# **Perfect Number**

Check if a given number is perfect or not. A number is perfect if sum of factorial of its digit is equal to the given number.

For example 145 is a Perfect Number because 1! + 4! + 5! = 145

You have been given a complete main() function,complete the code of PerfectSum() function and return the sum of factorial

of digits of that number.

Input:

The first line of input contains an integer T denoting the number of test cases. Then T test cases follow. The next T lines will contain an integer N.

Output:

Corresponding to each test case, in a new line, print "Perfect " if it follow above condition else print "Not Perfect" without quotes.

Constraints:

1 ≤ T ≤ 50

1 ≤ N ≤ 10000

**Sample Input:**

2

23

145

**Sample Output:**

NO

YES

**Explanation:**

23 is not a perfect number because 2!+3!=8 So ans is NO

145 is a perfect number because 1!+4!+5!=145

**int PerfectSum(int n)**

**{**

**int a,i,f,sum=0;**

**while(n>0){**

**a=n%10;f=1;**

**for(i=1;i<=a;i++)f=f\*i;**

**sum=sum+f;**

**n=n/10;**

**}**

**return sum;**

**}**

**int main() {**

**int testcase;**

**scanf("%d",&testcase);**

**while(testcase--)**

**{**

**int n;**

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

**int sum=PerfectSum(n);**

**if(sum==n)**

**printf("YES\n");**

**else**

**printf("NO\n");**

**}**

**return 0;**

**}**

# **Sum of Factorial of Even Digits**

You have been given a task to accept a number from the user.Find the sum of factorial of all even digits in that number.

You have been given a complete main() function.Complete the code of SumFactEven() function and return the sum of factorial of

even numbers in that number.

**Sample Input**

3

152

221

214

**Sample Output**

2

4

26

**Explanation**: In Sample output

152 has one digit as even i.e 2 and 2!=2 so sum=2.

221 has two digits as even and 2!=2,2!=2 so sum=2+2=4

214 has two digits as even i.e 2!=2 ,4!=24 so sum=2+24=26

**int SumFactEven(int n)**

**{**

**int a,i,f,sum=0;**

**while(n>0){**

**a=n%10;f=1;**

**if(a%2==0){for(i=1;i<=a;i++)f=f\*i;**

**sum=sum+f;}**

**n=n/10;**

**}**

**return sum;**

**}**

**int main() {**

**int testcase;**

**scanf("%d",&testcase);**

**while(testcase--)**

**{**

**int num;**

**scanf("%d",&num);**

**int sum=SumFactEven(num);**

**printf("%d\n",sum);**

**}**

**return 0;**

**}**

# **Sum of Prime Numbers**

Write a program to accept a number n from user and accept n elements in an array

Display Sum of all Prime numbers entered in an array.

You have been given complete main() function.Write the code for PrimeSum() function and return the sum

of prime numbers in array.

Sample Input

5

11 33 44 66 13

Sample output

24

**Explanation** :24 is sum of prime numbers i.e 11+13

**int PrimeSum(int arr[],int n)**

**{int sum=0,flag,e,i,j;**

**for(i=0;i<n;i++){**

**e=arr[i];flag=0;**

**for(j=2;j<e;j++){**

**if(e%j==0){**

**flag=1;**

**break;**

**}**

**}**

**if(flag==0)sum=sum+e;**

**}**

**return sum;**

**}**

**int main() {**

**int n,i,arr[100],sum;**

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

**for(i=0;i<n;i++)**

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

**sum=PrimeSum(arr,n);**

**printf("%d",sum);**

**return 0;**

**}**

# 

# **Armstrong Number**

You have been given a task to accept a number and check whether it is an Armstrong number or not.

An Armstrong number is an integer such that the sum of the cubes of its digits is equal to the number itself. For example, 371 is an Armstrong number since 3\*3\*3 + 7\*7\*7 + 1\*1\*1 = 371.

The main() function has been given to you.Complete the CheckArmstrong() number and return 1 if it is an armstrong number otherwise return 0.

Sample Input

3

153

12

13

Sample Output

YES

NO

NO

**YOUR ANSWER**

**int CheckArmstrong(int n)**

**{int a=n,i,j,k=0;**

**while(n>0){**

**i=n%10;**

**i=i\*i\*i;**

**k=k+i;n=n/10;**

**}**

**if(k==a)return 1;**

**else return 0;**

**}**

**int main() {**

**int testcase;**

**scanf("%d",&testcase);**

**while(testcase--)**

**{**

**int n;**

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

**int r=CheckArmstrong(n);**

**if(r==1)**

**printf("YES\n");**

**else**

**printf("NO\n");**

**}**

**return 0;**

**}**

# **Count of a number in a sorted list with duplicates**

**Problem statement**

Given a sorted list of integers with duplicates, find the count of a given number in this list .You have been given main() function ,complete the code of CountDuplicate() function and return

count of duplicates of given number.

**Input**

First line of input will contain a positive integer T = number of test cases. Each test case will contain 2 lines. First line of each test case will contain two number N = number of elements in list and K separated by space. Next line will contain N space separated integers.

**Output**

For each test case, print on a single line, the count of number K in this list.

**Sample Input**

3  
10 5  
1 2 2 5 5 5 7 7 7 8  
10 1  
1 1 1 1 1 1 1 2 2 3  
20 2  
1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4

**Sample Output**

3  
7  
5

**YOUR ANSWER**

**int countDuplicates(int a[],int n,int num)**

**{int i,c=0;**

**for(i=0;i<n;i++){**

**if(a[i]==num)c++;**

**}**

**return c;**

**}**

**int main() {**

**int testcase;**

**scanf("%d",&testcase);**

**while(testcase--)**

**{**

**int n,a[100],num,i,j,count;**

**scanf("%d%d",&n,&num);**

**for(i=0;i<n;i++)**

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

**count=countDuplicates(a,n,num);**

**printf("%d\n",count);**

**}**

**return 0;**

**}**

# **Third Largest number**

Given an array of distinct elements, Your task is to find the third largest element in it. You have to complete the function thirdLargest which takes two argument . The first argument is the array a[] and the second argument is the size of the array (n). The function returns an integer denoting the third largest element in the array a[].

The function should return -1 if there are less than 3 elements in input.

Input:

The first line of input contains an integer T denoting the number of test cases. Then T test cases follow . The first line of each test case is N,N is the size of array.The second line of each test case contains N space separated values of the array a[ ].

Output:

Output for each test case will be the third largest element of the array .

Constraints:

1 ≤ T ≤ 100

1 ≤ N ≤ 100

1 ≤ A[ ] ≤ 100

Example(To be used for only expected output):

Input:

1

5

2 4 1 3 5

Output:

3

**YOUR ANSWER**

int ThirdLargest(int a[],int n)

{

int large,i,id;

large=a[0];

for(i=0;i<n;i++){

if(large<a[i]){

large=a[i];

id=i;

}

}

a[id]=-1000;

large=-10;

for(i=0;i<n;i++){

if(large<a[i]){

large=a[i];

id=i;

}

}

a[id]=-1000;

large=-10;

for(i=0;i<n;i++){

if(large<a[i]){

large=a[i];

id=i;

}

}

return a[id];

}

int main() {

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int n,i;

scanf("%d",&n);

int a[100];

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

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

int number=ThirdLargest(a,n);

printf("%d\n",number);

}

return 0;

}

# **Verify Prime number**

Given a number, verify whether its prime or not.You have been given main() function,complete the code

of checkPrime() and return 1 if number is prime else return 0.

Input

First line of input will contain a number N = number of test cases. Next N lines will contain number n as test case where 0<=n<=1000000000

Output

For each input case, print "PRIME" if the number is prime, else print "NOT PRIME" (quotes for clarity)

Sample Input

4

0

1

2

3

Sample Output

NOT PRIME

NOT PRIME

PRIME

PRIME

**YOUR ANSWER**

int checkPrime(int n)

{ int flag,i;

if(n==0||n==1){

return 0;}

else {

flag=0;

for(i=2;i<n;i++){

if(n%i==0){

flag=1;

break;

}

}

if(flag==0)return 1;

else return 0;

}

}

int main()

{

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int num;

scanf("%d",&num);

int r=checkPrime(num);

if(r==1)

printf("PRIME\n");

else

printf("NOT PRIME\n");

}

}

# **Count the Zero's**

Given an array of size N consisting of only 0's and 1's.

You have to find the count of all the 0's.You have been given main() function,

Complete the CountZero(int a[],int n) function and return the count of zero's from this function.

Input:

The first line of input contains an integer T denoting the number of test cases. Then T test cases follow.

The first line of each test case contains an integer N, where N is the size of the array A[ ].

The second line of each test case contains N space separated integers consisting 1's and 0's, denoting elements of the array A[ ].

Output:

Print out the number of 0's in the array.

Constraints:

1 <= T <= 100

1 <= N <= 30

0 <= A[i] <= 1

Example :

Input:

3

12

1 1 1 1 1 1 1 1 1 0 0 0

5

0 0 0 0 0

6

1 1 1 1 1 1

Output:

3

5

0

int CountZero(int a[],int n)

{int i,c=0;

for(i=0;i<n;i++)if(a[i]==0)c++;

return c;

}

int main() {

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int n,a[100],i,j,count;

scanf("%d",&n);

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

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

count=CountZero(a,n);

printf("%d\n",count);

}

return 0;

}

# **Kth smallest element**

**Problem statement**

Given a sequence of integers and a number k, find the Kth smallest number in this sequence.You have been given main() function ,complete int KthSmallest(int a[],int n,int K)

function to return Kth smallest element from the array.

**Input**

First line of input will contain a number T = number of test cases. Each test case will contain two lines. First line will contain two integers N = number of elements in sequence, and K. Next line will contain a space separated list of N integers.

**Output**

For each test case, print the Kth smallest number in the sequence on a single line.

**Sample Input**

3  
5 2  
3 1 4 8 5  
7 5  
7 6 5 4 3 2 1  
3 1  
12 15 13

**Sample Output**

3  
5  
12

**YOUR ANSWER**

int KthSmallest(int a[],int n,int K)

{

int c,d,position,swap;

for ( c = 0 ; c < ( n - 1 ) ; c++ )

{

position = c;

for ( d = c + 1 ; d < n ; d++ )

{

if ( a[position] > a[d] )

position = d;

}

if ( position != c )

{

swap = a[c];

a[c] = a[position];

a[position] = swap;

}

}

return a[K-1];

}

int main() {

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int n,i,K;

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

int a[100];

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

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

int smallest=KthSmallest(a,n,K);

printf("%d\n",smallest);

}

return 0;

}

# **Maximum and Minimum element in array**

Given an array, find maximum and minimum elements from the array.You have been given main() function,Complete MaxMin() function to find maximum

and minimum element in array.

Input:

The first line of input contains an integer T denoting the number of test cases. The description of T test cases follows. The first line of each test case contains a single integer N denoting the size of array. The second line contains N space-separated integers A1, A2, ..., AN denoting the elements of the array.

Output:

Print the maximum and minimum element in a single line with space in between.

Constraints:

1 ≤ T ≤ 30

1 ≤ N ≤ 100

0 ≤A[i]<100

Example:

Input:

2

4

5 4 2 1

1

8

Output:

5 1

8 8

**YOUR ANSWER**

void MaxMin(int a[],int n)

{int large=a[0],small=a[0],i;

for(i=0;i<n;i++){

if(small>a[i])small=a[i];

if(large<a[i])large=a[i];

}

printf("%d %d",large,small);

printf("\n");

}

int main() {

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int n;

scanf("%d",&n);

int a[100],i,j;

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

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

MaxMin(a,n);

}

return 0;

}

# **Search a Number**

Given an integer array and an element x, find if element is present in array or not. If element is present, then print index of its first occurrence. Else print -1.

Input:

First line contains an integer, the number of test cases 'T' Each test case should contain an integer, size of array 'N' in the first line. In the second line Input the integer elements of the array in a single line separated by space. Element X should be inputted in the third line after entering the elements of array.

Output:

print the output in a separate line returning the index of the element X.If element not present then print -1.

Constraints:

1 <= T <= 100

1 <= N <= 100

1 <= Arr[i] <= 100

Example:

Input:

1

4

1 2 3 4

3

Output:

2

Explanation:

There is one test case with array as {1, 2, 3 4} and element to be searched as 3. Since 3 is present at index 2, output is 2

**YOUR ANSWER**

int SearchIndex(int a[],int n,int n1)

{int id=0,i;

for(i=0;i<n;i++)if(a[i]==n1){id=1;break;}

if(id==1)return i;

else return -1;

}

int main() {

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int n,a[100],num;

scanf("%d",&n);

for(int i=0;i<n;i++)

{

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

}

scanf("%d",&num);

int index=SearchIndex(a,n,num);

printf("%d",index);

}

return 0;

}

# **How many Swaps in selection sort ?**

**Problem statement**

As the problem title suggests, you need to count the number of swaps required to sort a given list of integers in ascending order using the selection sort algorithm.

You have been given the main() function ,complete the CountSwap() function.

**Input**

First line of the input contains the number of test cases **T**. T test cases follow. The first line of each test case contains the **N**, (1 ≤ N ≤ 50) the number of integers in the array. The second line contains N space separated integers. All N numbers are distinct.

**Output**

For each test case, print on one line, the number of swaps (minimum) to sort the given array in ascending order using selection sort.

**Sample Input**

2  
3  
4 2 5  
5  
10 11 8 7 1

**Sample Output**

1  
3

**YOUR ANSWER**

int CountSwap(int a[],int n)

{

int c,d,position,swap,c1=0;

for ( c = 0 ; c < ( n - 1 ) ; c++ )

{

position = c;

for ( d = c + 1 ; d < n ; d++ )

{

if ( a[position] > a[d] )

position = d;

}

if ( position != c )

{c1++;

swap = a[c];

a[c] = a[position];

a[position] = swap;

}

}

return c1;}

int main() {

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int n,a[50],i,count;

scanf("%d",&n);

for(int i=0;i<n;i++)

{

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

}

count=CountSwap(a,n);

printf("%d\n",count);

}

return 0;

}

# **Bubble Sort Swaps**

**Problem statement**

Given a list of integers with no duplicates, find out how many swaps it will take to sort the list in increasing order using Bubble sort.You have been given main() function

complete the CountSwap(int a[],int n) function and return count of swaps.

**Input**

First line of each test case will contain an integer T = number of test cases. Each test case will contain two lines. First line will contain a number N = no. of elements in the list. Next line will contain N space separated numbers. 1 <= N <= 50.

**Output**

For each test case, print on a single line, number of swaps required to sort the list in increasing order using Bubble sort.

**Sample Input**

3  
5  
2 1 4 6 3  
10   
123 21 34 45 25 675 23 44 55 900  
1  
23

**Sample Output**

3  
16  
0

**YOUR ANSWER**

int CountSwap(int array[],int n)

{

int c,d,swap,c1=0;

for (c = 0 ; c < ( n - 1 ); c++)

{

for (d = 0 ; d < n - c - 1; d++)

{

if (array[d] > array[d+1])

{c1++;

swap = array[d];

array[d] = array[d+1];

array[d+1] = swap;

}

}

}return c1;

}

int main()

{

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int n;

scanf("%d",&n);

int a[100];

int i;

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

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

int count=CountSwap(a,n);

printf("%d\n",count);

}

return 0;

}

# **Binary to Decimal**

**Problem statement**

Given the binary representation of a number, convert it to its decimal form.You have been given main() function ,complete the code of BinaryToDecimal() function and return decimal

number from function.

**Input**

The first line will contain an integer T = number of test case. Next T lines will contain a number representing the binary form of a number.

**Output**

For each number(binary form), print on a single line, the number in decimal number system.

**Sample Input**

4  
101  
100  
1011  
1111

**Sample Output**

5  
4  
11  
15

**YOUR ANSWER**

int BinaryToDecimal(int n)

{int c=0,a,sum=0;

while(n>0){

a=n%10;

sum=sum+a\*pow(2,c);

n=n/10;c++;

}

return sum;

}

int main() {

int testcase;

scanf("%d",&testcase);

while(testcase--)

{

int num;

scanf("%d",&num);

int r=BinaryToDecimal(num);

printf("%d\n",r);

}

return 0;

}

# **Sum of digits**

**Problem statement**

Given a natural number, find the sum of all its digits.You have been given main() function ,Complete SumDigits(int n) function and return the sum of digits

of given number.

**Input**

First line will contain T = number of test cases. Next T lines will each contain a non-negative integer N.

**Output**

For each test case, print on a separate line, the sum of its digits.

**Sample Input**

5  
123  
321  
2345  
1234  
12

**Sample Output**

6  
6  
14  
10  
3

**YOUR ANSWER**

int SumDigits(int n)

{int sum=0;

while(n>0){

sum=sum+n%10;

n=n/10;

}return sum;

}

# **Minimum Number Of Swaps**

Problem statement

As the problem title suggests, you need to count the number of swaps required to sort a given list of integers in ascending order using the selection sort algorithm.

Input

First line of the input contains the number of test cases T. T test cases follow. The first line of each test case contains the N, (1 ≤ N ≤ 50) the number of integers in the array. The second line contains N space separated integers. All N numbers are distinct.

Output

For each test case, print on one line, the number of swaps (minimum) to sort the given array in ascending order using selection sort.

Sample Input

2

3

4 2 5

5

10 11 8 7 1

Sample Output

1

3

void selectionSort(int a[],int n)

{

int c,d,position,swap,c1=0;

for ( c = 0 ; c < ( n - 1 ) ; c++ )

{

position = c;

for ( d = c + 1 ; d < n ; d++ )

{

if ( a[position] > a[d] )

position = d;

}

if ( position != c )

{c1++;

swap = a[c];

a[c] = a[position];

a[position] = swap;

}

}

count=c1;}

# **Selection Sort**

U have been given the task to accept n elements in array and display the the elements after every pass

Sample Input

7

64 34 25 12 22 11 90

Sample Output

11 34 25 12 22 64 90

11 12 25 34 22 64 90

11 12 22 34 25 64 90

11 12 22 25 34 64 90

11 12 22 25 34 64 90

11 12 22 25 34 64 90

11 12 22 25 34 64 90

Explanation

7 is number of elements

64 34 25 12 22 11 90 are elements of an array

11 34 25 12 22 64 90 is contents of array after first pass

11 12 25 34 22 64 90 after second pass

11 12 22 34 25 64 90 after third pass

11 12 22 25 34 64 90 after fourth pass

and so on....

void selectionSort(int a[], int n)

{

int c,d,position,swap;

for ( c = 0 ; c < ( n - 1 ) ; c++ )

{

position = c;

for ( d = c + 1 ; d < n ; d++ )

{

if ( a[position] > a[d] )

position = d;

}

if ( position != c )

{

swap = a[c];

a[c] = a[position];

a[position] = swap;

}

for(d=0;d<n;d++){

printf("%d ",a[d]);

}printf("\n");

}

}

# **Move all zeroes to front of array**

Given an array of random numbers, Push all the zero’s of a given array to the front of the array. For example, if the given arrays is {1, 9, 8, 4, 0, 0, 2, 7, 0, 6, 0}, it should be changed to {0,0,0,0,1, 9, 8, 4, 2, 7, 6}.

Input:

The first line contains an integer 'T' denoting the total number of test cases. In each test cases, First line is number of elements in array 'N' and second its values.

Output:

Print the array after shifting all 0's at the front.​

Note: An extra space is expected at the end after output of every test case

Constraints:

1 <= T <=100

1 <= N <=1000

0 <= a[i] <=100

Example:

Sample Input:

1

5

3 5 0 0 4

Sample Output:

0 0 3 5 4

**YOUR ANSWER**

#include <math.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <assert.h>

#include <limits.h>

#include <stdbool.h>

int main() {

int t,c,n,a[1000]={0},k,i,j;

scanf("%d",&t);

while(t--){c=0;

scanf("%d",&n);

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

for(i=0;i<n;i++){if(a[i]==0){for(j=c;j<n;j++){

if(a[j]!=0){

k=a[i];

a[i]=a[j];

a[j]=k;c++;

break;

}

}}

}

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

printf("\n");

}

return 0;

}

# **Leaders**

Write a program to print all the LEADERS in the array. An element is leader if it is greater than all the elements to its right side.

And the rightmost element is always a leader.

For example int the array {16, 17, 4, 3, 5, 2}, leaders are 17, 5 and 2.

Let the input array be arr[] and size of the array be N.

constraints

1<=N<10000

1<=arr[i]<10000

Sample Input

6

16 17 4 3 5 2

Sample output

17 5 2

Explanation: In sample input 6 represent the size of the array and second line indicates elements of the array.

In Sample input 17 is greater than all numbers on its right side ,5 is also greater than all its right numbers and 2 is rightmost element.

**YOUR ANSWER**

void printLeaders(int a[], int n)

{

int i,j,e,flag;

for(i=0;i<n;i++){

e=a[i];flag=0;

for(j=i+1;j<n;j++){

if(e<a[j])flag=1;

}

if(flag==0||i==n)printf("%d ",e);

}

}

# **Reverse Array in Group**

Given an array, reverse every sub-array formed by consecutive k elements.

Input:

The first line of input contains a single integer T denoting the number of test cases. Then T test cases follow. Each test case consist of two lines. The first line of each test case consists of an integer N, where N is the size of array.The second line of each test case contains N space separated integers denoting array elements.The third line of each test case consists of an integer K.

Output:

Corresponding to each test case, in a new line, print the modified array.

Constraints:

1 ≤ T ≤ 100

1 ≤ N ≤ 500

1 ≤ A[i] ≤ 500

Example:

Sample Input

1

5

1 2 3 4 5

3

Sample Output

3 2 1 5 4

**YOUR ANSWER**

#include <math.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <assert.h>

#include <limits.h>

#include <stdbool.h>

int main() {

int t,i,k,o,n,a[100]={0},m,n1,n2=0,flag=0;

scanf("%d",&t);

while(t--){

scanf("%d",&n);flag=0;

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

scanf("%d",&k);o=0;n2=0;

if(n%k==0)n1=n/k;

else {n1=(n/k)+1;flag=1;}

while(n1--){

m=a[o];

a[o]=a[o+k-1];

a[o+k-1]=m;o=o+k;

if(flag==1&&n1==1){

m=a[o];

a[o]=a[o+k-2];

a[o+k-2]=m;

break;

}

}

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

printf("\n");

}

return 0;

}