output 1:

```
false   // Output
true    // Output
```



```c
#include<stdio.h>
#include <math.h>
#include <stdbool.h>

bool isSquare(int x)
{
    int sq=sqrt(x);
    return(sq*sq==x);
}

bool isFib(int n)
{
    return isSquare(5*n*n+4)||isSquare(5*n*n-4);
}

int main()
{
    int t;
    scanf("%d",&t);

    int i;
    for(i=0;i<t;i++)
    {
        int num;
        scanf("%d",&num);

        if(isFib(num))
        {
            printf("true\n");
        }
        else
        {
            printf("false\n");
        }
    }
}
```



Break the number

Given an array of numbers, you need to separate all the digits in each number and then print the result

Input Format

The first line will take an input integer n

Second line will have elements of the array separparted by space.

Output Format

Print all the numbers after separating the digits

Sample Input:

4

13 25 83 77

Sample Output:

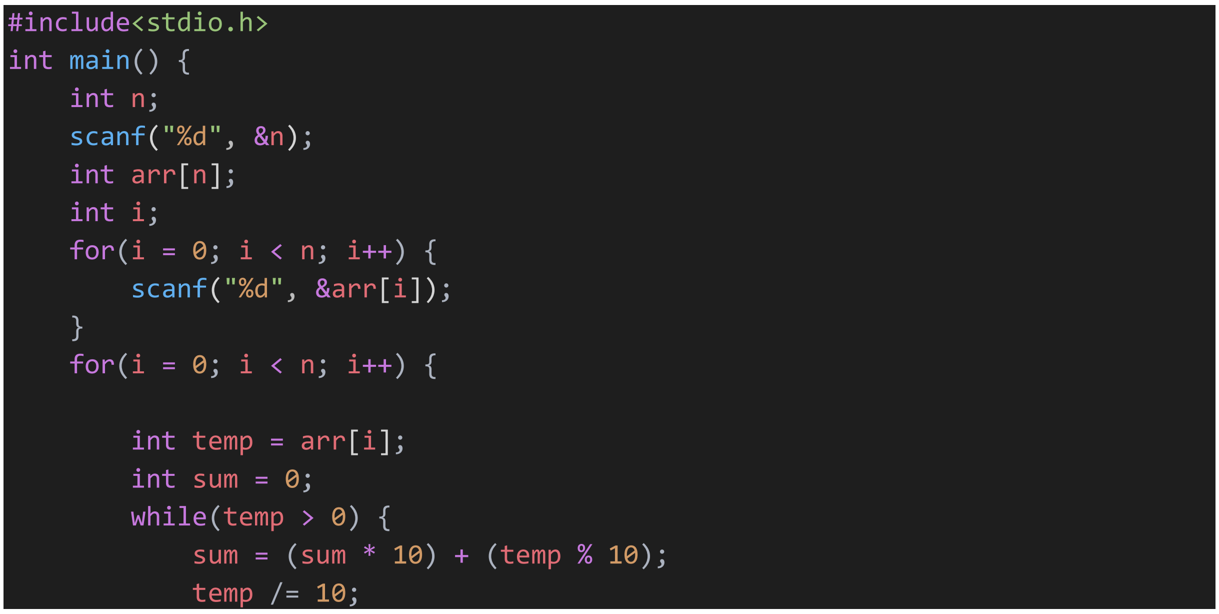
1 3 2 5 8 3 7 7

Sample Input

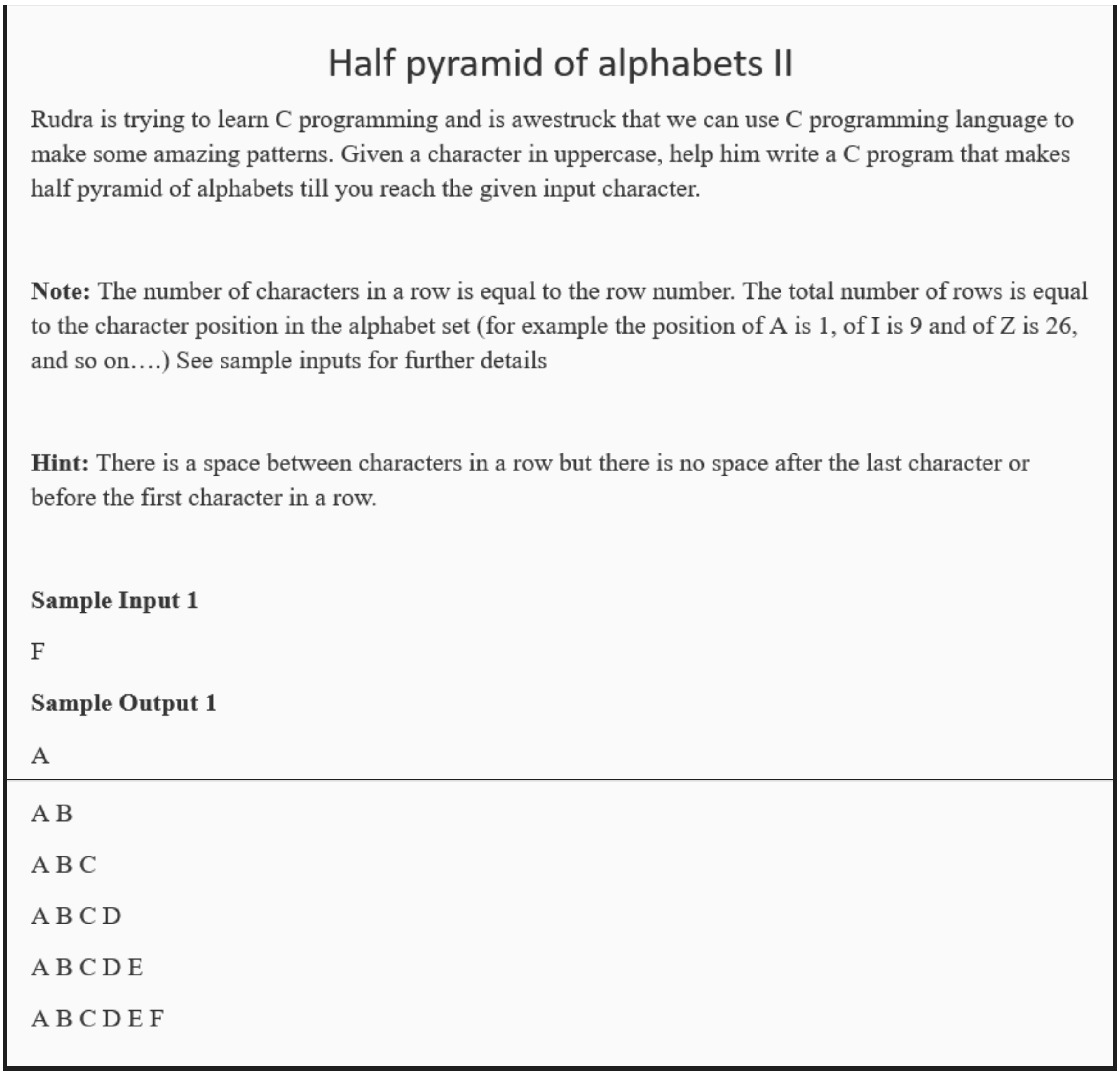
```
4

13 25 83 77
```

Sample Output

```
1 3 2 5 8 3 7 7
```



```
#include<stdio.h>
int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    int i;
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    for(i = 0; i < n; i++) {

        int temp = arr[i];
        int sum = 0;
        while(temp > 0) {
            sum = (sum * 10) + (temp % 10);
            temp /= 10;
        }
        while(sum > 0) {
            int digit = sum % 10;
            printf("%d ", digit);
            sum /= 10;
        }
    }
}
```



Half pyramid of alphabets II

Rudra is trying to learn C programming and is awestruck that we can use C programming language to make some amazing patterns. Given a character in uppercase, help him write a C program that makes half pyramid of alphabets till you reach the given input character.

Note: The number of characters in a row is equal to the row number. The total number of rows is equal to the character position in the alphabet set (for example the position of A is 1, of I is 9 and of Z is 26, and so on....) See sample inputs for further details

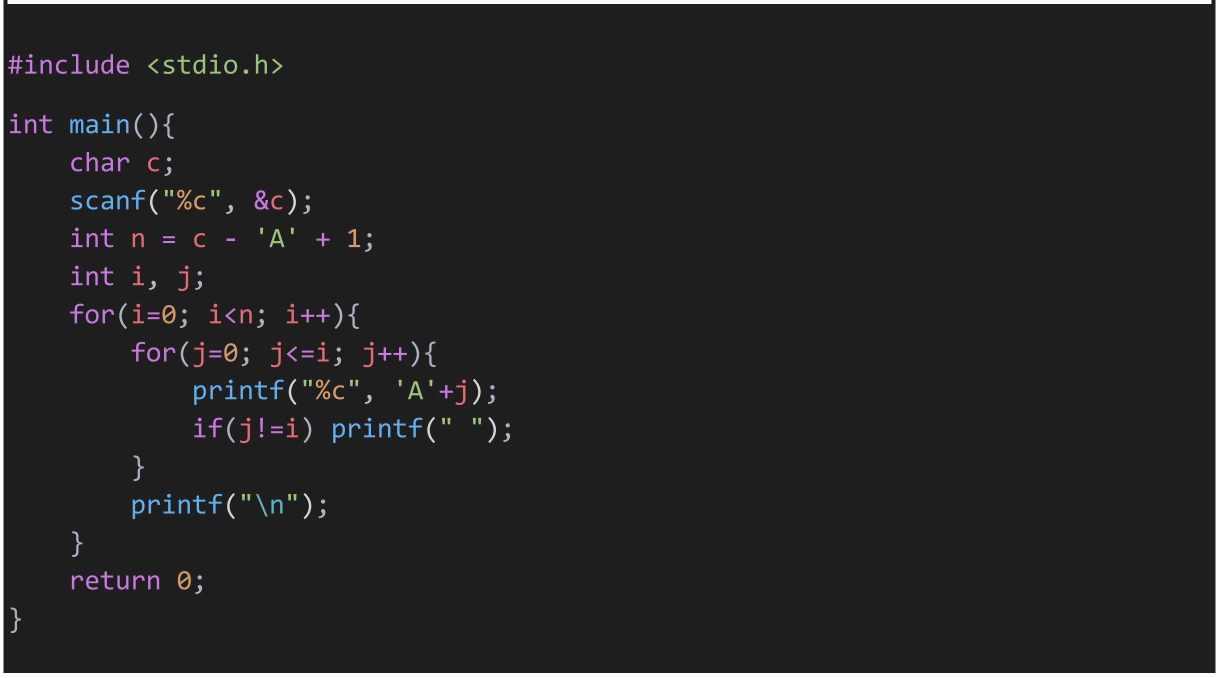
Hint: There is a space between characters in a row but there is no space after the last character or before the first character in a row.

Sample Input 1

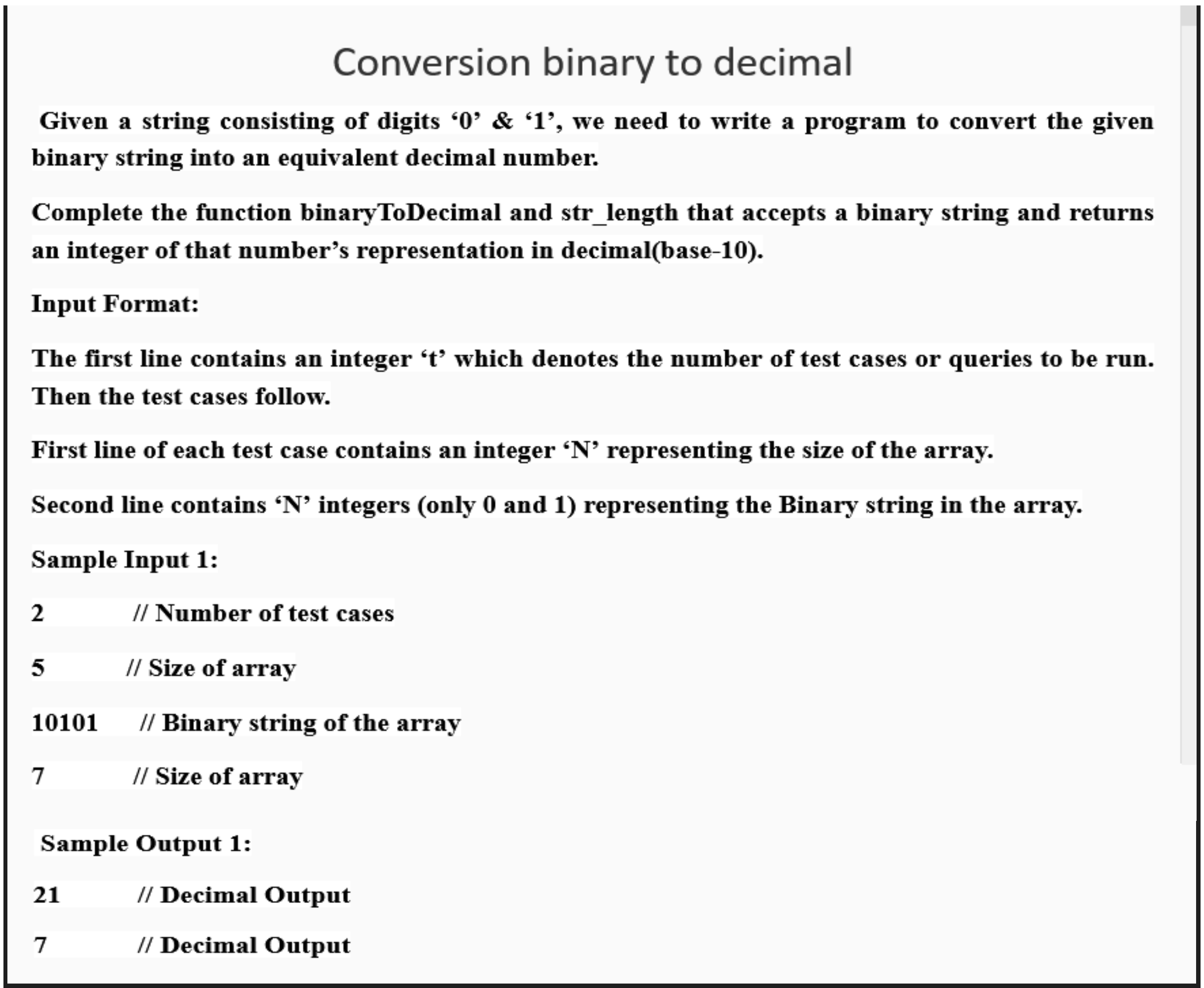
```
F
```

Sample Output 1

```
A

A B

A B C

A B C D

A B C D E

A B C D E F
```



```c
#include <stdio.h>

int main(){
    char c;
    scanf("%c", &c);
    int n = c - 'A' + 1;
    int i, j;
    for(i=0; i<n; i++){
        for(j=0; j<=i; j++){
            printf("%c", 'A'+j);
            if(j!=i) printf(" ");
        }
        printf("\n");
    }
    return 0;
}
```



Conversion binary to decimal

Given a string consisting of digits '0' & '1', we need to write a program to convert the given binary string into an equivalent decimal number.

Complete the function binaryToDecimal and str_length that accepts a binary string and returns an integer of that number's representation in decimal(base-10).

Input Format:

The first line contains an integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case contains an integer 'N' representing the size of the array.

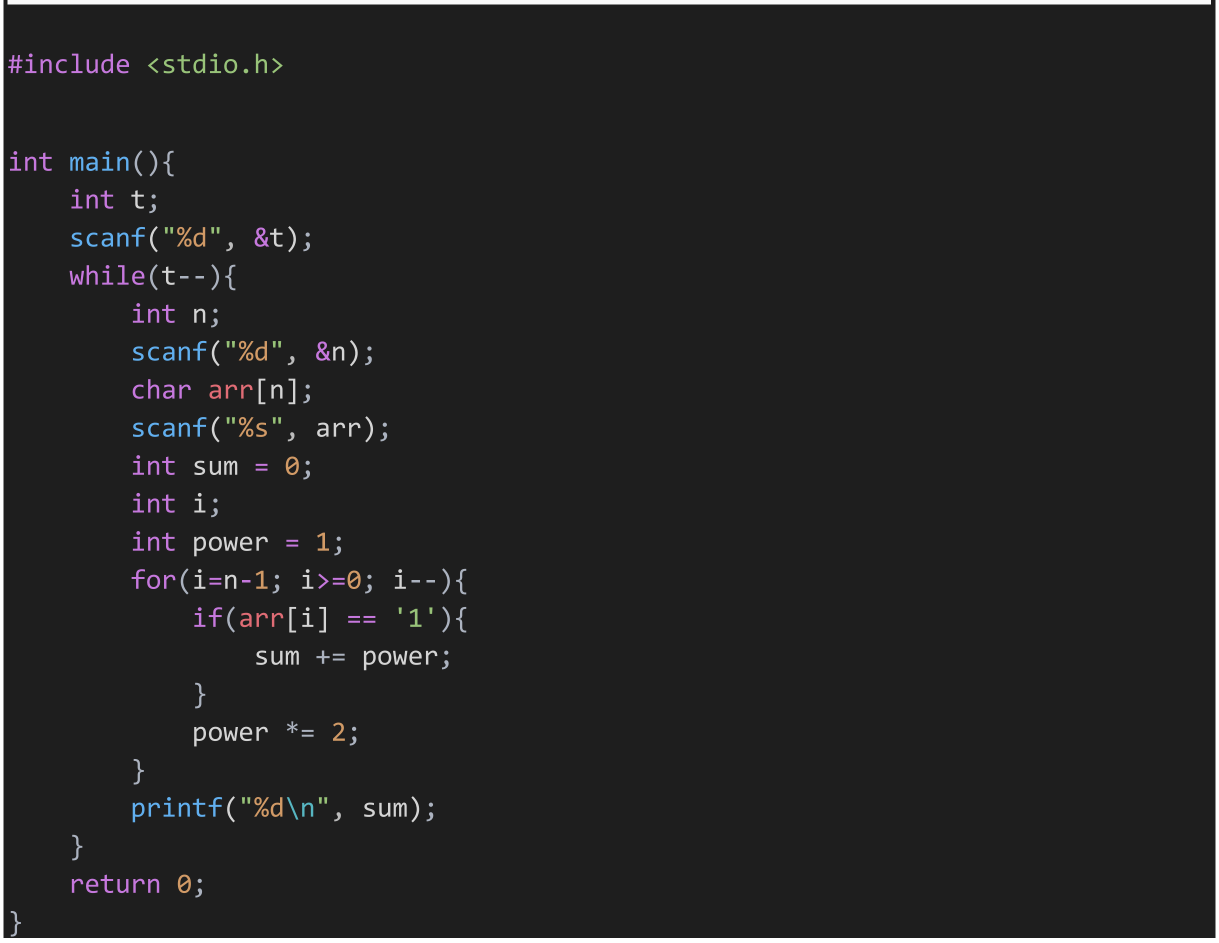
Second line contains 'N' integers (only 0 and 1) representing the Binary string in the array.

Sample Input 1:

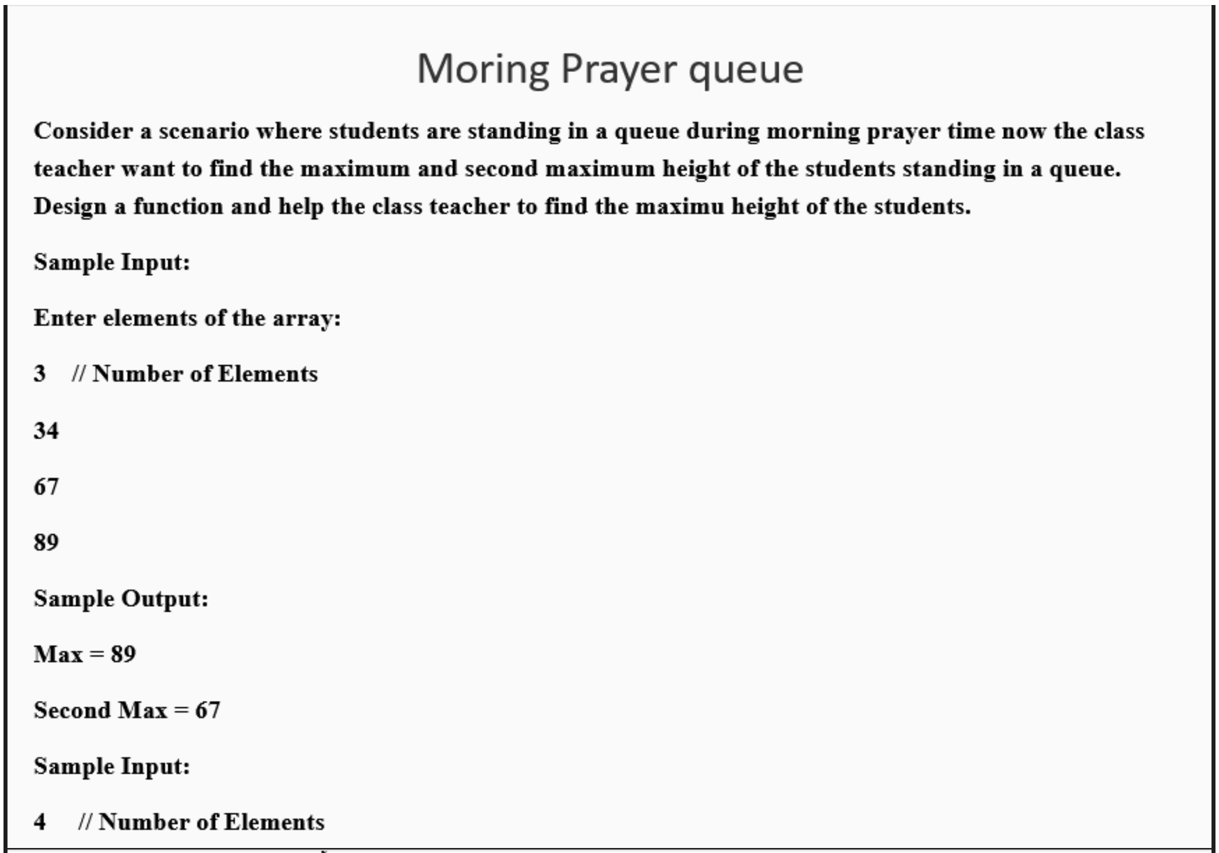
```
2        // Number of test cases
5        // Size of array
10101    // Binary string of the array
7        // Size of array
```

Sample Output 1:

```
21       // Decimal Output
7        // Decimal Output
```



```c
#include <stdio.h>

int main(){
    int t;
    scanf("%d", &t);
    while(t--){
        int n;
        scanf("%d", &n);
        char arr[n];
        scanf("%s", arr);
        int sum = 0;
        int i;
        int power = 1;
        for(i=n-1; i>=0; i--){
            if(arr[i] == '1'){
                sum += power;
            }
            power *= 2;
        }
        printf("%d\n", sum);
    }
    return 0;
}
```



Moring Prayer queue

Consider a scenario where students are standing in a queue during morning prayer time now the class teacher want to find the maximum and second maximum height of the students standing in a queue. Design a function and help the class teacher to find the maximu height of the students.

Sample Input:

Enter elements of the array:

3 // Number of Elements

34

67

89

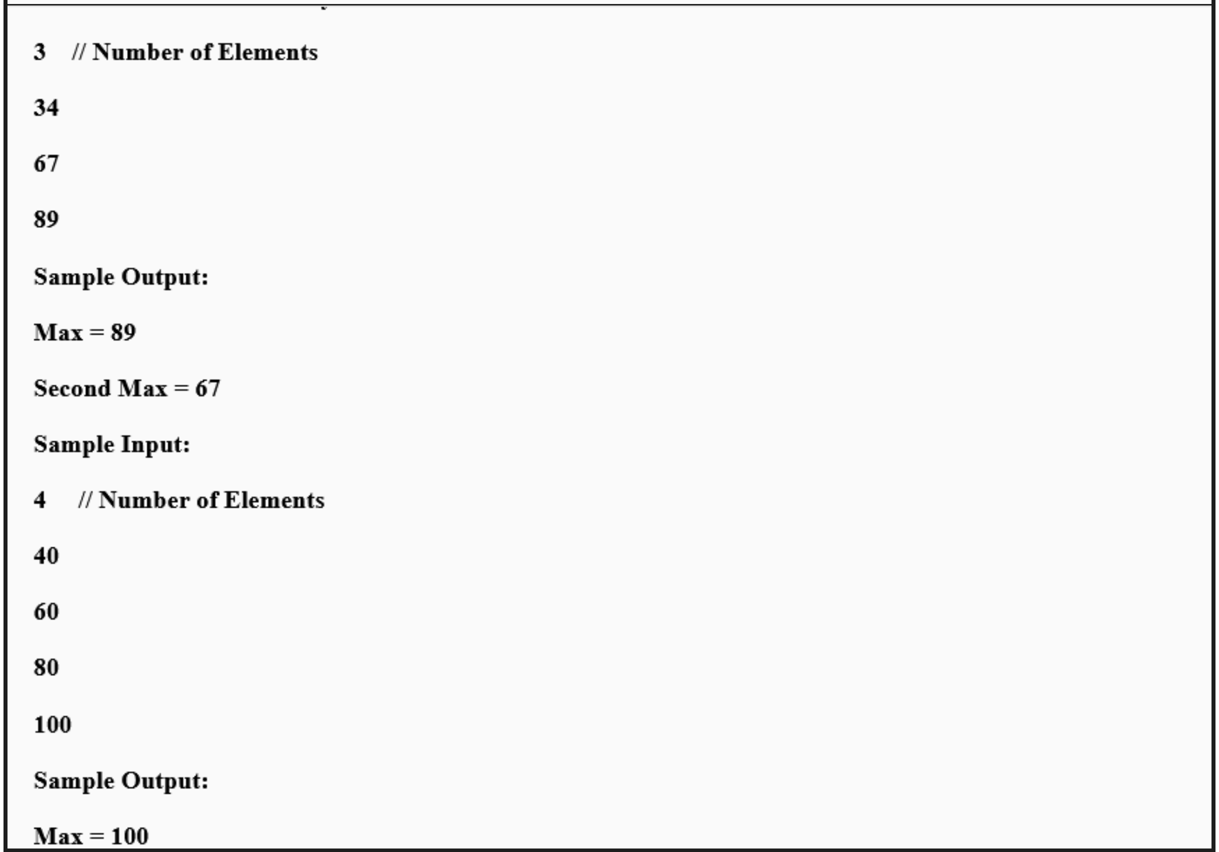
Sample Output:

Max = 89

Second Max = 67

Sample Input:

4 // Number of Elements



3 // Number of Elements

34

67

89

Sample Output:

Max = 89

Second Max = 67

Sample Input:

4 // Number of Elements

40

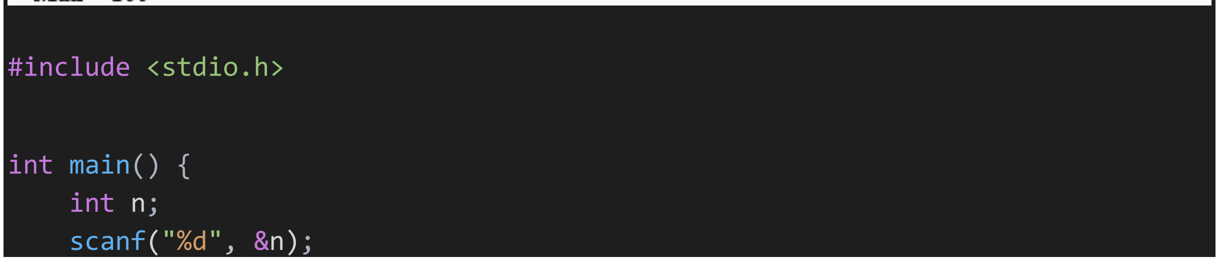
60

80

100

Sample Output:

Max = 100



```c
#include <stdio.h>

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

    int max = arr[0];
    int second_max = arr[0] > arr[1] ? arr[1] : arr[0];

    for (i = 1; i < n; i++) {
        if (arr[i] > max) {
            second_max = max;
            max = arr[i];
        } else if (arr[i] > second_max && arr[i] != max) {
            second_max = arr[i];
        }
    }

    printf("Max = %d\n", max);
    printf("Second Max = %d", second_max);

    return 0;
}
```

An array for max and min

Vaibhav is trying to learn how to return an array from a function.

Given an array of integers and an integer M, help him write a function which returns an array of maximum and minimum values of the integers provided and prints the maximum or minimum value depending on the value of M.

If M is **1 print minimum value** and if M is **2 print the maximum value.**

Sample Input 1

5 1 // Number of elements in an array & value of M

1 5 3 4 2 // Elements of array

Sample Output 1

1 //Printing minimum value

Sample Input 2:

5 2 // Number of elements in an array & value of M

1 3 5 4 2 // Elements of array

Sample Output 2

5 //Printing maximum value

Input Explanation:

The first line contains space separated 'N' and 'M' where N is the total number of integers and M is either 1 or 2 where if M=1 print minimum value and if M=2 print maximum value.

Second line contains the N space separated integers.

For Example:

In Sample Input 2,

First line shows that there are 5 input values and we have to print maximum of those values since M=2.

Next line are 5 space separated input values.

```c
#include <stdio.h>


int solve(int n,int arr[n],int maxMin){
    int i;
    if(maxMin==1){
        int min=arr[0];
        for (i = 0; i < n; i++) {
            if(arr[i]<min) min=arr[i];
        }
        return min;
    }
    else{
        int max=arr[0];
        for (i = 0; i < n; i++) {
            if(arr[i]>max) max=arr[i];
        }
        return max;
    }
}


int main(){
    int n;
    scanf("%d",&n);
    int maxMin;
    scanf("%d",&maxMin);
    int arr[n];
    int i;
    for (i = 0; i < n; i++) {
        scanf("%d",&arr[i]);
    }
    printf("%d",solve(n,arr,maxMin));
}
```

Calculate the total volume of water

John likes to conserve water. On a rainy day, he wants to calculate how many unite of water he can conserve between buildings. Help him to solve his problem.

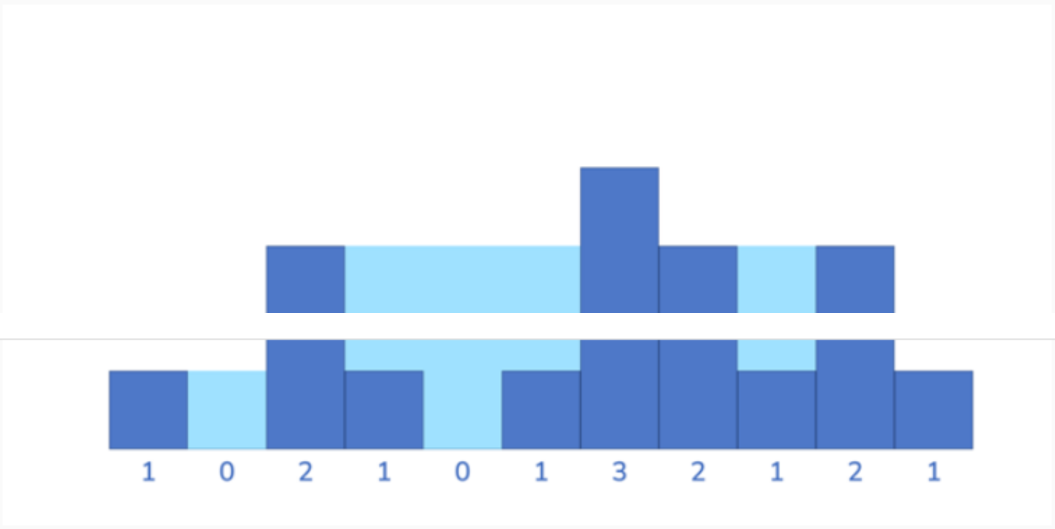
Given an array A where each element denotes the height of blocks, calculate the total volume of water that can be trapped when it rains.

Note: one cubic block has a volume of 1 unit.

Example:

A: [1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1]

The total volume of water is 1 + 3 + 1 + 1 = 6 units.



Input Format:

An integer ‘n’ denoting the size of the array A.

n space separated integers denoting the elements of array A.

Output Format:

The output has a line with an integer ‘w’ denoting the volume of trapped rainwater.

Constraints:

1 <= n <= 30000

0 <= Ai <= 104

Sample Input 1:

5

1 0 0 1 0

Sample output 1:

2

Sample input 2:

11

1 0 2 1 0 1 3 2 1 2 1

Sample output 2:

6

```c
#include <stdio.h>
int max(int a, int b){
    if(a > b) return a;
    else return b;
}
int min(int a, int b){
    if (a > b) return b;
    else return a;
}
int main(){
    int n;
    scanf("%d", &n);
    int arr[n];
    int i;
    for(i = 0; i<n; i++){
        scanf("%d", &arr[i]);
    }

    int rmax[n];
    int lmax[n];
    lmax[0] = arr[0];
    rmax[n-1] = arr[n-1];
    int res = 0;
    for(i=1; i<n; i++){
        lmax[i] = max(lmax[i-1], arr[i]);
    }

    for(i=n-2; i>=0; i--){
        rmax[i] = max(rmax[i+1], arr[i]);
    }

    for(i=1; i<n-1; i++){
        res += min(lmax[i], rmax[i]) - arr[i];
    }

    printf("%d", res);


}
```

Medals calculation

C countries participated in an event that happened in April of 2021 and 2022 in Dubai. There were K categories in which Gold, Silver and Bronze prizes were given to the participants. Given a country and type of medal, **write a C program to get the total number of (required type) medals have been won by the given country (in 2021 and 2022 combined).**

Hint: Matrix Addition

Sample Input 1

```
3 2 1                          // Participated countries=3, Country number =2, Medal type =1
12 1 10 10 4 5 5 12 18    // Medals count for the year 2021
10 6 5 12 4 1 18 1 8        // Medals count for the year 2022
```

Sample Output 1

```
22                             // 22 gold medal won by country number 2
```

Input Explanation: The first line contains three positive integers 'C', 'N' and 'M' where **'C'** is the number of countries that participated in 2021 and 2022 , **'N'** is the country number whose total medals is to be calculated and **'M'** is the medal type that's total is to be calculated. The second line contains '3C' space separated Gold, Silver and Bronze medal counts for each of the 'C' countries for the year 2021. The third line contains '3C' space separated Gold, Silver and Bronze medal counts for each of

Note:

N=1 means country number 1 , N=2 means country number 2 and so on….

M=1 means Gold, M=2 means Silver, M=3 means Bronze

For Example:

In Sample Input 1,

First line shows that 3 countries participated in 2021 and 2022 and we have to calculate the total Gold medals(since M=1) won by country number 2 in 2021 and 2022 combined .

Second line shows medal counts for 2021:

Country 1 won 12 Gold, 1 Silver and 10 Bronze

Country 2 won 10 Gold, 4 Silver and 5 Bronze

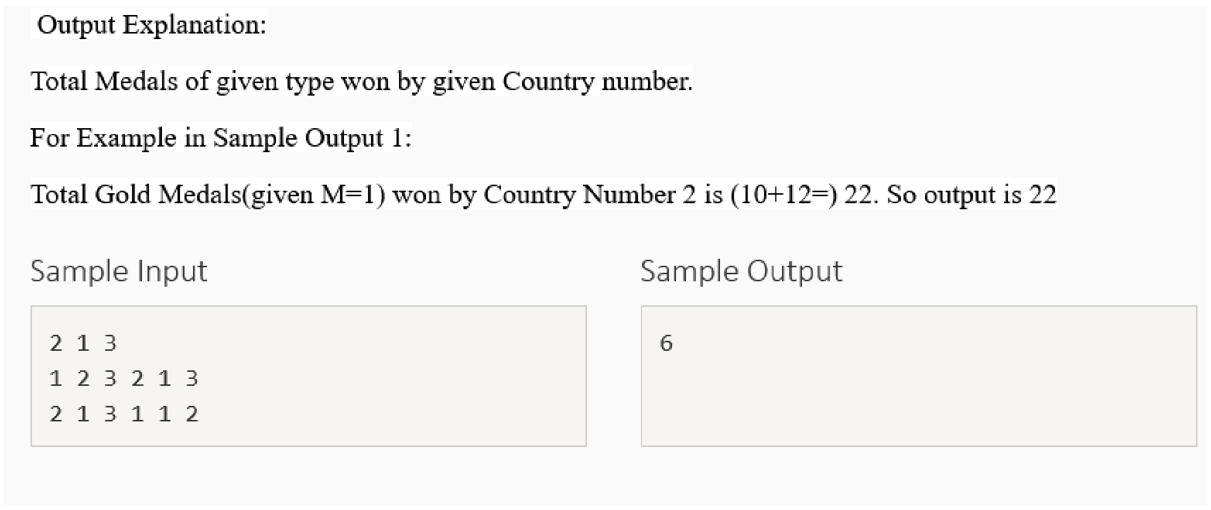
Country 3 won 5 Gold, 12 Silver and 18 Bronze

Third line shows medal counts for 2022:

Country 1 won 10 Gold, 6 Silver and 5 Bronze

Country 2 won 12 Gold, 4 Silver and 1 Bronze

Country 3 won 18 Gold, 1 Silver and 8 Bronze


Output Explanation:

Total Medals of given type won by given Country number.

For Example in Sample Output 1:

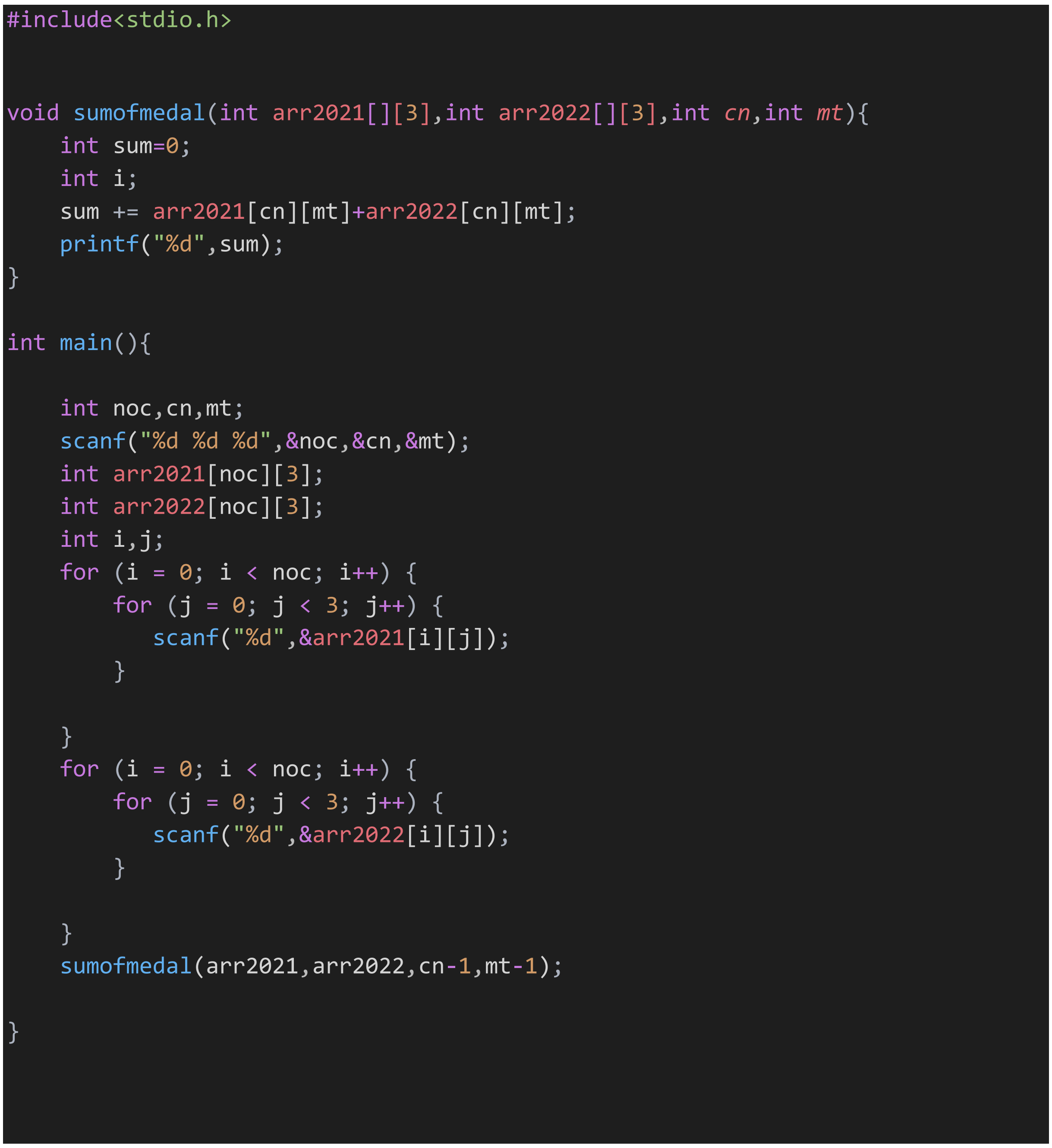
Total Gold Medals(given M=1) won by Country Number 2 is (10+12=) 22. So output is 22

Sample Input

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

Sample Output

```
6
```


```c
#include<stdio.h>


void sumofmedal(int arr2021[][3],int arr2022[][3],int cn,int mt){
    int sum=0;
    int i;
    sum += arr2021[cn][mt]+arr2022[cn][mt];
    printf("%d",sum);
}

int main(){

    int noc,cn,mt;
    scanf("%d %d %d",&noc,&cn,&mt);
    int arr2021[noc][3];
    int arr2022[noc][3];
    int i,j;
    for (i = 0; i < noc; i++) {
        for (j = 0; j < 3; j++) {
           scanf("%d",&arr2021[i][j]);
        }

    }
    for (i = 0; i < noc; i++) {
        for (j = 0; j < 3; j++) {
           scanf("%d",&arr2022[i][j]);
        }

    }
    sumofmedal(arr2021,arr2022,cn-1,mt-1);

}
```

Profit in confectionary

Shekhar opened a confectionary and had 5 items in his shop- Cookies, Cupcakes, Cake, Chocolate, and Muffins. Saturday and Sunday his shop remains closed. He observed that during first week, profit earned on Cookies on Monday is same as profit earned on Cupcakes on Tuesday, profit earned on Cake on Wednesday, profit earned on Chocolate on Thursday and profit earned on Muffins on Friday. Same thing happened with profit earned on Cupcakes on Monday was same as profit earned on Cake on Tuesday, profit earned on Chocolate on Wednesday and profit earned on Chocolate on Thursday. It kind of forms **Toeplitz Matrix** if we put **Weekday as rows** and **Items in the same order as mentioned above as columns**. Being a techie, help Shekhar write a C program to check **if his profits on those items form a Toeplitz matrix or not** if order of items is not changed.

Note: A Toeplitz (or diagonal-constant) matrix is a matrix in which each descending diagonal from left to right is constant, i.e., all elements in a diagonal are same.

Sample Input 1

```
1

1 2 3 4 5

6 1 2 3 4

7 6 1 2 3

8 7 6 1 2

9 8 7 6 1
```

Sample Output 1

```
TRUE
```

Input Explanation:

The first line contains a positive integer ‘t’ where ‘t’ is the number of test cases.

For each test case 5 lines follow.

First line of which are the profits of Cookies, Cupcakes, Cake, Chocolate, and Muffins for Monday

Second line are the profits of Cookies, Cupcakes, Cake, Chocolate, and Muffins for Tuesday

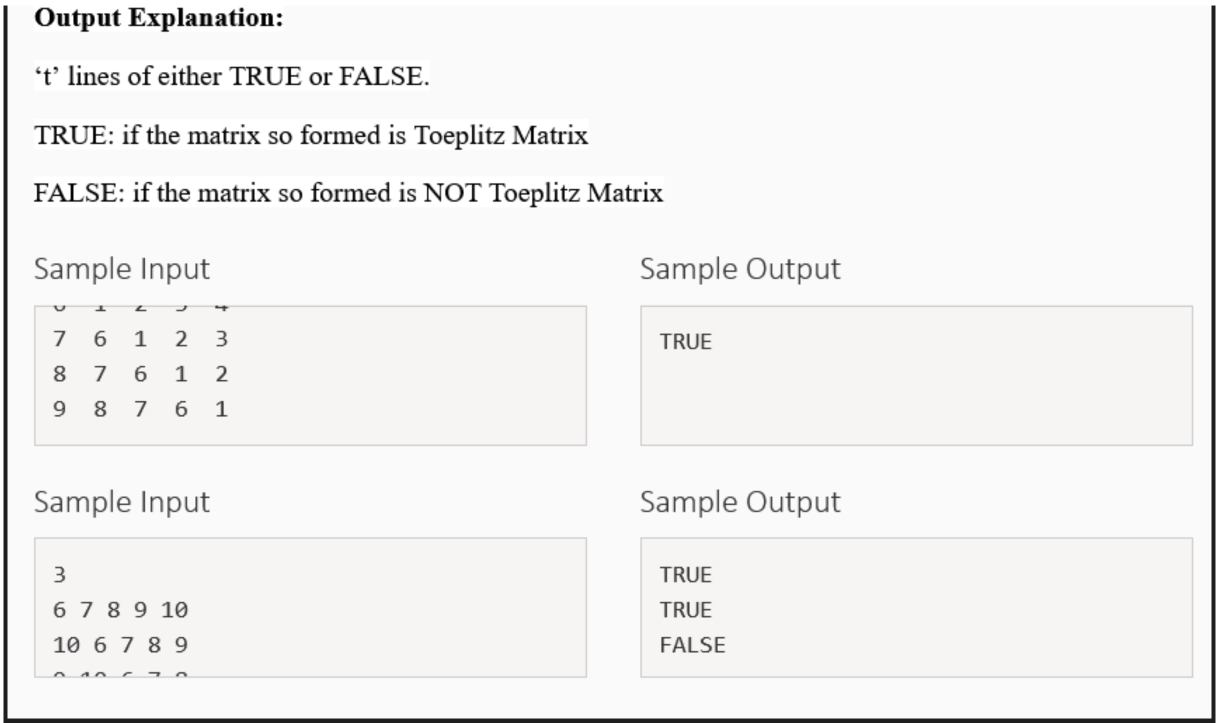
Third line are the profits of Cookies, Cupcakes, Cake, Chocolate, and Muffins for Wednesday

Fourth line are the profits of Cookies, Cupcakes, Cake, Chocolate, and Muffins for Thursday

Fifth line are the profits of Cookies, Cupcakes, Cake, Chocolate, and Muffins for Friday

For Example:

Output Explanation:



Output Explanation:

‘t’ lines of either TRUE or FALSE.

TRUE: if the matrix so formed is Toeplitz Matrix

FALSE: if the matrix so formed is NOT Toeplitz Matrix

Sample Input

```
7   6   1   2   3
8   7   6   1   2
9   8   7   6   1
```

Sample Output

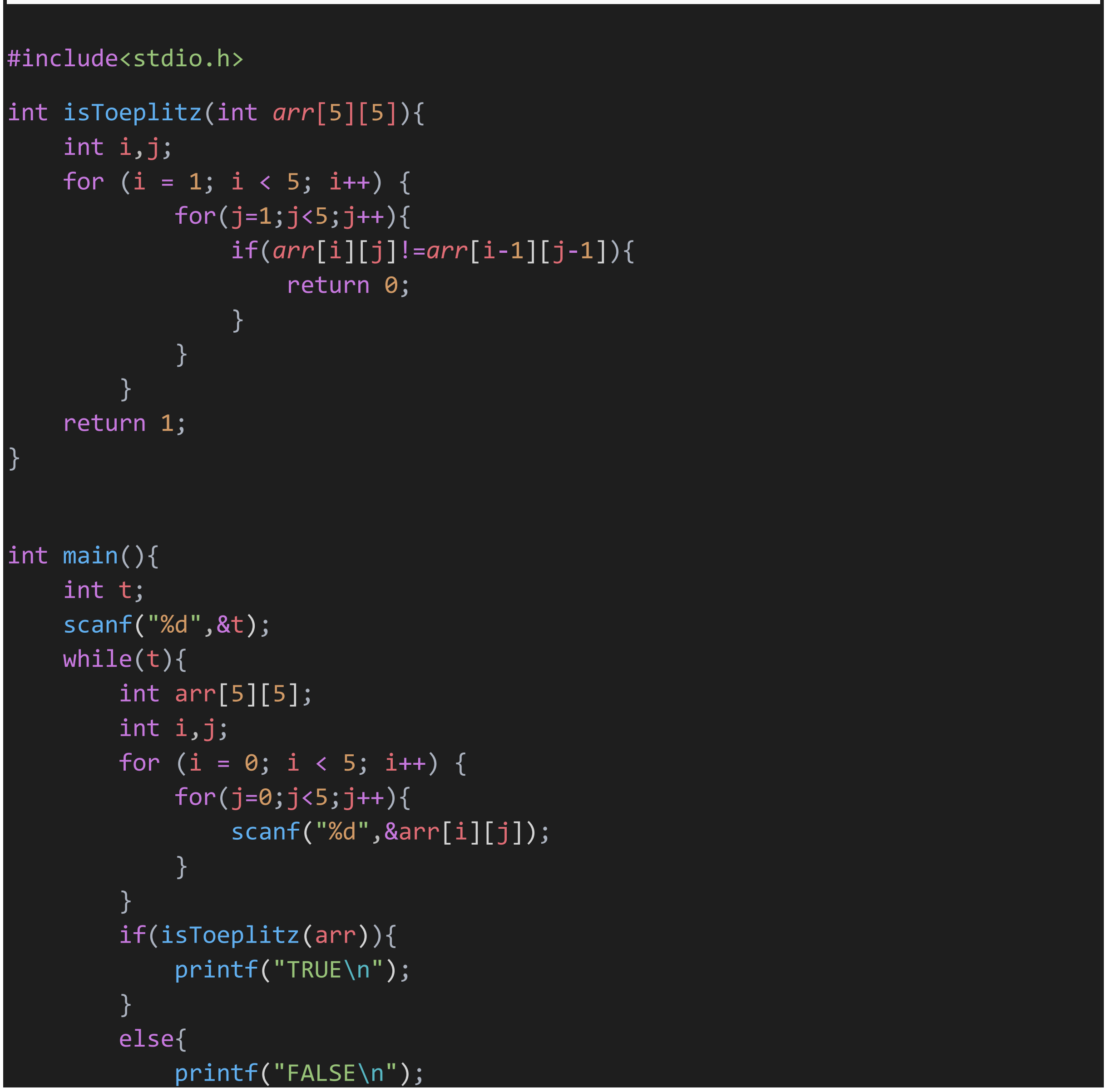
```
TRUE
```

Sample Input

```
3
6 7 8 9 10
10 6 7 8 9
```

Sample Output

```
TRUE
TRUE
FALSE
```



```c
#include<stdio.h>

int isToeplitz(int arr[5][5]){
    int i,j;
    for (i = 1; i < 5; i++) {
            for(j=1;j<5;j++){
                if(arr[i][j]!=arr[i-1][j-1]){
                    return 0;
                }
            }
        }
    return 1;
}


int main(){
    int t;
    scanf("%d",&t);
    while(t){
        int arr[5][5];
        int i,j;
        for (i = 0; i < 5; i++) {
            for(j=0;j<5;j++){
                scanf("%d",&arr[i][j]);
            }
        }
        if(isToeplitz(arr)){
            printf("TRUE\n");
        }
        else{
            printf("FALSE\n");
        }

        t--;
    }


}
```

Sort an array

Write a function in C to sort an array of strings using a doubly pointer.

Sample input:

4 //Number of strings

pr

fr

gy

hu

Sample output:

The sorted array of strings is:

fr

gy

hu

pr

Sample input:

2

python

c

Sample Output:

The sorted array of strings is:

c

python

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

void swap(char **str1, char **str2) {
    char *temp = *str1;
    *str1 = *str2;
    *str2 = temp;
}

void sortStrings(char **arr, int n) {
    int i, j;
    for (i = 0; i < n-1; i++) {
        for (j = 0; j < n-i-1; j++) {
            if (strcmp(arr[j], arr[j+1]) > 0) {
                swap(&arr[j], &arr[j+1]);
            }
        }
    }
}

int main() {
    int n;
    scanf("%d", &n);
    char *arr[n];
    int i;
    for (i = 0; i < n; i++) {
        arr[i] = malloc(sizeof(char)*100);
        scanf("%s", arr[i]);
    }

    sortStrings(arr, n);

    printf("The sorted array of strings is:\n");
    for (i = 0; i < n; i++) {
        printf("%s\n", arr[i]);
    }

    return 0;
}
```

Return that duplicate number

Given an integer array of size N which contains numbers from 0 to N. Each number is present at least once. That is, if N = 5, the array constitutes values ranging from 0 to 5 and among these, there is a single integer value that is present twice. **You need to find and return that duplicate number present in the array.**

Note:

Duplicate number is always present in the given array.

Input format:

The first line contains an Integer 't' which denotes the number of test cases or queries to be run. Then the test cases follow.

First line of each test case or query contains an integer 'N' representing the size of the array/list.

Second line contains 'N' single space separated integers representing the elements in the array/list.

Output Format:

For each test case, print the duplicate element in the array/list.

Output for every test case will be printed in a separate line.

Sample Input 1:

```
2   //Number of test case

5    //Number of elements in the first array

0 2 1 3 1

7     //Number of elements in second array

0 3 1 5 4 3 2
```

Sample Output 1:

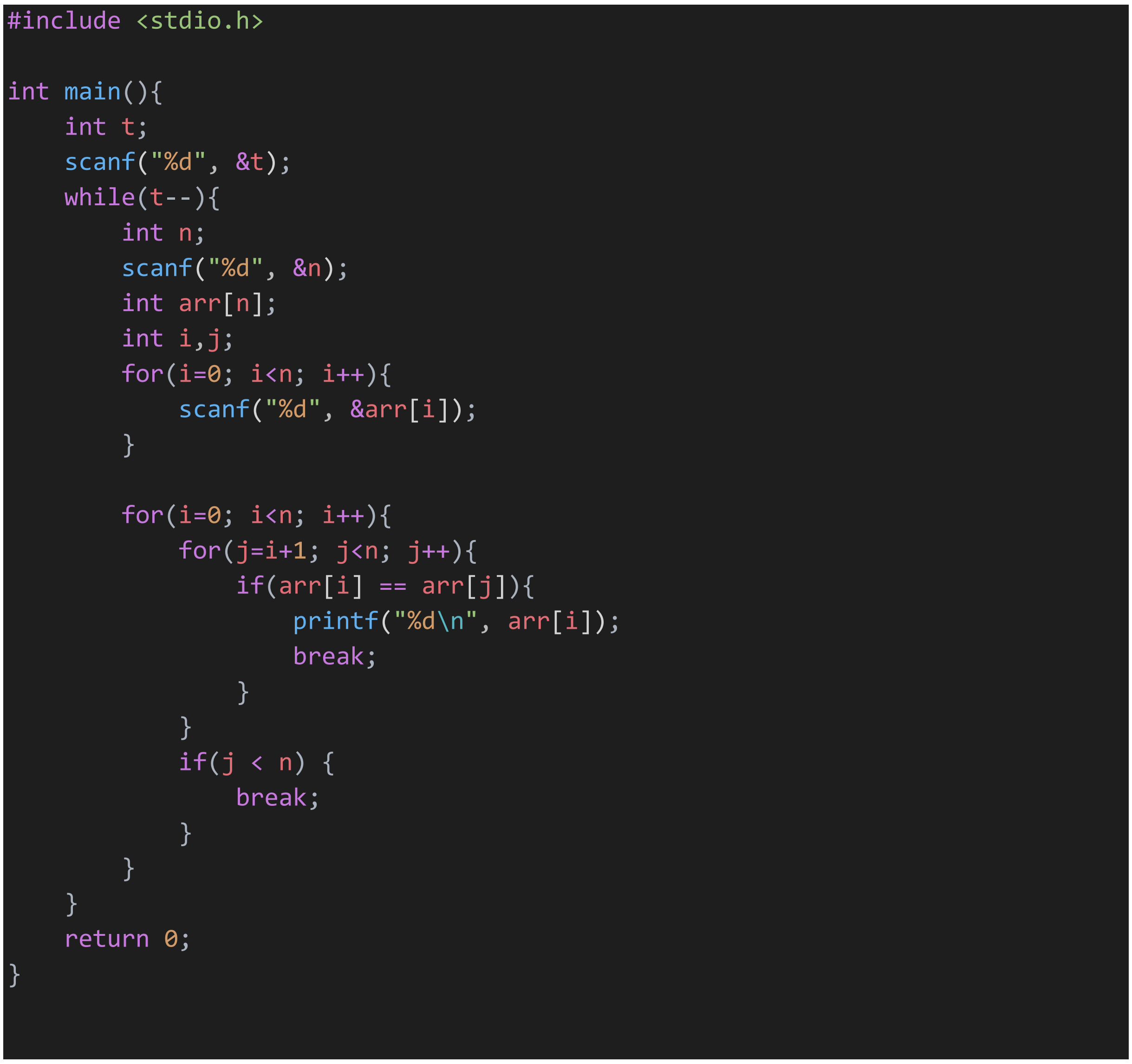
```
1

3
```

Sample Input 2:

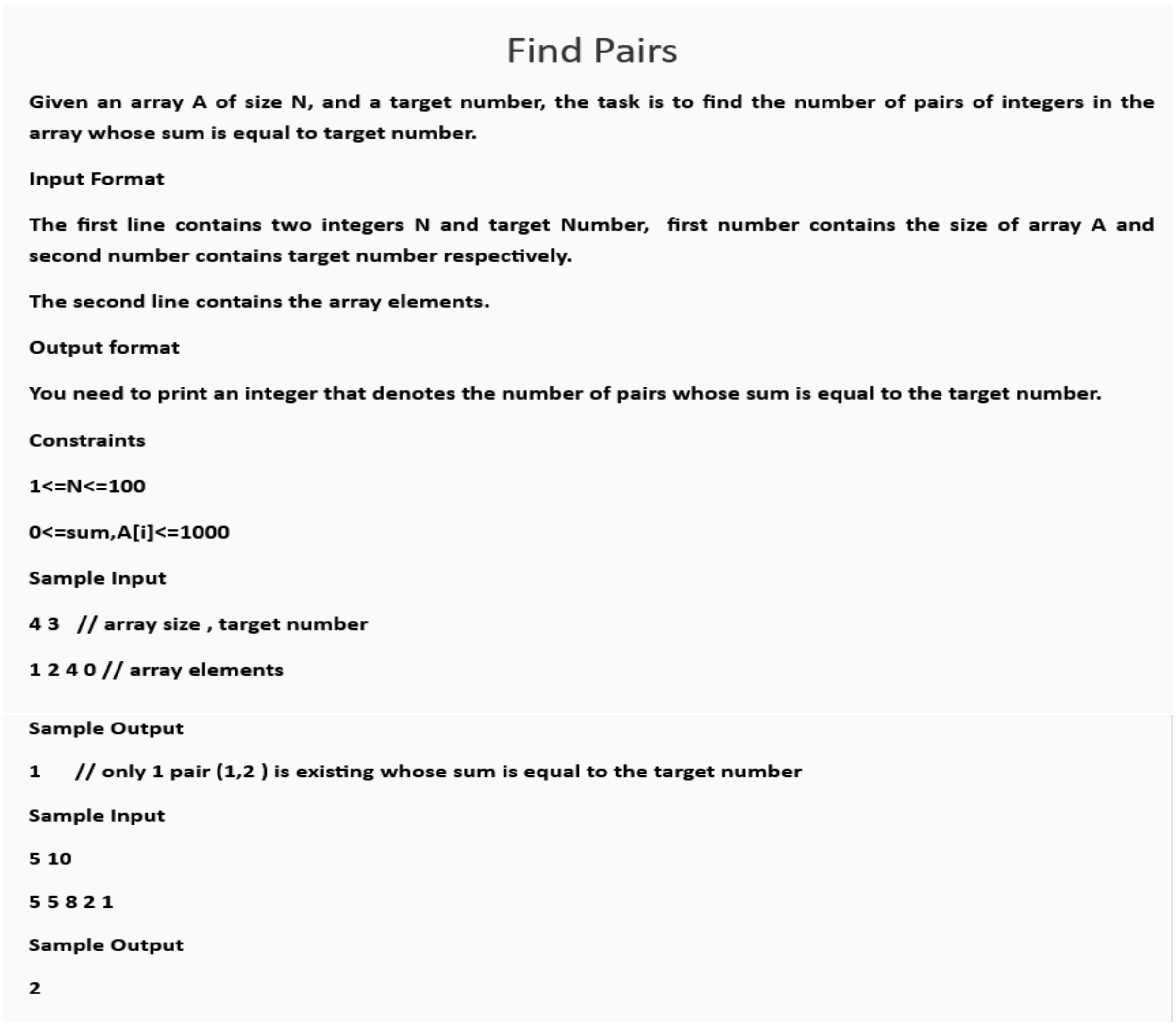
```
2

5

1 1 2 3 6

4

2 2 3 5
```

Sample Output 2:

```
1

2
```



```c
#include <stdio.h>

int main(){
    int t;
    scanf("%d", &t);
    while(t--){
        int n;
        scanf("%d", &n);
        int arr[n];
        int i,j;
        for(i=0; i<n; i++){
            scanf("%d", &arr[i]);
        }

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



Find Pairs

Given an array A of size N, and a target number, the task is to find the number of pairs of integers in the array whose sum is equal to target number.

Input Format

The first line contains two integers N and target Number, first number contains the size of array A and second number contains target number respectively.

The second line contains the array elements.

Output format

You need to print an integer that denotes the number of pairs whose sum is equal to the target number.

Constraints

1<=N<=100

0<=sum,A[i]<=1000

Sample Input

4 3   // array size , target number

1 2 4 0 // array elements

Sample Output

1   // only 1 pair (1,2) is existing whose sum is equal to the target number

Sample Input

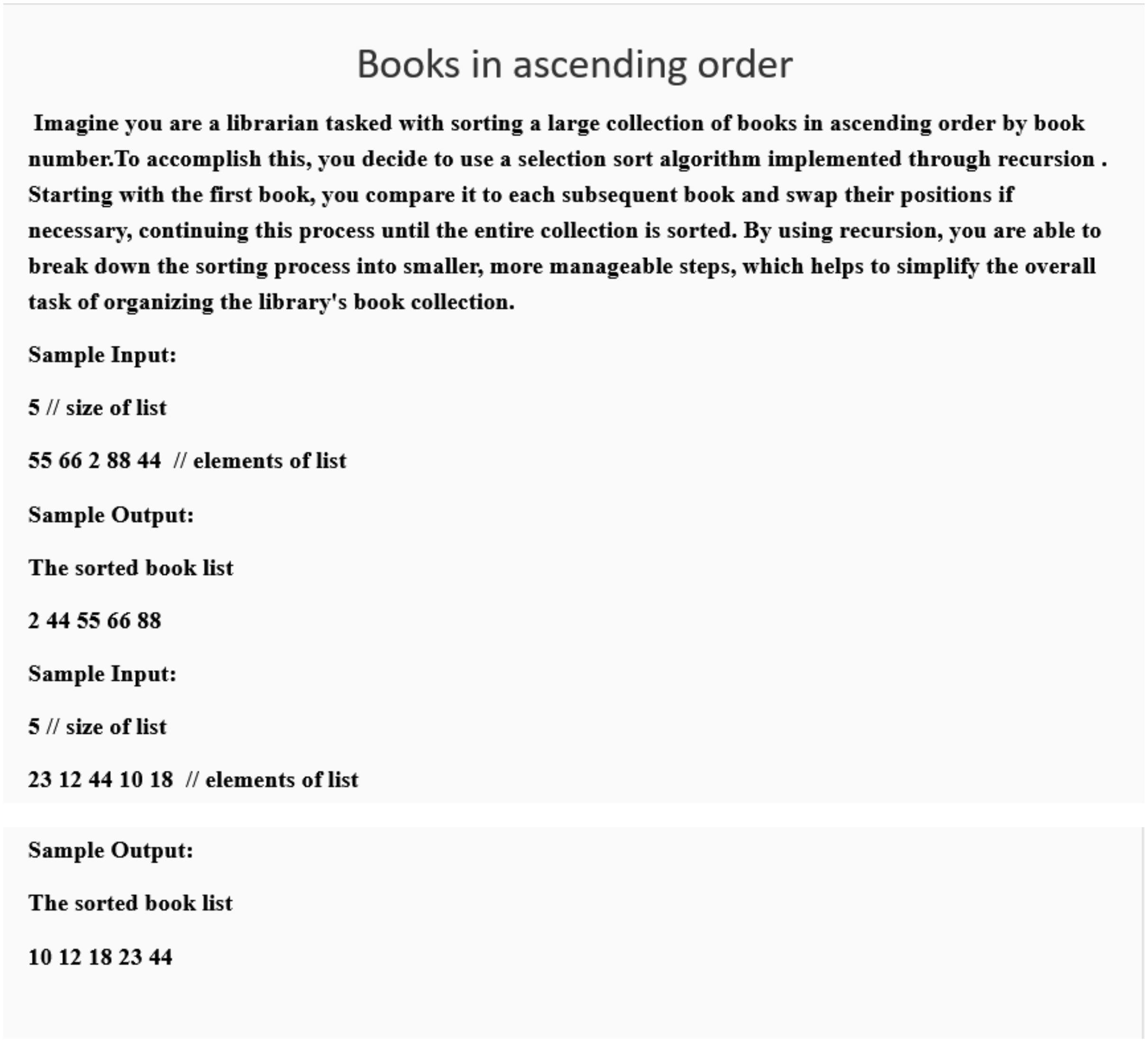
5 10

5 5 8 2 1

Sample Output

2

```c
#include<stdio.h>
int main(){
    int n, m;
    scanf("%d", &n);
    scanf("%d", &m);

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

    int count = 0;

    for(i=0; i<n-1; i++){
        for(j=i+1; j<n; j++){
            if(arr[i] + arr[j] == m) count++;
        }
    }

    printf("%d", count);
}
```



Books in ascending order

Imagine you are a librarian tasked with sorting a large collection of books in ascending order by book number.To accomplish this, you decide to use a selection sort algorithm implemented through recursion . Starting with the first book, you compare it to each subsequent book and swap their positions if necessary, continuing this process until the entire collection is sorted. By using recursion, you are able to break down the sorting process into smaller, more manageable steps, which helps to simplify the overall task of organizing the library's book collection.

Sample Input:

5 // size of list

55 66 2 88 44 // elements of list

Sample Output:

The sorted book list

2 44 55 66 88

Sample Input:

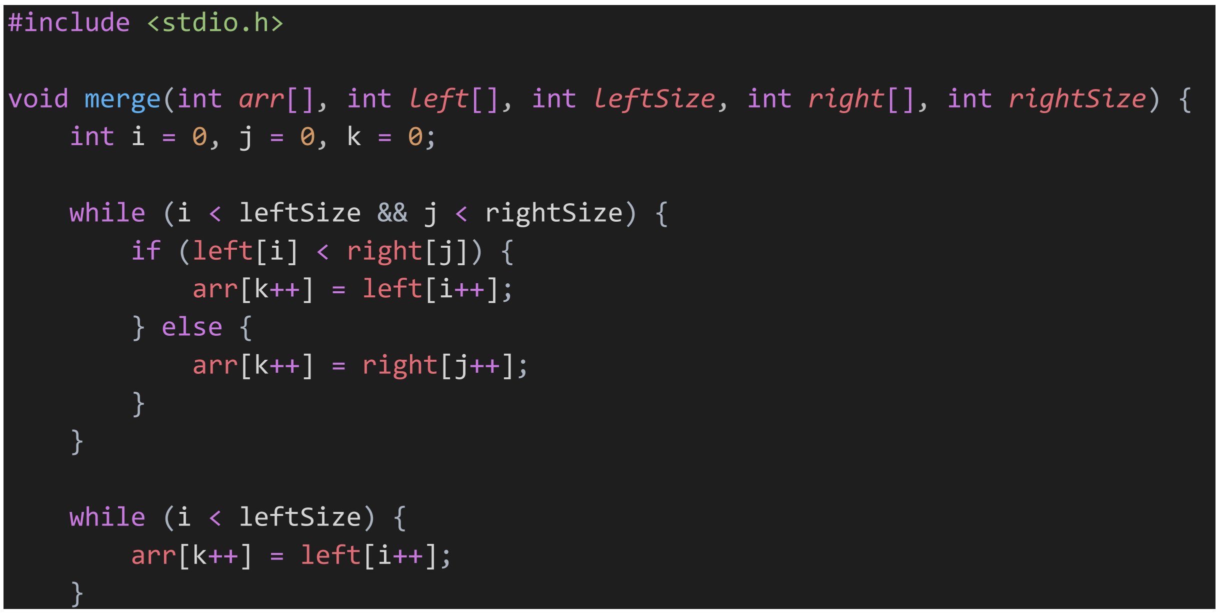
5 // size of list

23 12 44 10 18 // elements of list

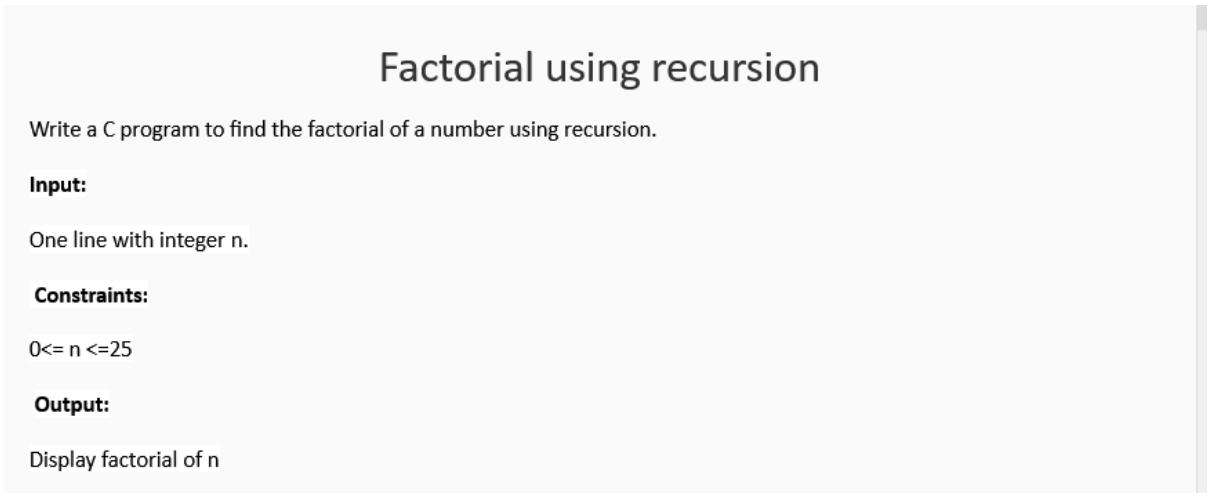
Sample Output:

The sorted book list

10 12 18 23 44



```c
#include <stdio.h>

void merge(int arr[], int left[], int leftSize, int right[], int rightSize) {
    int i = 0, j = 0, k = 0;

    while (i < leftSize && j < rightSize) {
        if (left[i] < right[j]) {
            arr[k++] = left[i++];
        } else {
            arr[k++] = right[j++];
        }
    }

    while (i < leftSize) {
        arr[k++] = left[i++];
    }
    while (j < rightSize) {
        arr[k++] = right[j++];
    }
}

void mergeSort(int arr[], int size) {
    if (size <= 1) {
        return;
    }

    int mid = size / 2;

    int left[mid];
    int i;
    for (i = 0; i < mid; i++) {
        left[i] = arr[i];
    }

    int right[size - mid];
    for (i = mid; i < size; i++) {
        right[i - mid] = arr[i];
    }

    mergeSort(left, mid);
    mergeSort(right, size - mid);

    merge(arr, left, mid, right, size - mid);
}

int main() {
    int size;
    scanf("%d", &size);
    int i;
    int arr[size];
    for(i=0; i<size; i++){
        scanf("%d", &arr[i]);
    }
    mergeSort(arr, size);
    printf("The sorted book list \n");
    for (i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }

    return 0;
}
```



Factorial using recursion

Write a C program to find the factorial of a number using recursion.

Input:

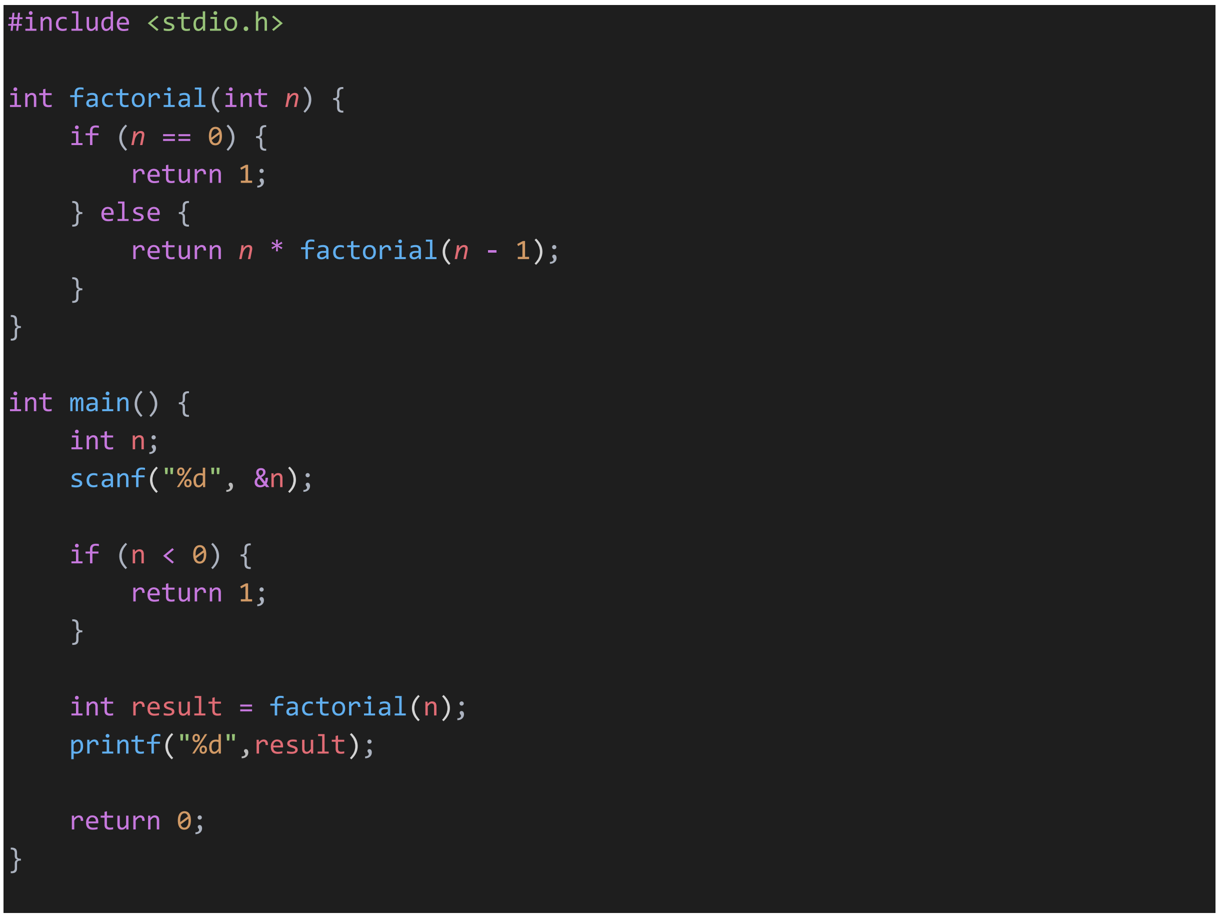
One line with integer n.

Constraints:

0<= n <=25

Output:

Display factorial of n



```c
#include <stdio.h>

int factorial(int n) {
    if (n == 0) {
        return 1;
    } else {
        return n * factorial(n - 1);
    }
}

int main() {
    int n;
    scanf("%d", &n);

    if (n < 0) {
        return 1;
    }

    int result = factorial(n);
    printf("%d",result);

    return 0;
}
```

Find the Missing Number

Problem Statement

You are given a list of N-1 elements having values from 1 to N with one element missing. Your task is to find out the missing number. The given list of N-1 elements may not be a sorted list.

Note: The sum of numbers from 1 to N is calculated as (n*(n+1))/2

Input Format

The first line contains N.

Second-line contains N-1 space-separated integers

Output Format

Print the missing number.

```c
#include <stdio.h>

int main() {
    int n, i, sum = 0, expected_sum;
    scanf("%d", &n);

    int arr[n];
    for (i = 0; i < n-1; i++) {
        scanf("%d", &arr[i]);
        sum += arr[i];
    }

    expected_sum = (n * (n+1)) / 2;

    printf("%d\n", expected_sum - sum);

    return 0;
}
```

Standard University

Standard University conducts an exam. For this, the question papers for a class of 7 students are prepared. On every sheet, the roll number of student who has to attempt the paper is written. First student gets all the sheets and searches for the sheet containing his roll number. He takes the sheet and passes all other sheets to the next student. Next student repeats the process and it continues till the time last student gets the sheet.

The students are sitting as per the roll number from 1 to 7. The sheets are not ordered and the sequence is given as input. When one student gets the sheet, all other sheets are shifted. Hence the index of each sheet changes. You have to print the indexes from which each student will get his sheet.

Write a recursive method sheet(int sheets[7], int curroll, int max) for this.

Main Function is written for this.

```
int main()

{

int i;

int a[7];

for(i = 0; i<7; i++)

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

sheet(a, 1, 7);

}
```

Sample Input 1

7 6 5 4 3 2 1 //   sequence of the sheets given to the students

Sample Output 1

6 5 4 3 2 1 0 // index of sheets according to roll.no

Explanation:

Student 1 gets sheet from index 6 (no shifting)

Student 2 gets sheet from index 5 (no shifting)

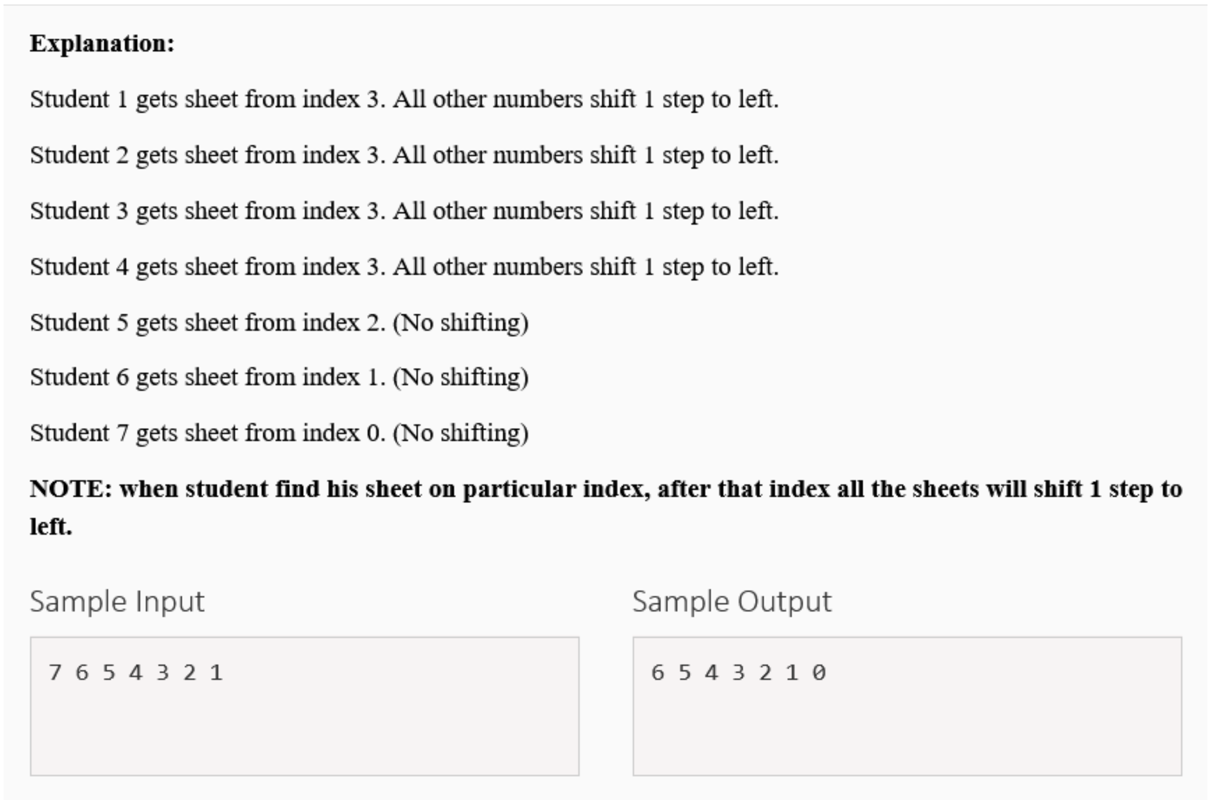
Sample Input 2

7 6 5 1 2 3 4

Sample Output 2

3 3 3 3 2 1 0

Explanation:



Explanation:

Student 1 gets sheet from index 3. All other numbers shift 1 step to left.

Student 2 gets sheet from index 3. All other numbers shift 1 step to left.

Student 3 gets sheet from index 3. All other numbers shift 1 step to left.

Student 4 gets sheet from index 3. All other numbers shift 1 step to left.

Student 5 gets sheet from index 2. (No shifting)

Student 6 gets sheet from index 1. (No shifting)

Student 7 gets sheet from index 0. (No shifting)

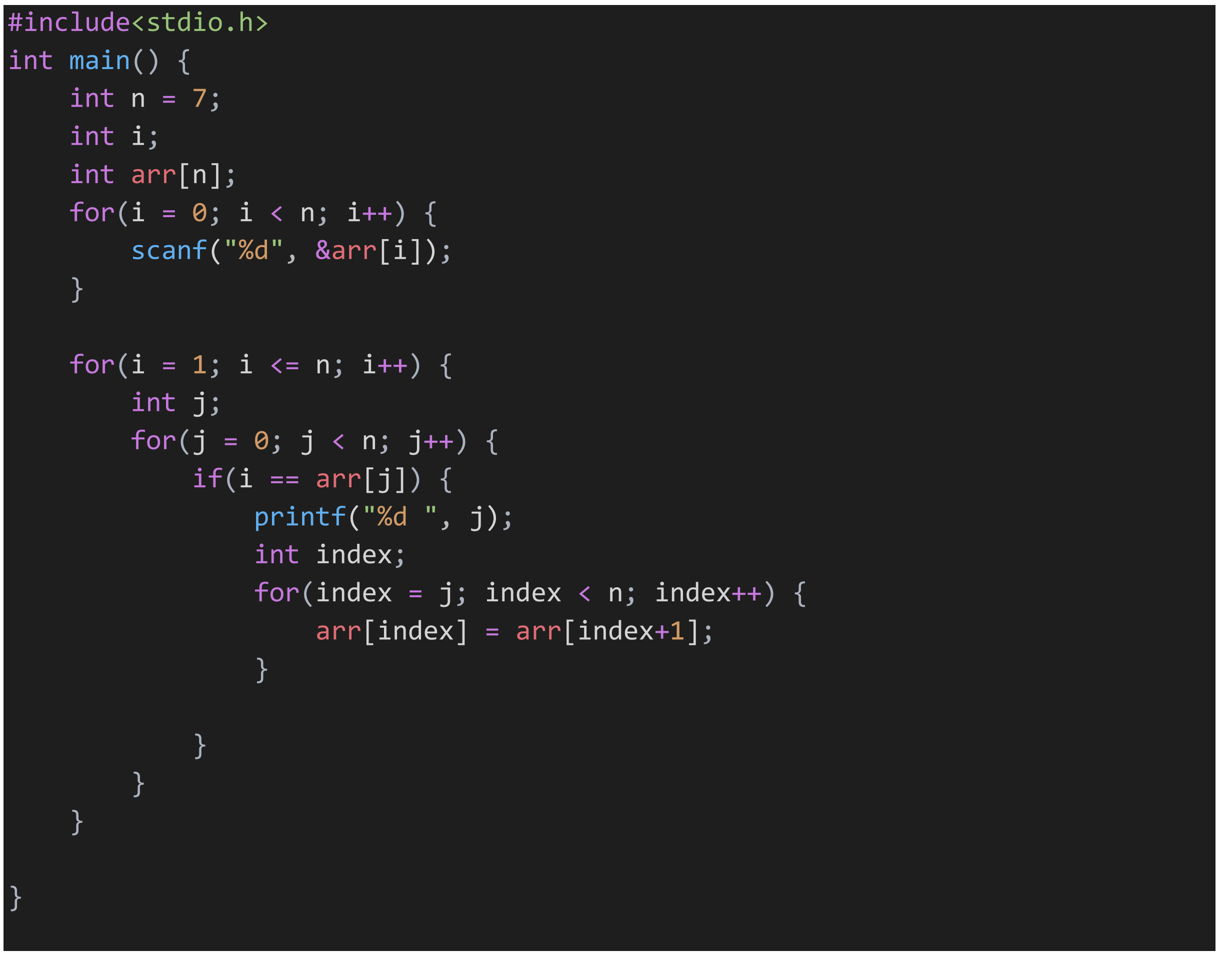
NOTE: when student find his sheet on particular index, after that index all the sheets will shift 1 step to left.

Sample Input

```
7 6 5 4 3 2 1
```

Sample Output

```
6 5 4 3 2 1 0
```



```c
#include<stdio.h>
int main() {
    int n = 7;
    int i;
    int arr[n];
    for(i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    for(i = 1; i <= n; i++) {
        int j;
        for(j = 0; j < n; j++) {
            if(i == arr[j]) {
                printf("%d ", j);
                int index;
                for(index = j; index < n; index++) {
                    arr[index] = arr[index+1];
                }

            }
        }
    }

}
```

Class of each IP address

I work as a network administrator at a large corporation, and one of my responsibilities is to monitor the network traffic and identify any potential security threats. To do this, I need to know the class of each IP address that is connected to the network. Using a program to find the class of an IP address, I am able to quickly determine whether a particular address is a part of a private or public network, which helps me to take appropriate action in case of any security breach.

Hint: In an IP address 192.168.50.60 you have four parts separated by a dot, so in this case 192 is first part, 168 is second part, 50 is third part and 60 is fourth part, so you need to extract first part from IP address and accordingly you need to print the class of it according to the range given.

Note: 0-127 (Class A)

128-191 (Class B)

192-224 (Class C)

225-239 (Class D)

Above 240 (Class E)

Sample Input:

192.168.1.1   // IP Address (xxx.xxx.xxx.xxx format):

Sample Output:

Class C IP Address

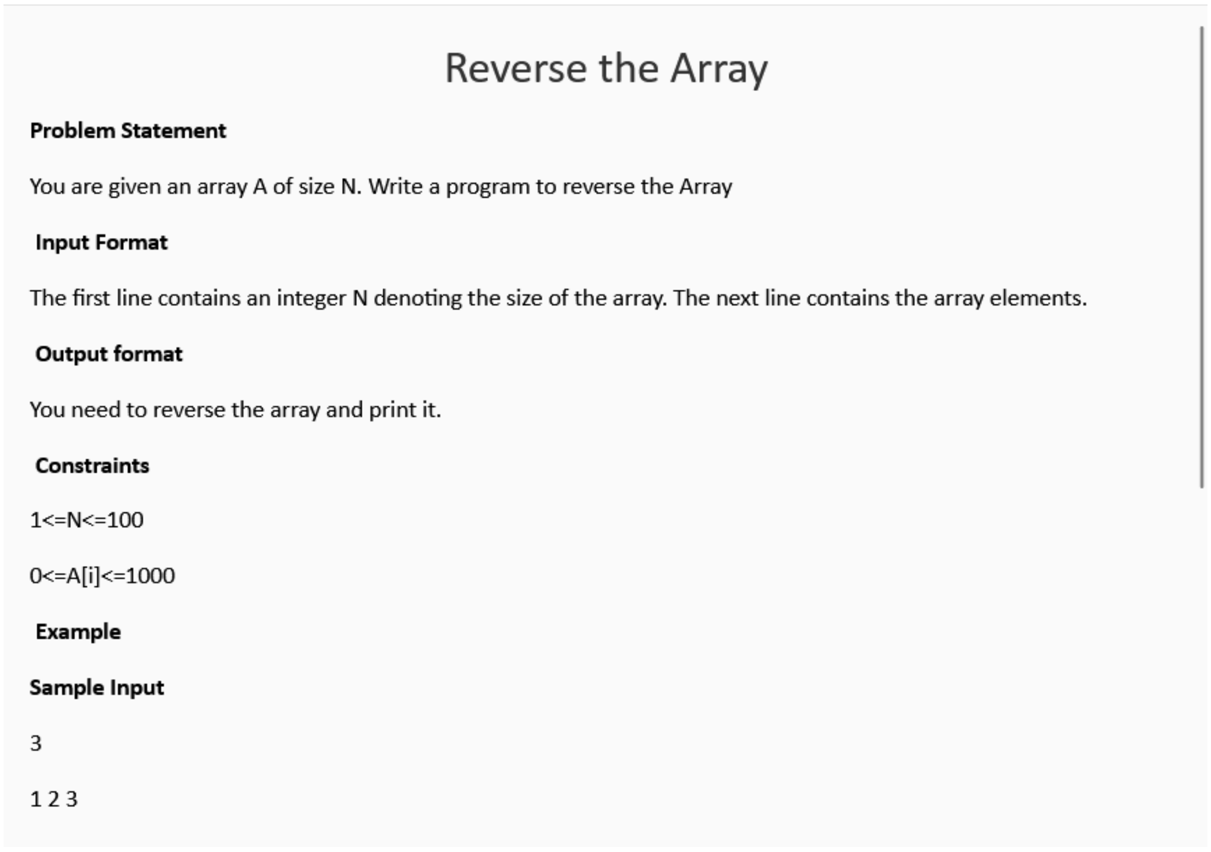
Sample Input:

121.32.65.89 // IP Address (xxx.xxx.xxx.xxx format):

Sample Output:

Class A IP Address

```c
#include <stdio.h>
int main(){
    char arr[50];
    scanf("%s", arr);

    int i = 0;
    int num = 0;
    while(arr[i]!= '.'){
        int r = arr[i] - '0';
        num = num * 10 + r;
        i++;
    }

    // printf("%d", num);
    if(num >= 0 && num <= 127){
        printf("Class A IP Address");
    }

    else if(num >=128 && num <= 191){
        printf("Class B IP Address");
    }
    else if(num >=192 && num <= 224){
        printf("Class C IP Address");
    }
    else if(num >=225 && num <= 240){
        printf("Class D IP Address");
    }
    else{
        printf("Class E IP Address");
    }
}
```



Reverse the Array

Problem Statement

You are given an array A of size N. Write a program to reverse the Array

Input Format

The first line contains an integer N denoting the size of the array. The next line contains the array elements.

Output format

You need to reverse the array and print it.

Constraints

1<=N<=100

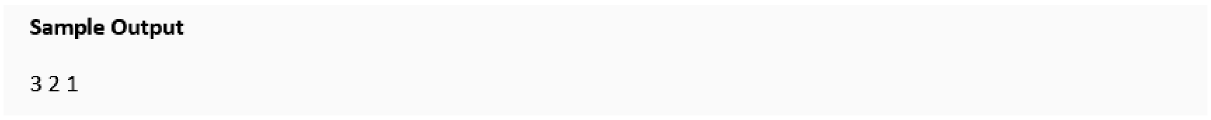
0<=A[i]<=1000

Example

Sample Input

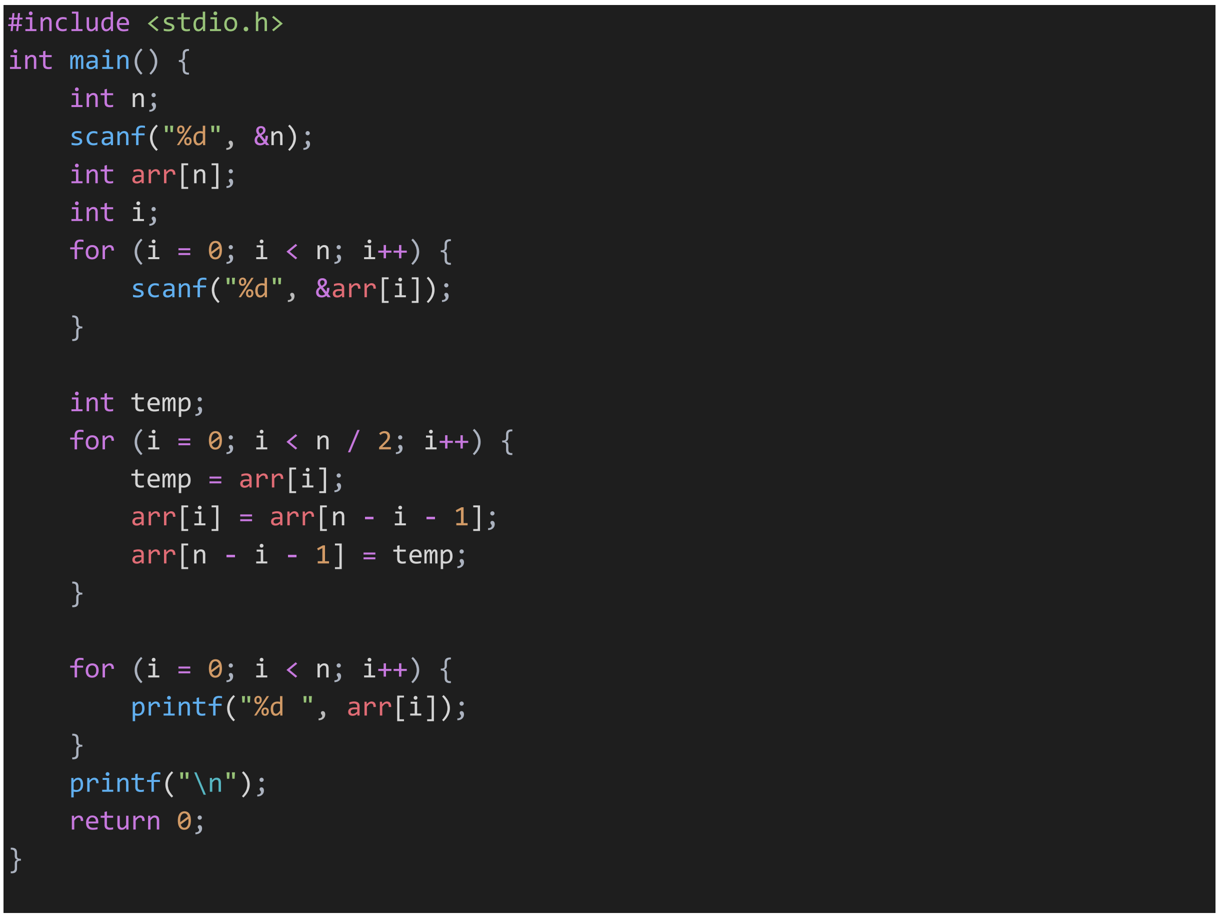
3

1 2 3



Sample Output

3 2 1



```c
#include <stdio.h>
int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    int i;
    for (i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    int temp;
    for (i = 0; i < n / 2; i++) {
        temp = arr[i];
        arr[i] = arr[n - i - 1];
        arr[n - i - 1] = temp;
    }

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

Fibonacci number using recursion

You are a programmer who has been tasked with writing a C program to find the nth Fibonacci number using recursion. How would you approach this task?

Input:

an integer n

Constraints:

0<= n <= 30

Output:

Print nth term of Fibonacci sequence (0,1,1,2,3,5........). Indexing of Fibonacci terms starts from Zero, i.e., the first element will be said at position no. Zero(0) in the series.

```c
#include <stdio.h>

int fibonacci(int n) {
    if (n == 0) {
        return 0;
    } else if (n == 1) {
        return 1;
    } else {
        return fibonacci(n - 1) + fibonacci(n - 2);
    }
}

int main() {
    int n;
    scanf("%d", &n);

    if (n < 0) {
        return 1;
    }

    int result = fibonacci(n);
    printf("%d", result);

    return 0;
}
```

First and Last Digit

Problem Statement

You are given a number N and your task is to find the first and last digit of the number N.

Input Format

The first line contains the single integer N.

Output format

Print the first and last digits of the number separated by space.

Constraints

1 <= N <= 10^4

Example

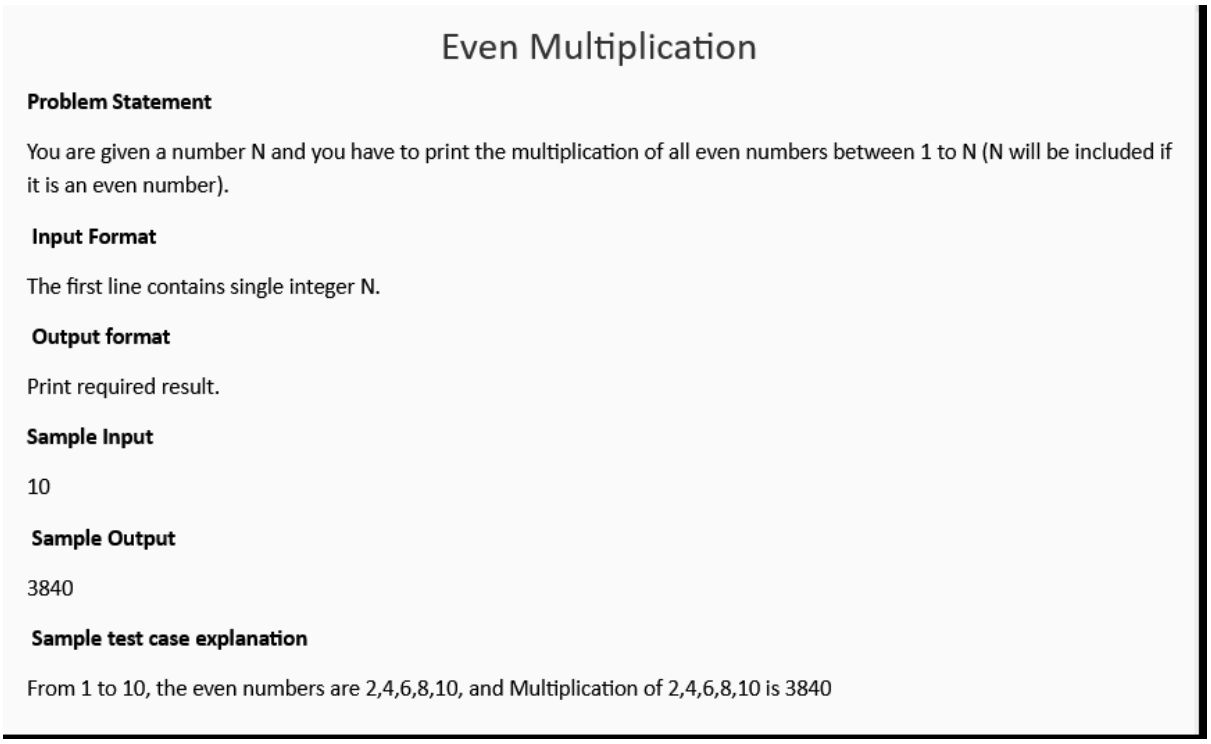
Sample Input

998

Sample Output

9 8 //First and last digit separated by Space.

```c
#include<stdio.h>


int main()

{
    int n;
    scanf("%d", &n);
    int l = n%10;
    int f;
    while(n!=0){
        f = n%10;
        n /= 10;
    }

    printf("%d %d", f,l);
 return 0;

}
```



Even Multiplication

Problem Statement

You are given a number N and you have to print the multiplication of all even numbers between 1 to N (N will be included if it is an even number).

Input Format

The first line contains single integer N.

Output format

Print required result.

Sample Input

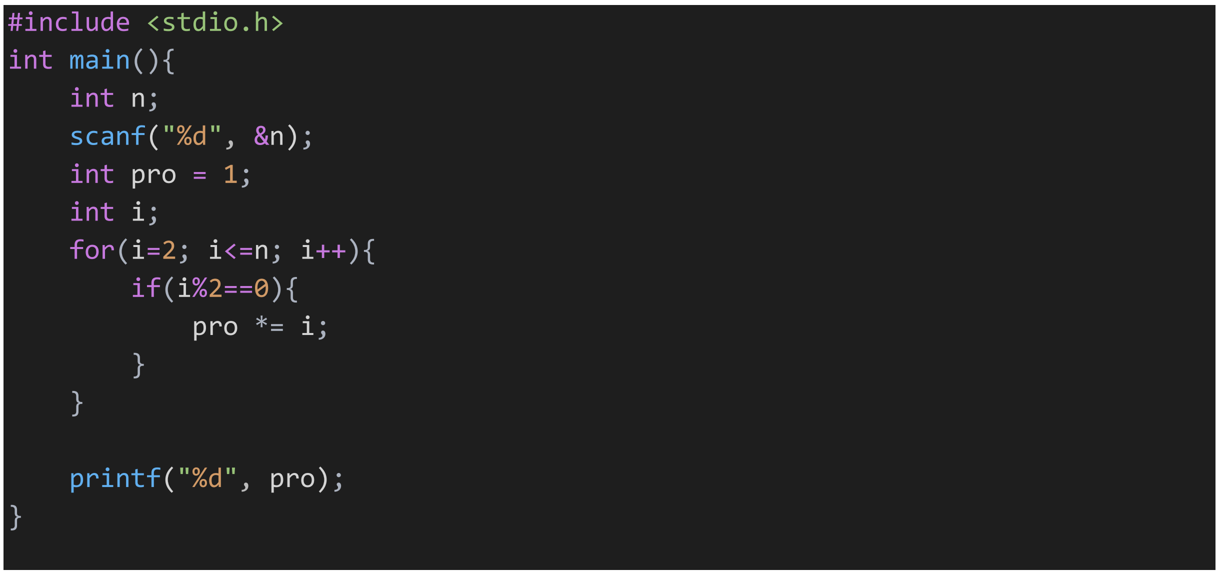
10

Sample Output

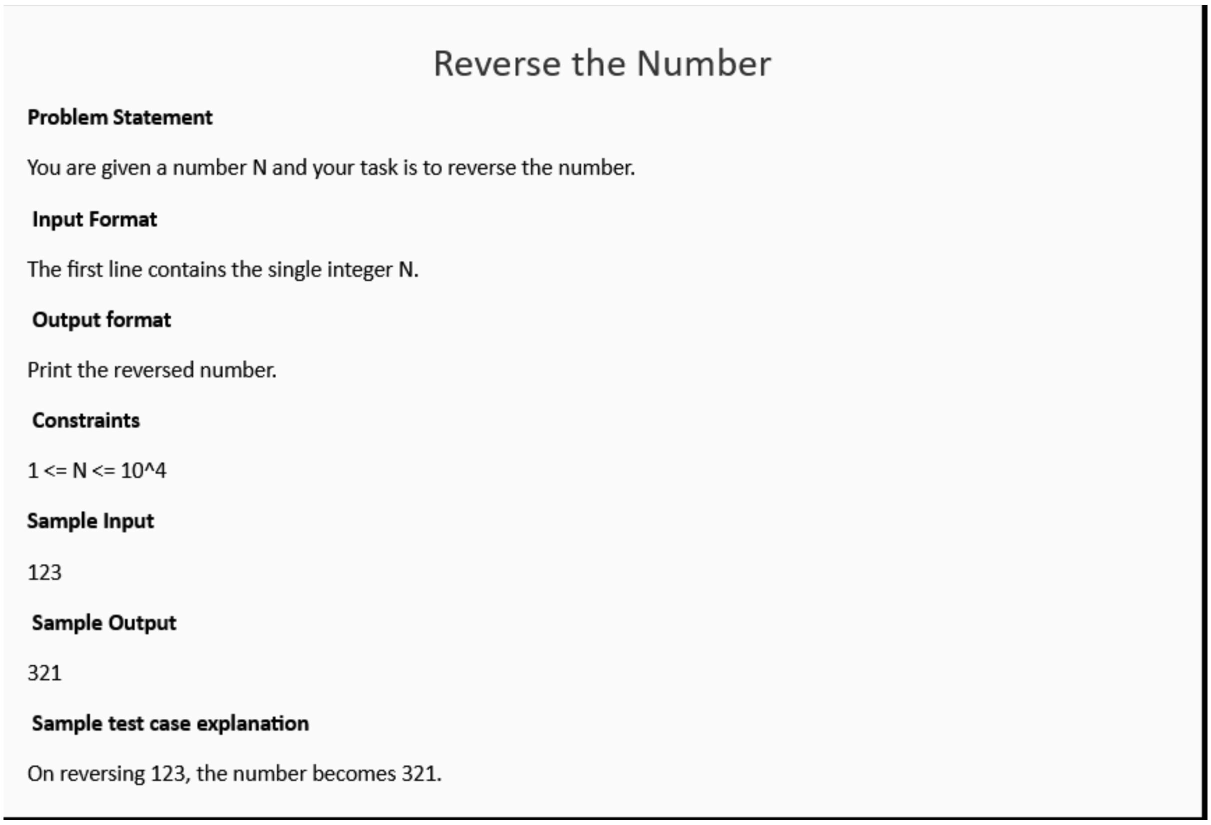
3840

Sample test case explanation

From 1 to 10, the even numbers are 2,4,6,8,10, and Multiplication of 2,4,6,8,10 is 3840



```c
#include <stdio.h>
int main(){
    int n;
    scanf("%d", &n);
    int pro = 1;
    int i;
    for(i=2; i<=n; i++){
        if(i%2==0){
            pro *= i;
        }
    }

    printf("%d", pro);
}
```



Reverse the Number

Problem Statement

You are given a number N and your task is to reverse the number.

Input Format

The first line contains the single integer N.

Output format

Print the reversed number.

Constraints

1 <= N <= 10^4

Sample Input

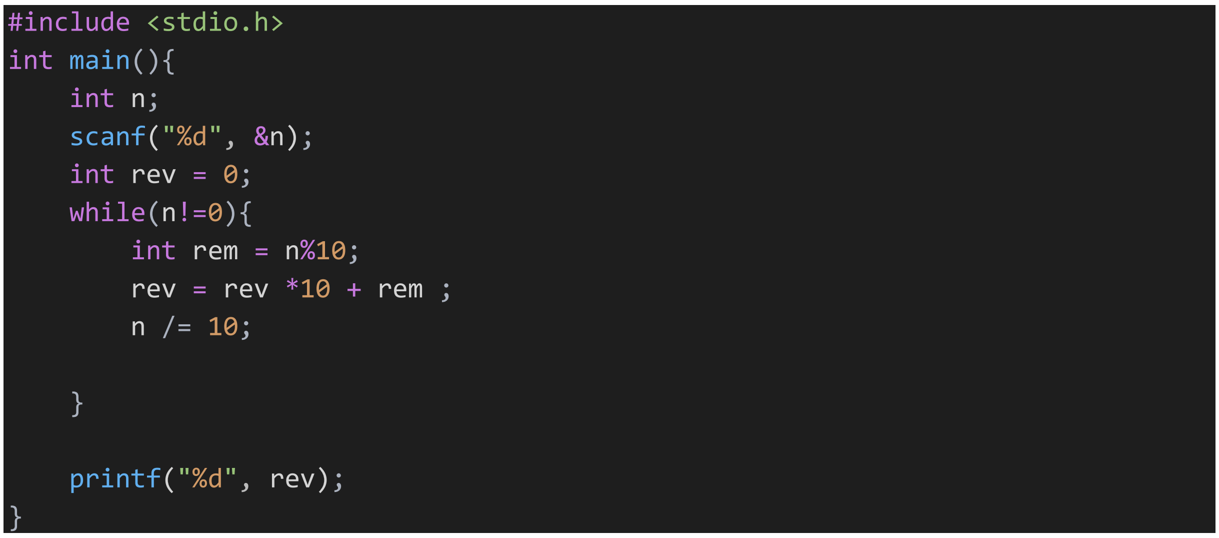
123

Sample Output

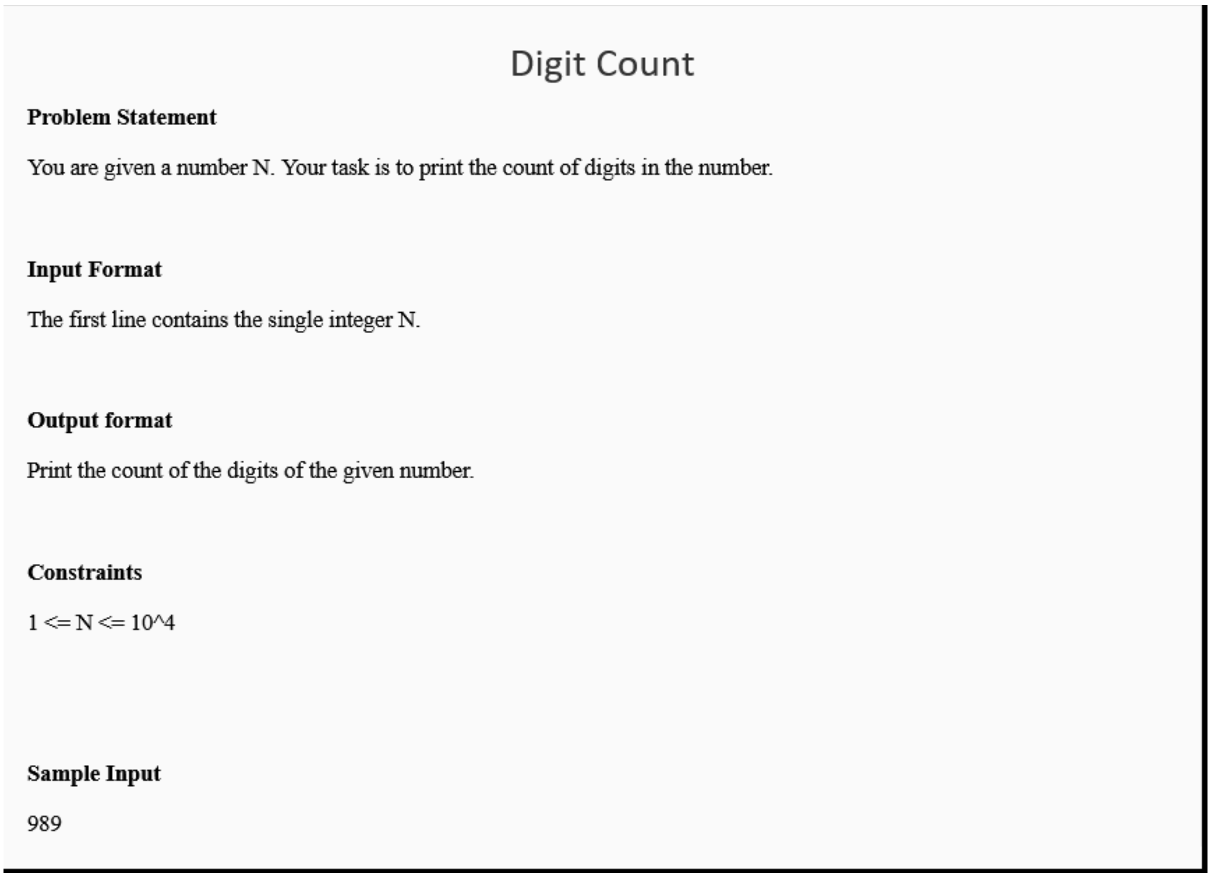
321

Sample test case explanation

On reversing 123, the number becomes 321.



```c
#include <stdio.h>
int main(){
    int n;
    scanf("%d", &n);
    int rev = 0;
    while(n!=0){
        int rem = n%10;
        rev = rev *10 + rem ;
        n /= 10;

    }

    printf("%d", rev);
}
```



Digit Count

Problem Statement

You are given a number N. Your task is to print the count of digits in the number.

Input Format

The first line contains the single integer N.

Output format

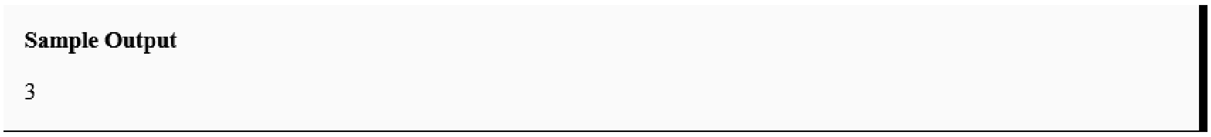
Print the count of the digits of the given number.

Constraints

1 <= N <= 10^4

Sample Input

989



Sample Output

3

```
#include <stdio.h>

int main()

{
    int n;
    scanf("%d", &n);

    int count = 0;
    while(n!=0){
        count++;
        n/=10;
    }

    printf("%d", count);
    return 0;

}
```

Solutions Proposed by: Dhruv