

TCS Digital

Day 1 Slot 1

Question 1

Problem Description -: Given an array `Arr[]` of `N` integers and a positive integer `K`. The task is to cyclically rotate the array clockwise by `K`.

Note : Keep the first of the array unaltered.

Example 1:

- 5 —Value of `N`
- {10, 20, 30, 40, 50} —Element of `Arr[]`
- 2 —Value of `K`

Output :

40 50 10 20 30

Example 2:

- 4 —Value of `N`
- {10, 20, 30, 40} —Element of `Arr[]`
- 1 —Value of `K`

Output :

40 10 20 30

C++

```

#include<bits/stdc++.h>
using namespace std;

vector<int> rotate(int nums[], int n, int k) {

    if (k > n)
        k = k % n;

    vector<int> ans(n);

    for (int i = 0; i < k; i++) {
        ans[i] = nums[n - k + i];
    }

    int index = 0;
    for (int i = k; i < n; i++) {
        ans[i] = nums[index++];
    }
    return ans;
}

int main()
{
    int Array[] = { 1, 2, 3, 4, 5 };
    int N = sizeof(Array) / sizeof(Array[0]);
    int K = 2;

    vector<int> ans = rotate(Array, N, K);
    for (int i = 0; i < N; ++i) {
        cout << ans[i] << ' ';
    }
}

```

Java

[Link for this Question](#)

TCS Digital Day 1 Slot 1 Question 2

Problem Description -: Given two non-negative integers n1 and n2, where n1

For example:

Suppose n1=11 and n2=15.

There is the number 11, which has repeated digits, but 12, 13, 14 and 15 have no repeated digits. So, the output is 4.

Example1:

Input:

- 11 – Vlaue of n1
- 15 – value of n2

Output:

- 4

Example 2:

Input:

- 101 – value of n1
- 200 – value of n2

Output:

- 72

C++

```
#include<bits/stdc++.h>
using namespace std;

int find(int n1, int n2) {
    int count = 0;
    for (int i = n1 ; i <= n2 ; i++) {
        int num = i;

        vector<bool> visited;
        visited.assign(10, false);

        while (num > 0) {
            if (visited[num % 10] == true)
                break;
            visited[num % 10] = true;
            num /= 10;
        }

        if (num == 0)
            count++;
    }
    return count;
}

int main()
{
    int n1 = 101, n2 = 200;
    cout << find(n1, n2);
}
```

Java

Question 3

Problem Description -: In this 3 Palindrome, Given an input string word, split the string into exactly 3 palindromic substrings. Working from left to right, choose the smallest split for the first substring that still allows the remaining word to

be split into 2 palindromes.

Similarly, choose the smallest second palindromic substring that leaves a third palindromic substring.

If there is no way to split the word into exactly three palindromic substrings, print “Impossible” (without quotes). Every character of the string needs to be consumed.

Cases not allowed –

- After finding 3 palindromes using above instructions, if any character of the original string remains unconsumed.
- No character may be shared in forming 3 palindromes.

Constraints

- 1 <= the length of input sting <= 1000

Input

- First line contains the input string consisting of characters between [a-z].

Output

- Print 3 substrings one on each line.

Time Limit

1

Examples

Example 1

Input

nayannamantenet

Output

nayan

naman

tenet

Explanation

- The original string can be split into 3 palindromes as mentioned in the output.
- However, if the input was nayanamantenet, then the answer would be “Impossible”.

Example 2

Input

aaaaaa

Output

a

a

aaa

Explanation

- The other ways to split the given string into 3 palindromes are as follows –
- [a, aaa, a], [aaa, a, a], [aa, aa, a], etc.
- Since we want to minimize the length of the first palindromic substring using left to right processing, the correct way to split is [a, a, aaa].

Example 3

Input

aaaabaaaa

Output

a

aaabaaa

a

Explanation

- The other ways to split the given string into 3 palindromes are as follows –
- [aaaa, b, aaaa], [aa, aabaa, aa], etc.
- Since we want to minimize the length of the first palindromic substring using left to right processing, the correct way to split is [a, aaabaaa, a].

C++

```
#include<bits/stdc++.h>
typedef long long int lld;
#define mod 1000000007
using namespace std;
bool pali(string s)
{
    if(s.length()==1) return true;
    string s1=s;reverse(s1.begin(),s1.end());
    return (s1==s);
}
int main()
{
    speed;
    string s,s1,s2,s3;
    cin>>s;
    int l=s.length();
    for(int i=1;i<l-1;i++)
    {
        s1=s.substr(0,i);
        if(pali(s1))
            for(int j=1;j<l-i;j++)
            {
                s2=s.substr(i,j);s3=s.substr(i+j,l-i-j);
                if(pali(s2)&&pal(s3))
                {
```

<pre> cout<<s1<<endl<<s2<<endl<<s3;return 0; } } } cout<<"Impossible"; return 0; }</pre>
Python
Java

Question 4

Problem Statement -: In this even odd problem Given a range [low, high] (both inclusive), select K numbers from the range (a number can be chosen multiple times) such that sum of those K numbers is even.

Calculate the number of all such permutations.

As this number can be large, print it modulo (1e9 +7).

Constraints

- 0 <= low <= high <= 10^9
- K <= 10^6.

Input

- First line contains two space separated integers denoting low and high respectively
- Second line contains a single integer K.

Output

- Print a single integer denoting the number of all such permutations

Time Limit

1

Examples

Example 1

Input

4 5

3

Output

4

Explanation

There are 4 valid permutations viz. {4, 4, 4}, {4, 5, 5}, {5, 4, 5} and {5, 5, 4} which sum up to an even number.

Example 2

Input

1 10

2

Output

50

Explanation

There are 50 valid permutations viz. {1,1}, {1, 3},... {1, 9} {2,2}, {2, 4},... {2, 10} . . . {10, 2}, {10, 4},... {10, 10}. These 50 permutations, each sum up to an even number.

C++

```
#include<bits/stdc++.h>
using namespace std;
typedef long long int lld;
#define mod 1000000007

long e_sum(long m,long n,long K,long N)
{
    if(K==1)
    {
        return n;
    }
    else
    {
        return (N-(m-n)*e_sum(m,n,K-1,N)%(1000000007));
    }
}

int main()
{
    long low,high,K,m,n,diff,Out,N,i;
    scanf("%ld",&low);
    scanf("%ld",&high);
    scanf("%ld",&K);
    diff=high-low+1;
    if(diff%2==0)
    {
        m=diff/2;
        n=m;
    }
    else
    {
        if(low%2==0)
        {
            m=(diff-1)/2;
            n=m+1;
        }
        else
        {
            m=(diff+1)/2;
            n=m-1;
        }
    }
    Out=e_sum(m,n,K,N);
    printf("%ld",Out);
    return 0;
}
```

<pre> n=m-1, } } N=m; for(i=0;i<K-1;i++) { N=(N*(m+n))%1000000007; } Out=e_sum(m,n,K,N)%1000000007; printf("%ld",Out); return 0; }</pre>
Python
Java

Question 5

Roco is an island near Africa which is very prone to forest fire. Forest fire is such that it destroys the complete forest. Not a single tree is left. This island has been cursed by God, and the curse is that whenever a tree catches fire, it passes the fire to all its adjacent tree in all 8 directions, North, South, East, West, North-East, North-West, South-East, and South-West. And it is given that the fire is spreading every minute in the given manner, i.e. every tree is passing fire to its adjacent tree. Suppose that the forest layout is as follows where T denotes tree and W denotes water.

Your task is that given the location of the first tree that catches fire, determine how long would it take for the entire forest to be on fire. You may assume that the layout of the forest is such that the whole forest will catch fire for sure and that there will be at least one tree in the forest

Input Format:

- First line contains two integers, M, N, space separated, giving the size of the forest in terms of the number of rows and columns respectively.
- The next line contains two integers X, Y, space separated, giving the coordinates of the first tree that catches the fire.
- The next M lines, where ith line containing N characters each of which is either T or W, giving the position of the Tree and Water in the ith row of the forest.

Output Format:

Single integer indicating the number of minutes taken for the entire forest to catch fire

Constraints:

- $3 \leq M \leq 20$
- $3 \leq N \leq 20$

Sample Input 1:

```
3 3
W T T
T W W
W T T
```

Sample Output 1:

5

Explanation:

In the second minute, tree at (1,2) catches fire, in the third minute, the tree at (2,1) catches fire, fourth minute tree at (3,2) catches fire and in the fifth minute the last tree at (3,3) catches fire.

Sample Input 2:

```
6 6
1 6
W T T T T T
T W W W W W
W T T T T T
W W W W W T
T T T T T T
T W W W W W
```

Sample Output 2:

16

C++

```
#include <bits/stdc++.h>
using namespace std;

char f[21][21];
int n,m;
struct node{int a,b;};
bool valid(int x,int y) {return (x>=0&&y>=0&&x<n&&y<m);}
bool step(node temp){return (temp.a== -1&&temp.b== -1);}
int main()
{
    cin>>n>>m;
    int x,y,i,j,count=0;int ans=1;
    cin>>x>>y;x--;y--;
    for(i=0;i<n;i++)
        for(j=0;j<m;j++)
            cin>>f[i][j];
    f[x][y]='X';

    queue<node>q;
    node temp;
    temp.a=x;temp.b=y;
    q.push(temp);
    temp.a=-1;temp.b=-1;
    q.push(temp);

    while(!q.empty())
```

```

{
    bool flag=false;
    while(!step(q.front()))
    {
        node count=q.front();
        if(valid(count.a+1,count.b)&&f[count.a+1][count.b]=='T')//a+1,b
        {
            if(flag==false){flag=true;ans++;}
            f[count.a+1][count.b]='X';
            count.a++;
            q.push(count);
            count.a--;
        }
        if(valid(count.a+1,count.b+1)&&f[count.a+1][count.b+1]=='T')//a+1,b+1
        {
            if(flag==false){flag=true;ans++;}
            f[count.a+1][count.b+1]='X';
            count.a++;count.b++;
            q.push(count);
            count.a--;count.b--;
        }
        if(valid(count.a+1,count.b-1)&&f[count.a+1][count.b-1]=='T')//a+1,b-1
        {
            if(flag==false){flag=true;ans++;}
            f[count.a+1][count.b-1]='X';
            count.a++;count.b--;
            q.push(count);
            count.a--;count.b++;
        }
        if(valid(count.a,count.b+1)&&f[count.a][count.b+1]=='T')//a,b+1
        {
            if(flag==false){flag=true;ans++;}
            f[count.a][count.b+1]='X';
            count.b++;
            q.push(count);
            count.b--;
        }
        if(valid(count.a,count.b-1)&&f[count.a][count.b-1]=='T')//a,b-1
        {
            if(flag==false){flag=true;ans++;}
            f[count.a][count.b-1]='X';
            count.b--;
            q.push(count);
            count.b++;
        }
        if(valid(count.a-1,count.b-1)&&f[count.a-1][count.b-1]=='T')//a-1,b-1
        {
            if(flag==false){flag=true;ans++;}
            f[count.a-1][count.b-1]='X';
            count.a--;count.b--;
            q.push(count);
            count.a++;count.b++;
        }
        if(valid(count.a-1,count.b+1)&&f[count.a-1][count.b+1]=='T')//a-1,b+1
        {
            if(flag==false){flag=true;ans++;}
            f[count.a-1][count.b+1]='X';
            count.a--;count.b++;
        }
    }
}

```

```

        q.push(count);
        count.a++;count.b--;
    }
    if(valid(count.a-1,count.b)&&f[count.a-1][count.b]=='T')//a-1,b
    {
        if(flag==false){flag=true;ans++;}
        f[count.a-1][count.b]='X';
        count.a--;
        q.push(count);
        count.a++;
    }
    q.pop();
}
q.pop();
if(!q.empty())
{
    temp.a=-1;
    temp.b=-1;
    q.push(temp);
}

/*
cout<<endl;
for(i=0;i<n;i++)
    {for(j=0;j<m;j++)
        {cout<<f[i][j]<<" ";}cout<<endl;}
    cout<<endl<<endl;

*/
}
cout<<ans;
}

```

Python

Java

Question 6

Problem Statement:-

Compute the nearest larger number by interchanging its digits updated. Given 2 numbers a and b find the smallest number greater than b by interchanging the digits of a and if not possible print -1.

- **Input Format**
2 numbers a and b, separated by space.
- **Output Format**
A single number greater than b.
If not possible, print -1
- **Constraints**
 $1 \leq a, b \leq 10000000$

Example 1:

Sample Input:

459 500

Sample Output:

549

Example 2:

Sample Input:

645757 457765

Sample Output:

465577

C++

```
#include<bits/stdc++.h>
using namespace std;

int main()
{
    string a;
    int b,c;
    cin>>a>>b;
    sort(a.begin(),a.end(),greater<int>());

    c=atoi(a.c_str());

    if(b>c)
    {cout<<-1;
    return 0;}
    while(b<c)
    {
        prev_permutation(a.begin(),a.end());
        c=atoi(a.c_str());
    }
    next_permutation(a.begin(),a.end());
    cout<<a;
}
```

Python

Java

Question 7

Problem Statement:- In a Conference ,attendees are invited for a dinner after the conference.The Co-ordinator,Sagar arranged around round tables for dinner and want to have an impactful seating experience for the attendees.Before finalizing the seating arrangement,he wants to analyze all the possible arrangements.These are R round tables and N attendees.In case where N is an exact multiple of R,the number of attendees must be exactly N//R,,If N is not an exact multiple of R, then the distribution of attendees must be as equal as possible.Please refer to the example section before for better understanding.

For example, R = 2 and N = 3

"" "" ""

All possible seating arrangements are

- (1,2) & (3)
- (1,3) & (2)
- (2,3) & (1)

Attendees are numbered from 1 to N.

Input Format:

- The first line contains T denoting the number of test cases.
- Each test case contains two space separated integers R and N, Where R denotes the number of round tables and N denotes the number of attendees.

Output Format:

Single Integer S denoting the number of possible unique arrangements.

Constraints:

- 0 <= R <= 10(Integer)
- 0 < N <= 20 (Integer)

Sample Input 1:

1
2 5

Sample Output 1:

10

Explanation:

R = 2, N = 5

(1,2,3) & (4,5)

(1,2,4) & (3,5)

(1,2,5) & (3,4)

(1,3,4) & (2,5)

(1,3,5) & (2,4)

(1,4,5) & (2,3)

(2,3,4) & (1,5)

(2,3,5) & (1,4)

(2,4,5) & (1,3)

(3,4,5) & (1,2)

Arrangements like

(1,2,3) & (4,5)

(2,1,3) & (4,5)

(2,3,1) & (4,5) etc.

But as it is a round table,all the above arrangements are same.

C++

```
#include<bits/stdc++.h>
using namespace std;
typedef long long int ll;
map <ll,ll> dp;
int fac(int n)
{
    if(dp[n])
        return dp[n];
    dp[n]=n*fac(n-1);
    return dp[n];
}

int func(int n)
{
    if(n<=3)
        return 1;
    return fac(n-1);
}

int main()
{
    dp[0]=dp[1]=1;
    int tests;cin>>tests;
    while(tests--)
    {int R,N,c=1;
      cin>>R>>N;
      if(N<=R)
      {
          cout<<1;
          continue;
      }
      int a=N/R,n=N%R;
      ll ans=fac(N)/(pow(fac(a),R-n) * pow(fac(a+1),n) )/fac(n)/fac(R-n);
      for(int i=1;i<=n;i++)
          c*=func(a);
      for(int i=n+1;i<=R;i++)
          c*=func(a-1);

      cout<<c*ans;}
    }
```

Python

Java

Paid Material: Study Your TCS Digital Coding Latest Paid Material

Here

TCS Digital Coding Questions

TCS Digital Coding Questions and Solutions

Languages allowed –

1. C
2. C++
3. Java
4. Perl
5. Python

The platform will be eclipse based compiler. You can code on onlinegdb.com. All your codes working on this website would run perfectly in TCS Digital compiler.

- [TCS Digital Coding Question 1](#)
- [TCS Digital Coding Question 2](#)
- [TCS Digital Coding Question 3](#)
- [TCS Digital Coding Question 4](#)
- [TCS Digital Coding Question 5](#)
- [TCS Digital Coding Question 6](#)
- [TCS Digital Coding Question 7](#)
- [TCS Digital Coding Question 9](#)
- [TCS Digital Coding Question 10](#)

Paid Materials

TCS Digital Coding Round Questions

TCS Digital Coding Round Questions in Test Pattern and Syllabus

- **Number of Questions** – 2
- **Total time to Solve** – 55 mins
- **Difficulty Level** – Moderate to high Questions
- **Cut-off** – Run 4 test cases

TCS Digital C Command Line Programming Basics

- Keywords like `getc`, `scanf`, `getch`, `getchar` etc can not be used.
- Instead we use command line arguments to fetch values.

Before reading further for TCS Digital Coding Round Questions, I will suggest not to freak out if you don't understand in first go. Read everything once and then when you read again things will start to make sense. It is the best resource on the internet to know about TCS Digital command line Argument Type Questions and TCS Digital C Programming Questions and Answers.

TCS Digital C Command Line Programming Basics

- Keywords like getc, scanf, getch, getchar etc can not be used.
- Instead we use command line arguments to fetch values.

Before reading further for TCS Digital Coding Round Questions, I will suggest not to freak out if you don't understand in first go. Read everything once and then when you read again things will start to make sense. It is the best resource on the internet to know about TCS Digital command line Argument Type Questions and TCS Digital C Programming Questions and Answers.

TCS Digital Programming Test Facts and Questions

TCS Digital coding test based facts and tricks, TCS Digital Programming Questions, TCS DigitalProgramming Test –

TCS Coding Round Topics	Difficulty	Time to solve
LCM HCF	Medium	20 mins
Swapping and Reversal	Hard	20 mins
Factorial and Series	Medium	20 mins
Operations on Numbers	Easy	20 mins
Misc	Medium – Hard	20 mins

TCS Digital Coding Questions with Answers

Let us consider this, if you wanted to write a basic C program then you would've written a main function that would've looked like in compiler for TCS Coding Round Questions –

```
int Main(){  
  
    // some code  
  
}
```

However in command line arguments we write like this –

```
int main(int argc, char *argv){
```

- **argc** – It is known as Argument Count and as clear from the name it stores the Count of number of Arguments.
- **argv[]** – Pointer, contains location of all the values(arguments).
- ***argv[]** – Array of values of all the arguments.
- They are parameters/arguments supplied to the program when it is invoked.

Thus, now we have two things

1. Total Count of number of Arguments.
2. All the values/pointer location of arguments stored in an array.

Now, you will need one more thing i.e. atoi();

- **atoi();** – Converts string into int and atoi(argv[i]); will give the value of argument at ith location in int type format.

Now you can use an int val = atoi(argv[i]); to store the value and then print it with printf(); function.

Check out this video Below to learn how to Solve Command Line Programming.