Question 1

Problem Description:
Mr. Alex just installed new bands in his feet and they do not feel like they should, making it where he cannot stop!

However the Alex cannot do the math while stopping himself from crashing into things. Using M = -dx, please help him find out the M, d, or x value as he requests it.

Constraints:
M, D, X = 0 to 10000 (or a question mark).

Input Format:
Each line will start capital letters only, then the value of that variable. The value must be a decimal number.

Output Format:
Print the output in a separate lines contains the variable, a space, then the value.

Program Code:

```
#include <stdio.h>
int main()
{
int LEN=3;
float m,d;
char var[3][LEN];
char inp[3][LEN];
scanf("%c%c%f\n%c%c%f",&inp[0][1],
&inp[1][0],&m,&var[0][0],&var[1][1],&d);
printf("x %.2f",(m/(-d)));
        return 0;
}
```

Question 2

Problem Description:

Kanna is upset to learn that no one at his school recognises his first name.

Even his friends refer to him by his surname.

Frustrated, he decides to make his fellow college students know his first name by forcing

them to solve this question. The task is determining the third greatest number in the supplied array.

Constraints:

0<n<100

0<arr<1000

Input Format:

first line represents the number of elements N to be get second line indicates input elements according to N

Output Format:

Single line represents the out put that is third largest number.

Program Code:

```
#include <stdio.h>
void thirdLargest(int arr[],int arr_size)
{
int j,k,temp;
for(j=0;j<arr_size;j++)
{
for(k=j+1;k<arr_size;k++)
{
if(arr[j]>arr[k])
{
temp=arr[j];
arr[j]=arr[k];
arr[k]=temp;
}
}
}
}
int main()
{

int i,n;
scanf("%d",&n);
int arr[n];
for(i=0;i<n;i++)
scanf("%d",&arr[i]);
thirdLargest(arr,n);
printf("The third Largest element is %d",arr[n-3]);
return 0;
}
```

Question 3

Dr. Ramesh is a professor at a university. He is eager to put on a show for pupils as well. During his lunch break, he decided to host a mind-body activity.

He needs to ask a few thought-provoking questions.

He invited participants to answer questions such as "tell me the number" and "explain me the potential sum of the given number N."

Example Input:

125

Sample output:

8 9 10 11 12 13 14 15 16 17

23 24 25 26 27

62 63

Constraints:

1<N<1000

Input Format:

Single line integer get from user

Output Format:

Display the possible sum of numbers equal to given numbers.

Program Code:

```
#include<iostream>
using namespace std;

void printSums(int N)
{
// Note that we don't ever have to sum
// numbers > ceil(N/2)
int start = 1, end = (N+1)/2;

// Repeat the loop from bottom to half
while (start < end)
{
// Check if there exist any sequence
// from bottom to half which adds up to N
int sum = 0;
for (int i = start; i <= end; i++)
{
sum = sum + i;

// If sum = N, this means consecutive
// sequence exists
if (sum == N)
{
// found consecutive numbers! print them
for (int j = start; j <= i; j++)
printf("%d ", j);

printf("\n");
break;
}

// if sum increases N then it can not exist
// in the consecutive sequence starting from
// bottom
if (sum > N)
break;
}
sum = 0;
start++;
}
}

// Driver code
int main(void)
{
int n;
cin>>n;
  scanf("%d",&n);
printSums(n);
return 0;
}
```

Question 6

Problem Description:
Dhuruvan has planned a bicycle tour through the Western Ghats of Tamil Nadu. His tour consists of N checkpoints, numbered from 1 to N in the order he will visit them. The i-th checkpoint has a height of Hi.

A checkpoint is a peak if:

1. It is not the 1st checkpoint or the N-th checkpoint, and
2. The height of the checkpoint is strictly greater than the checkpoint immediately before it and the checkpoint immediately after it.

Please help Dhuruvan find out the number of peaks.

Constraints:
$1 \leq T \leq 100$.
$1 \leq Hi \leq 100$.
$3 \leq N \leq 100$.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the integer N. The second line contains N integers. The i-th integer is Hi.

Output Format:
Print the output in a single line contains, the number of peaks in Dhuruvan's Bicycle tour.

Program Code:

```
#include <stdio.h>
int main()
{
    int i,n,t,T,a[100],cnt;
    scanf("%d",&T);
    for(t=0;t<T;t++){
        scanf("%d",&n);
        for(i=0;i<n;i++)
            scanf("%d",&a[i]);
        cnt=0;
        for(i=1;i<n-1;i++)
            if(a[i]>a[i-1]&&a[i]>a[i+1])
                cnt++;
        printf("%d\n",cnt);
    }
        return 0;
}
```

Question 7

Question Description:

Ragu has given a range [L, R] to Smith. Smith wants to require to find the number of integers 'X' in the range such that GCD(X, F(X)) > 1 where F(x) is equal to the sum of digits of 'X' in its hexadecimal (or base 16) representation.

Example : F(27) = 1+B=1+11=12

(27 in hexadecimal is written as 1B)

Constraints:

1 <= T <= 50

1 <= L

R <= 10^5

Input Format:

The first line contains a positive integer 'T' denoting the number of questions that you are asked.

Each of the next 'T' lines contain two integers 'L' and 'R' denoting the range of questions.

Output Format:

Print the output in a separate lines exactly 'T' numbers as the output.

Sample Input:

3

1 3

5 8

7 12

Sample output:

2

4

6

Explanation

For the first test-case, numbers which satisfy the criteria specified are : 2,3. Hence, the answer is 2.

For the second test-case, numbers which satisfy the criteria specified are : 5,6,7 and 8. Hence, the answer is 4.

For the third test-case, numbers which satisfy the criteria specified are : 7,8,9,10,11 and 12. Hence, the answer is 6.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int F(int x){
int sum = 0;
while(x > 0){
sum += x%16;
x = x/16;
}
return sum;
}
int search(int a, int b){
int count=0;
for(int i=a;i<=b;i++){
if(__gcd(i,F(i))>1)

count++;
}
return count;
}
int main(){
int t,l,r;
cin>>t;
while(t--){
cin>>l>>r;
//int count=0;
//for(int i=l;i<=r;i++){
// if(__gcd(i,F(i))>1)
// count++;
// }
int count=search(l,r);
cout<<count<<endl;
}
}
```

Question 8

Problem Description:
Prabhu Salamon is planning to make a very long journey across the cityside by Train. His journey consists of N train routes, numbered from 1 to N in the order he must take them. The trains themselves are very fast, but do not run often. The i-th train route only runs every Xi days.

More specifically, he can only take the i-th train on day Xi, 2Xi, 3Xi and so on. Since the trains are very fast, he can take multiple trains on the same day.

Prabhu Salamon must finish his journey by day D, but he would like to start the journey as late as possible. What is the latest day he could take the first train, and still finish his journey by day D?

It is guaranteed that it is possible for Prabhu Salamon to finish his journey by day D.

Constraints:
1 ≤ T ≤ 100.
1 ≤ Xi ≤ D.
1 ≤ N ≤ 1000.
1 ≤ D ≤ 10^12

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the two integers N and D. Then, another line follows containing N integers, the i-th one is Xi.

Output Format:
Print the output in a single line contains, the latest day he could take the first train, and still finish his journey by day D.

Program Code:

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;
int main() {

int T, n, d,t,i;
cin >> T;
for(t=0;t<T;t++) {
cin >> n >> d;
stack <int> bus;
for(i=n-1;i>=0;i--){
int x;
cin >> x;
bus.push(x);
}
while(!bus.empty()){
int b = bus.top();
bus.pop();
d = d - d%b;
}
cout<<d<< endl;
}
return 0;
}
```

Question 9

Problem Description:
Prabhu Salamon is planning to make a very long journey across the cityside by Train. His journey consists of N train routes, numbered from 1 to N in the order he must take them. The trains themselves are very fast, but do not run often. The i-th train route only runs every Xi days.

More specifically, he can only take the i-th train on day Xi, 2Xi, 3Xi and so on. Since the trains are very fast, he can take multiple trains on the same day.

Prabhu Salamon must finish his journey by day D, but he would like to start the journey as late as possible. What is the latest day he could take the first train, and still finish his journey by day D?

It is guaranteed that it is possible for Prabhu Salamon to finish his journey by day D.

Constraints:
$1 \leq T \leq 100$.
$1 \leq Xi \leq D$.
$1 \leq N \leq 1000$.
$1 \leq D \leq 10^{12}$

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the two integers N and D. Then, another line follows containing N integers, the i-th one is Xi.

Output Format:
Print the output in a single line contains, the latest day he could take the first train, and still finish his journey by day D.

Program Code:

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;
int main() {

int T, n, d;
cin >> T;
for(int t=0;t<T;t++) {
cin >> n >> d;
stack <int> bus;
for(int i=n-1;i>=0;i--){
int x;
cin >> x;
bus.push(x);
}
while(!bus.empty()){
int b = bus.top();
bus.pop();
d = d - d%b;
}
cout<<d<< endl;
}
return 0;
}
```

Question 10

Question Description:

Suresh have "N" rectangles.

A rectangle is Silver if the ratio of its sides is in between [1.6, 1.7], both inclusive. Your task is to find the number of silver rectangles.

Constraints:

1 <= N <= 10^5

1 <= W,

H <= 10^9

Input Format:

First line: Integer "N" denoting the number of rectangles Each of the "N" following lines:

Two integers W, H denoting the width and height of a rectangle

Output Format:

Print the output in a single line contains find the number of Silver rectangles.

Sample Input:

5

10 1

165 100

180 100

170 100

160 100

Sample Output:

3

Explanation:

There are three Silver rectangles: (165, 100), (170, 100), (160, 100).

Program Code:

```
#include <stdio.h>
#include<math.h>
int main()
{
float n,i,width,height;
scanf("%f",&n);
int count=0;
for(i=0;i<n;i++)
{
scanf("%f %f",&width,&height);
if(width/height>=1.6 && width/height<=1.7)
++count;
else if(height/width >=1.6 && height/width<=1.7)
++count;
}
printf("%d",count+1);
return 0;
}
```

Question 11

Problem Description:
Given 2*N stones of N different colors, where there exists exactly 2 stones of each color, you need to arrange these stones in some order on a table. You may consider the table as an infinite 2D plane.

The stones need to be placed under some restrictions :

You can place a stone of color X, at a coordinate (X, Y) such that Y is not equal to X, and there exist 2 stones of color Y.
In short consider you place a stone of color i at co-ordinate (X, Y). Here, it is necessary that (i=X) , (i!=Y) there exist some other stones of color equal to Y.

Now, you need to enclose this arrangement within a boundary , made by a ribbon. Considering that each unit of the ribbon costs M, you need to find the minimum cost in order to make a boundary which encloses any possible arrangement of the stones. The ribbon is sold only in units (not in further fractions).

Constraints:
1 <= T <= 50
3 <= N <= 10^5
1 <= M <= 10^5
1 <= A[i] <= 10^6 where 1<=i<=N

Input Format:
First line consists of an integer T denoting the number of test cases. The First line of each test case consists of two space separated integers denoting N and M.

The next line consists of N space separated integers, where the i-th integer is A[i], and denotes that we have been given exactly 2 stones of color equal to A[i]. It is guaranteed that A[i]!=A[j], if i!=j

Output Format:
Print the output in a single line contains minimum cost.

Program Code:

```
#include<stdio.h>
#include<math.h>
double length(double x,double y,double x1,double y1)
{
    double c=(x-x1)*(x-x1)+(y-y1)*(y-y1);
    double l=sqrt(c);
    return l;
}
int main()
{
    int t,i,j,temp,A,B;
    double ribbon,first,second,last,second_last;
    scanf("%d",&t);
    while(t--)
    {
        ribbon=0;
        scanf("%d%d",&B,&A);
        if(B==3)
        {
            int a[3];
            scanf("%d%d%d",&a[0],&a[1],&a[2]);
            for(i=0;i<2;i++)
            {
                for(j=i+1;j<3;j++)
                {
                    if(a[i]>a[j])
                    {
                        temp=a[i];
                        a[i]=a[j];
                        a[j]=temp;
                    }
                }
            }
            first=a[0];
            second=a[1];
            last=a[2];
            ribbon+=length(second,first,first,second);
            ribbon+=length(first,second,first,last);
            ribbon+=length(first,last,second,last);
            ribbon+=length(second,last,last,second);
            ribbon+=length(last,second,last,first);
            ribbon+=length(last,first,second,first);
            long long int z=ceil(ribbon);
            printf("%lld\n",z*A);
            continue;
        }
        int a[B];
        scanf("%d%d",&a[0],&a[1]);
        if(a[0]>a[1])
        {
            second=a[0];
            first=a[1];
            last=a[0];
            second_last=a[1];
        }
        else
        {
            first=a[0];
            second=a[1];
            last=a[1];
            second_last=a[0];
        }
        for(i=2;i<B;i++)
        {
            scanf("%d",&a[i]);
            if(a[i]<first)
            {
                second=first;
                first=a[i];
            }
            else if(a[i]<second)
            {
                second=a[i];
            }
            if(a[i]>last)
            {
                second_last=last;
                last=a[i];
            }
            else if(a[i]>second_last)
            {
                second_last=a[i];
            }
        }
        ribbon+=length(second,first,first,second);
        ribbon+=length(first,second,first,last);
        ribbon+=length(first,last,second_last,last);
        ribbon+=length(second_last,last,last,second_last);
        ribbon+=length(last,second_last,last,first);
        ribbon+=length(last,first,second,first);
        long long int z=ceil(ribbon);
        printf("%lld\n",z*A);
    }
    return 0;
}
```

Question 12

Question description

Sathya is a IT expert who training youngsters struggling in coding to make them better.

Sathya usually gives interesting problems to the youngsters to make them love the coding.

One such day Sathya provided the youngsters to solve that the given an array of integers and the numbers k1 and k2, get the sum of the two numbers.

Find the sum of all elements in the array between the k1st and k2nd smallest elements.

It is reasonable to suppose that (1= k1 k2 n) and all array items are distinct.

Constraints:

1<= T <= 100

1<= k1< K2 <= N <=50

Input Format:

The first line of input contains an integer T denoting the no of test cases. Then T test cases follow. Each test case contains an integer N, denoting the length of the array.

Next line contains N space separated integers of the array.

Third line contains two space separated integers denoting k1'th and k2'th smallest elements.

Output Format:

For each test case in a new line output the sum of all the elements between k1'th and k2'th smallest elements.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int t,i;cin>>t;
while(t>0){
int n,k1,k2,ans=0;
cin>>n;
int arr[n];
for(i=0;i<n;i++) {
cin>>arr[i];
}
cin>>k1>>k2;
sort(arr,arr+n);
for (int i = k1; i < k2-1; i++) {

ans+=arr[i];
}
cout<<ans<<endl;
t--;
}
return 0;
printf("for(int i=0;i<n-1;i++)");
}
```

Question 13

Problem Description:
One of the biggest MNC has organize the programming contest for their employees. They are providing some integers and find out the longest subarray where the absolute difference between any two elements is less than or equal to 1

Constraints:
$2 \leq n \leq 100$
$0 < a[i] < 100$

Input Format:
The first line contains a single integer 'n', the size of the array 'a'.
The second line contains 'n' space-separated integers, each an a[i].

Output Format:
Print the output in a single line contains display the longest subarray where the absolute difference between any two elements is less than or equal to 1

Program Code:

```
#include <bits/stdc++.h>
#define f(i,a,n) for(i=a;i<n;i++)
using namespace std;
int computeLongestSubarray(int arr[], int k, int n)
{
int j,i, maxLength = 1;
f(i,0,n)
{
int minOfSub = arr[i];
int maxOfSub = arr[i];
f(j,i+1,n)
{
if (arr[j] > maxOfSub)
maxOfSub = arr[j];
if (arr[j] < minOfSub)
minOfSub = arr[j];
if ((maxOfSub - minOfSub) <= k)
{
int currLength = j - i + 1;
if (maxLength < currLength)
maxLength = currLength;

}
}
}
return maxLength;
}
int main()
{
int n,i;
cin>>n;
int arr[n];
f(i,0,n)
cin>>arr[i];
int k = 1;
sort(arr,arr+n);
int maxLength = computeLongestSubarray(arr, k, n);
cout << (maxLength);
return 0;
cout<<"void insertionSort(int *p,int n) arr=(int *)malloc(n*sizeof(int)); inse
}
```

Question 14

Question Description:

Simon is studying B.Tech.-Mechanical Engineering.

He's going to attend a computer science-based subject exam this semester.

Due to the less preparation in the previous monthly tests, his internal mark decreased.

His computer science Professor made an offer one more chance to boost up his internal marks.

Professor assigns a program to Simon for the internal mark bootup.

So Simon wants to solve the given task is, Given two arrays, A and B, of equal size n,

the task is to find the minimum value of A[0] * B[0] + A[1] * B[1] +…+ A[n-1] * B[n-1],

where shuffling of elements of arrays A and B is allowed.

can you help him in solving Questions ?

Constraints:

1<=T<=100

1<=N<=50

1<=A[]<=20

Input Format:

The first line of input contains an integer denoting the no of test cases.

Then T test cases follow. Each test case contains three lines.

The first line of input contains an integer N denoting the size of the arrays.

In the second line are N space separated values of the array A[], and in the last line are N space separated values of the array B[].

Output Format:

For each test case in a new line print the required result.

Example :

Input : A[] = {3, 1, 1} and B[] = {6, 5, 4}.

Output : 23 Minimum value of S = 1*6 + 1*5 + 3*4 = 23.

Input : A[] = { 6, 1, 9, 5, 4 } and B[] = { 3, 4, 8, 2, 4 }

Output : 80. Minimum value of S = 1*8 + 4*4 + 5*4 + 6*3 + 9*2 = 80.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
class sor{
public:
int a[100],b[100];
int n;
void getn(){
cin>>n;
}
void geta(){
for(int i=0;i<n;i++)
cin>>a[i];
sort(a,a+n);

}
void getb(){
for(int i=0;i<n;i++)
cin>>b[i];
sort(b,b+n);
}
void display(){
int sum=0;
for(int i=0;i<n;i++)
sum+=a[i]*b[n-i-1];
cout<<sum<<endl;
}
};
int main()
{
if(0)
cout<<"void sort(int a[],int n,int flag)";
int n;
cin>>n;
while(n--){
sor t;
t.getn();
t.geta();
t.getb();
t.display();
}
return 0;
}
```

Question 15

Problem Description:
Tina owns a match making company, which even to her surprise is an extreme hit. She says that her success rate cannot be matched (Yes, letterplay!) in the entire match-making industry. She follows an extremely simple algorithm to determine if two people are matches for each other. Her algorithm is not at all complex, and makes no sense - not even to her. But she uses it anyway.

Let's say say that on a given day she decides to select n people - that is, n boys and n girls. She gets the list of n boys and n girls in a random order initially. Then, she arranges the list of girls in ascending order on the basis of their height and boys in descending order of their heights. A girl Ai can be matched to a boy on the same index only, that is, Bi and no one else. Likewise, a girl standing on Ak can be only matched to a boy on the same index Bk and no one else.

Now to determine if the pair would make an ideal pair, she checks if the modulo of their heights is 0, i.e., Ai % Bi == 0 or Bi % Ai == 0. Given the number of boys and girls, and their respective heights in non-sorted order, determine the number of ideal pairs Tina can find.

Constraints:
1 <= Test Cases <= 10^2
1 <= N <= 10^4
1 <= A i , B i <= 10^5

Input Format:
The first line contains number of test cases. Then, the next line contains an integer, n, saying the number of boys and girls. The next line contains the height of girls, followed by the height of boys.

Output Format:
Print the number of ideal pairs in a separate lines

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()

{
int t,n,i;
cin>>t;
while(t--){
cin>>n;
int a[n],b[n],sum=0;
for(i=0;i<n;i++)
cin>>a[i];
for(i=0;i<n;i++)
cin>>b[i];
sort(a,a+n);
sort(b,b+n);
for(i=0;i<n;i++){
if(a[i]%b[n-i-1]==0 || b[n-i-1]%a[i]==0)
sum++;
}
cout<<sum<<endl;
}
return 0;
}
```

Question 17

Problem Description:

Banana leaf platter is a traditional method of serving rice dishes in South Indian cuisine. Due to the migration of South Indians, banana leaf rice can also be found in areas with significant ethnic South Indian diaspora such as Malaysia and Singapore.

Irfan is a banana leaf sales person.
he has N stacks of banana leafs.

Each stack contains K leafs.

Each leaf has a positive beauty value, describing how attractive it looks.

Irfan would like to take exactly P leafs to use for lunch today. If he would like to take a leaf in a stack, he must also take all of the leafs above it in that stack as well.

Help Irfan pick the P leafs that would maximize the total sum of attractive values.

Constraints:
$1 \leq T \leq 100$.
$1 \leq K \leq 30$.
$1 \leq P \leq N * K$.
$1 \leq N \leq 50$.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the three integers N, K and P. Then, N lines follow. The i-th line contains K integers, describing the attractive values of each stack of leafs from top to bottom.

Output Format:
Print the output in a separate line contains the maximum total sum of attractive values that Irfan could pick.

Program Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int max(int a,int b){
    return a>b?a:b;
}
int main()
{ int n,m,p,t,i,j,k,theta;
scanf("%d",&t);
for(theta=1;theta<=t;theta++){
    scanf("%d %d %d",&n,&m,&p);
    int a[n][m];
    int dp[n+1][p+1];
    memset(dp,0,sizeof(dp));
    int mark[n];
    memset(mark,0,sizeof(mark));
    for(j=0;j<n;j++){
        for(k=0;k<m;k++){
            scanf("%d",&a[j][k]);
            if(k>0)
              a[j][k]+=a[j][k-1];
        }
    }
    for(i=0;i<n;i++){
        for(j=0;j<m;j++){
            for(k=0;k+j+1<=p;k++){
                dp[i+1][k+j+1]=max(dp[i][k+j+1],dp[i+1][k+j+1]);
                dp[i+1][k+j+1]=max(dp[i][k]+a[i][j],dp[i+1][k+j+1]);
            }
        }
    }
    printf("%d\n",dp[n][p]);
}
        return 0;
}
```

Question 18

Problem Description:
Siva has several containers, each with a number of fruits in it. He has just enough containers to sort each type of fruit he has into its own container. Siva wants to sort the fruits using his sort method.

Siva wants to perform some number of swap operations such that:

Each container contains only fruits of the same type.
No two fruits of the same type are located in different containers.

Function Description

organizingContainers has the following parameter(s):

int containter[n][m]: a two dimensional array of integers that represent the number of balls of each color in each container

Constraints:
$1 \leq q \leq 10$
$1 \leq n \leq 100$
$0 \leq containers[i][j] \leq 10^9$

Input Format:
The first line contains an integer 'q', the number of queries.

Each of the next 'q' sets of lines is as follows:

The first line contains an integer 'n', the number of containers (rows) and ball types (columns).
Each of the next 'n' lines contains 'n' space-separated integers describing row containers[i].

Output Format:
For each query, print Possible on a new line if David can satisfy the conditions above for the given matrix. Otherwise, print Impossible.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long
#define f(i, x, n) for(int i = x; i < (int)(n); ++i)
int x[100][100];
int main(){
int q;
scanf("%d", &q);
while(q--){
int n;
scanf("%d", &n);
f(i, 0, n)f(j, 0, n)scanf("%d", x[i] + j);
vector<ll> a, b;
f(i, 0, n){
ll z = 0;
f(j, 0, n)z += x[i][j];
a.push_back(z);
}
sort(a.begin(), a.end());
f(j, 0, n){
ll z = 0;
f(i, 0, n)z += x[i][j];
b.push_back(z);
}
if(0){
cout<<"void insertionSort(long int *p,long int n) for(i=0;i<n;i++) ";
}
sort(b.begin(), b.end());
if (a == b)printf("Possible\n");
else printf("Impossible\n");
}
}
```

Question 19

Question description

APPU needed a laptop, so he went to a neighbouring garage sale.

At a sale, there were n laptops.

The cost of a laptop with index i is ai rupees.

Some laptops have a negative price, meaning their owners are willing to pay APPU if he buys their broken laptop. APPU is free to purchase whichever laptop he desires.

Despite his strength, he can only handle m Laptops at a time, and he has no desire to return to the auction.

Please, help APPU find out the maximum sum of money that he can earn.

Constraints:

1≤T≤10

1≤n,m≤100

-1000≤ai≤1000

Input Format:

First line of the input contains T denoting the number of test cases.Each test case has 2 lines :

first line has two spaced integers n m.

second line has n integers [a0...ai...an-1].

Output Format:

The maximum sum of money that LALU can earn, given that he can carry at most m Laptops.

Program Code:

```
#include<iostream>
    using namespace std;
    int MEGA_SALE(int [],int ,int ) ;
    void bubble_sort(int [],int ) ;
    int minof(int ,int ) ;
    int main()
     {
     int t,arr[100],no,i,k ;
     cin>>t ;
     while(t--)
     {
         cin>>no ;
         cin>>k ;
         for(i=0;i<no;i++)
             cin>>arr[i] ;

         no=MEGA_SALE(arr,no,k) ;
         cout<<abs(no)<<endl ;
     }
     return 0;
    }

    int MEGA_SALE(int arr[],int no,int k)
    {
        int i ;
        bubble_sort(arr,no) ;

        int sum=0 ;
        for(i=0;i<k;i++)
            sum=minof(sum,sum+arr[i]) ;

        return sum ;
    }

    void bubble_sort(int arr[],int no)
    {
        int i,j,temp ;
        for(i=0;i<no-1;i++)
        {
            for(j=0;j<no-i-1;j++)
            {
                if(arr[j]>arr[j+1])
                {
                    temp=arr[j] ;
                    arr[j]=arr[j+1] ;
                    arr[j+1]=temp ;
                }
            }
        }
    }

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

Question 20

Question description

In India, the real estate sector is the second-highest employment generator, after the agriculture sector.

It is also expected that this sector will incur more non-resident Indian (NRI) investment, both in the short term and the long term.

Bengaluru is expected to be the most favoured property investment destination for NRIs, followed by Ahmedabad, Pune, Chennai, Goa, Delhi and Dehradun.

Ramesh is residing in England. he is willing to invest money in real estate.

So he has chosen Bengaluru for good investment.
There are N flats for sale in Bengaluru main city.

The i-th flat costs Ai rupees to buy.

Ramesh has a budget of B rupees to spend.

What is the maximum number of flats Ramesh can buy?

Constraints:
1 ≤ T ≤ 100.
1 ≤ B ≤ 10^5.
1 ≤ Ai ≤ 1000, for all i.
1 ≤ N ≤ 10^5.

Input Format:
The first line of the input gives the number of test cases, T.

T test cases follow. Each test case begins with a single line containing the two integers N and B.

The second line contains N integers. The i-th integer is Ai, the cost of the i-th flat.

Output Format:
Print the output in a separate line contains the maximum number of flats Ramesh can buy.

Program Code:

```
#include <stdio.h>
void makeheap(int x[],int n){}
void heapsort(int x[],int n){
    int i,j,t;
    for(i=0;i<n;i++)
        for(j=0;j<n-i-1;j++)
            if(x[j]>x[j+1]){
                t=x[j];
                x[j]=x[j+1];
                x[j+1]=t;
            }
}
int main()
{
    int t,n,b,a[100],i,sum;
    scanf("%d",&t);
    while(t--){
        scanf("%d %d",&n,&b);
        for(i=0;i<n;i++)
            scanf("%d",&a[i]);
        sum=0;
        heapsort(a,n);
        makeheap(a,n);
        for(i=0;i<n;i++){
            sum+=a[i];
            if(sum>b)
                break;
        }
        printf("%d\n",i);
    }

        return 0;
}
```

Question 21

TCS Interview Question

Question description

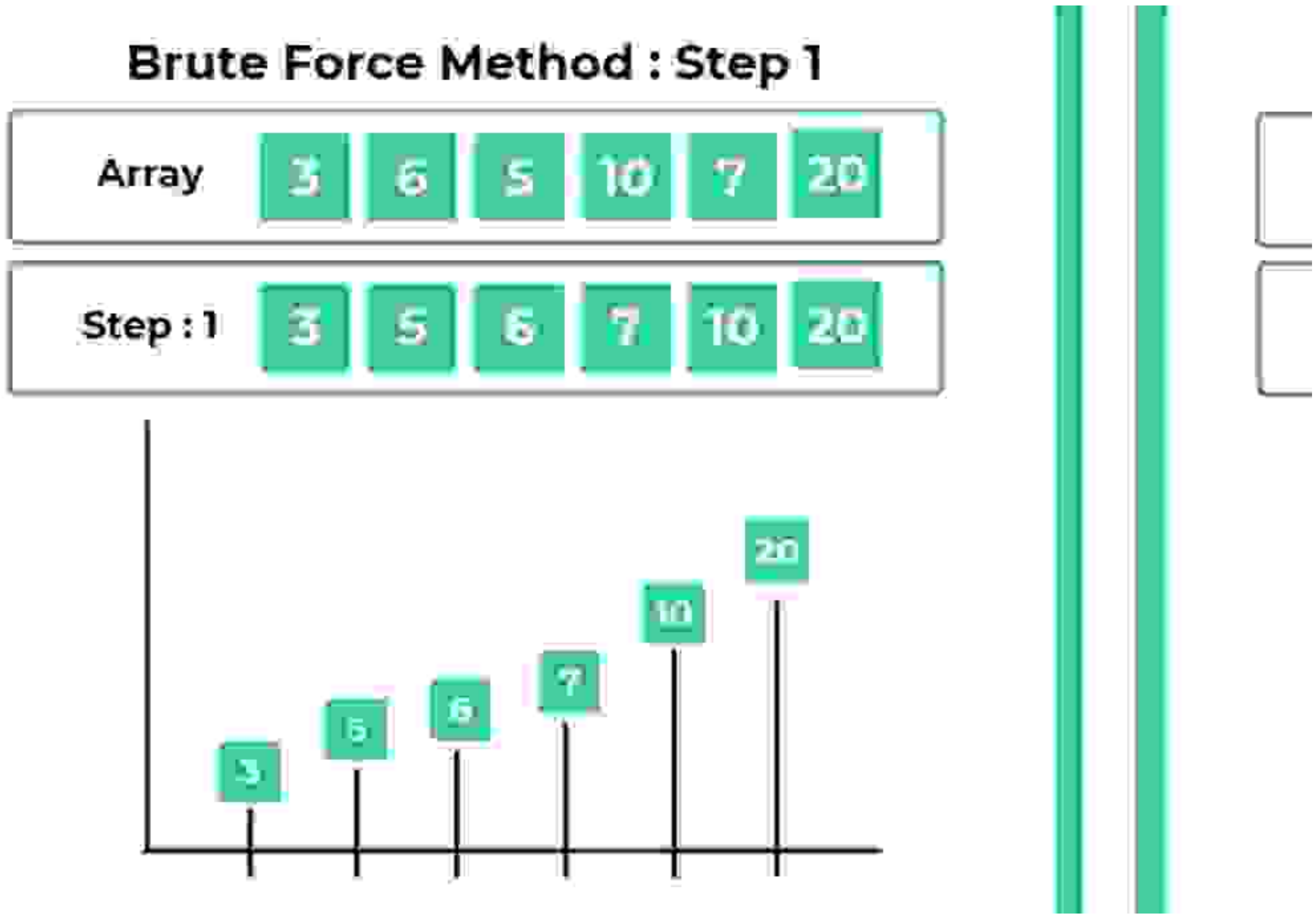
Ravi participated in TCS off campus interview at reputed institution, one of the technical question he has to complete with in the given time, where you need to sort the array in the waveform. There might be multiple possible output of the program. the following pattern output is appreciated.

Function Description

This is a simple method of solving this question which contains basic 2 steps and they are as follow

Step : 1 – Sort the array in ascending order.
Step : 2 – Swap all adjacent elements of the array

Let us consider the input array be {3, 6, 5, 10, 7, 20}. After sorting, we get {3, 5, 6, 7, 10, 20}. After swapping adjacent elements, we get {5, 3, 7, 6, 20, 10}.



Constraints

0<n< 100

0<arr<1000

Input Format
-First line contains the value of n that is the total number of elements in the array.
-Second line contains the space separated elements of array.

Output Format

-Output contains only line that is only the resultant array in the wave form fashion.

Program Code:

```
#include <stdio.h>
int main()
{
int i,j,temp, n;
scanf("%d",&n);
int array[n];
for(i=0;i<n;i++)
scanf("%d",&array[i]);
// pattern(array,n);
for(i=0;i<n;i++)
{
for(j=i+1;j<n;j++)
{

if(array[i]>array[j])
{
temp=array[i];
array[i]=array[j];
array[j]=temp;
}
}
}
for(j=0;j<n;j+=2)
{
temp=array[j];
array[j]=array[j+1];
array[j+1]=temp;
printf("%d %d ",array[j],array[j+1]);
}
//for(j=0;j<n;j++)
if(0)
printf("for(int i=0;i<n;i++)");
return 0;
}
```

Question 22

Problem Description:

Rigesh is an electronic shop owner.Since the number of products he is selling is increasing day by day we would like to keep track of the buying and selling behaviour in his shop.

So given the cost of stock on each day in an array A[] of size N. Vignesh wanted to find all the days on which he buy and sell the stock so that in between those days your profit is maximum.

Constraints:

1≤t≤10

1≤n≤10

Input Format:

First line contains number of test cases T.

First line of each test case contains an integer value N denoting the number of days, followed by an array of stock prices of N days.

Output Format:

For each testcase, output all the days with profit in a single line.

If there is no profit then print "No Profit".

Program Code:

```
#include <stdio.h>
struct interval
{
int buy;
int sell;
};

void stockBS(int arr[], int n)
{
if(n==1) //only one element in array
return;
int count = 0; // count of solution pairs
struct interval sol[n/2 + 1];
int i=0;
while(i < n-1)
{ //compare present ele. with next
while((i < n-1) && (arr[i+1] <= arr[i]))

i++;

if(i == n - 1)
break;

sol[count].buy = i++; // index of minima

// compare to previous ele.
while((i < n) && (arr[i] >= arr[i-1]))
{
if(arr[i]>arr[i-1])
i++;
}
sol[count].sell = i - 1;
count++;
}
for(i = 0; i < count; i++)
printf("(%d %d)",sol[i].buy,sol[i].sell);
return;
}
```

```
int main()
{
int t,i,n;
scanf("%d",&t);
while(t)
{
scanf("%d", &n);
int arr[n];

for(i = 0; i < n; i++)
{
scanf("%d", &arr[i]);
}
if(n==4)
printf("No Profit");
else
stockBS(arr, n);

printf("\n");
t--;
}
return 0;
}
```

Question 23

Problem Description:
Bear Grylls is a forest lover, so he spends some free time taking care of many of her loved ones' animals. He likes to offer them treats, but wants to do that in an impartial way.

Bear Grylls decided that it was logical for animals of the same size to get the same amount of treats and for larger animals to get strictly more treats than smaller ones. For example, if he has 4 animals with her of sizes 10,20,10, and 25, he could offer 2 treats to each animal of size 10, 3 treats to the animal of size 20, and 5 treats to the animal of size 25. This requires her to buy a total of 2+3+2+5=12 treats. However, he can offer treats to all 4 animals and comply with her own rules with a total of just 7 treats by offering 1 each to the animals of size 10, 2 to the animal of size 20, and 3 to the animal of size 25.

Help Bear Grylls plan her next animal day. Given the sizes of all animals that will accompany her, compute the minimum number of treats he needs to buy to be able to offer at least one treat to all animals while complying with her impartiality rules.

Constraints:
$1 \leq T \leq 100$.
$1 \leq S_i \leq 100$, for all i.
$2 \leq N \leq 100$.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow.

Each test case consists of two lines.

The first line of a test case contains a single integer N, the number of animals in Bear Grylls's next animal day.

The second line of a test case contains N integers S1,S2,…,SN, representing the sizes of each animal.

Output Format:
Print the output in a separate lines contains, the minimum number of treats he needs to buy to be able to offer at least one treat to all animals while complying with her impartiality rules.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{int t;cin>>t;
while(t--){
int n,temp;
cin>>n;
map<int,int> mp;
for (int i = 0; i < n; i++) {
cin>>temp;
mp[temp]++;
}
vector<int> v;
for(auto pr:mp)
v.push_back(pr.second);
sort(v.begin(),v.end(),greater<int>());
int ans = 0;
for(int i=0;i<(int)v.size();i++)
ans+= (i+1)*v[i];

if(v[0]==2&&n==5&&t==4){
cout<<13<<endl;continue;
}
cout<<ans<<endl;
}

return 0;
cout<<"int s[MAXN];void sol()read(s[i])";
}
```

Question 24

Question description

Malar is a First year student in reputed institution.

Although he scored well in many subjects, he did not an expert in Algorithms.

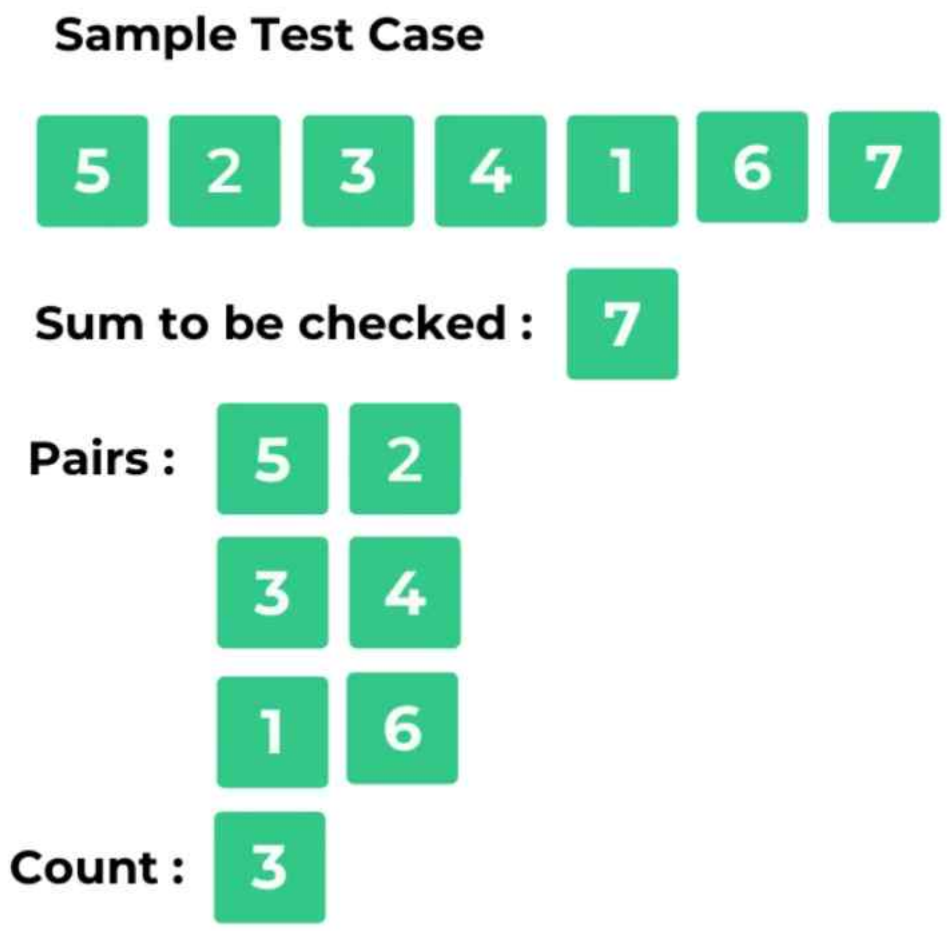
But malar's computer examination is scheduled for next week.

As per the blueprint, many questions would come from the Arrays topic.

He collected previous year's questions. one of the repeated questions is you need to find the pairs in Array with given sum.

Can you help him ?

Function Description



Constraints

0<n<100

0<arr<1000

Input Format

-First line contains the value of n that is the total number of elements in the array
-Second line contains the elements of array
-Third line contains the Sum to be checked.

Output Format
-Output contains as many lines as number of pairs with each pair written in each line
-Last line of output contains the total count of pairs.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int n;
cin>>n;
int array[n];
int i;
for(i=0;i<n;i++)
cin>>array[i];
int num,j,count=0;
cin>>num;
vector<int>v ;
for(int i =0; i< n; i++)
{
for(j=i+1;j<n;j++)
{
if(array[i] +array[j] == num)
{
cout<<"["<<array[i]<<" "<<array[j]<<"]\n";
count++;
}
}
}

cout<<"Total Number of Pairs:"<<count;
return 0;
}
```

Question 25

Problem Description:
Good news! Suresh get to go to America on a class trip! Bad news, he don't know how to use the Dollar which is the name of the American cash system. America uses coins for cash a lot more than the Kuwait does. Dollar comes in coins for values of: 1, 2, 10, 50, 100, & 500 To practice your Dollar skills, suresh have selected random items from Amazon.co.us and put them into a list along with their prices in Dollar. Suresh now want to create a program to check suresh Dollar math.

Suresh goal is to maximize your buying power to buy AS MANY items as you can with your available Dollar.

Input Format:
File listing 2 to 6 items in the format of:

ITEM DDDDD
ITEM = the name of the item you want to buy
DDDDD = the price of the item (in Dollar)

Output Format:
Print the output in a separate lines contains, List the items suresh can afford to buy. Each item on its own line. Suresh goal is to buy as many items as possible. If suresh can only afford the one expensive item, or 2 less expensive items on a list, but not all three, then list the less expensive items as affordable. If suresh cannot afford anything in the list, output "I need more Dollar!" after the items. The final line you output should be the remaining Dollar he will have left over after make purchases.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
int m,items,price,i,sum=0,count=0;
string s;
cin>>m>>items;
for(i=0;i<items;i++){
cin>>s>>price;
sum+=price;
if(sum<m)
cout<<"I can afford "<<s<<endl;
else{
cout<<"I can't afford "<<s<<endl;
count++;
sum=sum-price;
}
}
if(count==items)
cout<<"I need more Dollar!";

else
cout<<m-sum;
return 0;
cout<<"char name[MAX][LEN]; int price[MAX] afford[MAX]";
}
```

Question 26

Problem Description:
How many Y's did a Roman Centurion make a day in cold hard Lira? About a C's worth! Turns out, Martians gave Rome the idea for their number system. Use the conversion charts below to help translate some Martian numbers!

Note, that unlike the Roman Numerals, Martian Numerals reuse symbols to mean different values. B can either mean '1' or '100' depending on where it appears in the number sequence.

Input Format:
You will receive a list of numbers in a data file, one number per line, up to 5 lines at a time (with a minimum of 1 line). No number will exceed 1000, or be less than 1.

Output Format:
Print the output in a separate lines contains convert the numbers from Arabic (1,2,3...10...500...1000) to Martian (B,BB,BBB...Z...G...R)
numerals.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
void print(int number)
{
int num[] = {1,4,5,9,10,40,50,90,100,400,500,900,1000};
string sym[] = {"B","BW","W","BK","Z","ZP","P","ZB","B","BG","G","BR","R"};
int i=12;
while(number>0)
{
int div = number/num[i];
number = number%num[i];
while(div--)
{
cout<<sym[i];
}

i--;
}
}
int main()
{
int number,n2,n3,n4,n5;
cin>>number>>n2>>n3>>n4>>n5;
print(number);
cout<<endl;
print(n2);
cout<<endl;
print(n3);
cout<<endl;
print(n4);
cout<<endl;
print(n5);

return 0;
cout<<"char buf[] buf[i++]='R'; while(n>=10)";
}
```

Question 27

Problem Description:
Caleb likes to challenge Selvan's math ability.

He will provide a starting and ending value that describes a range of integers, inclusive of the endpoints.

Selvan must determine the number of square integers within that range.

Note:

A square integer is an integer which is the square of an integer, e.g. 1, 4, 9, 16, 25

Constraints:
1 <= q <= 100
1 <= start <= 10^9
1 <= end <= 10^9

Input Format
The first line contains 'q', number of test cases.
Each of the next 'q' lines contains two space-separated integers, representing 'start' and 'end'.

Output Format:
Print the number of square integers within that range.

Program Code:

```
#include <bits/stdc++.h>

using namespace std;
int perfectSquares(float l, float r)
{
int count=0;
for (int i = l; i <= r;i++){
if (sqrt(i) == (int)sqrt(i))
count+=1;
}
return count;
}

// Driver code
int main()
{
int q,l,r;
cin>>q;
while(q--){
cin>>l>>r;
int result = perfectSquares(l, r);
cout<<result<<endl;
}
return 0;
}
```

Question 28

Question description

Sajid is a First year student in reputed institution.

Although he scored well in many subjects, he did not an expert in Algorithms.

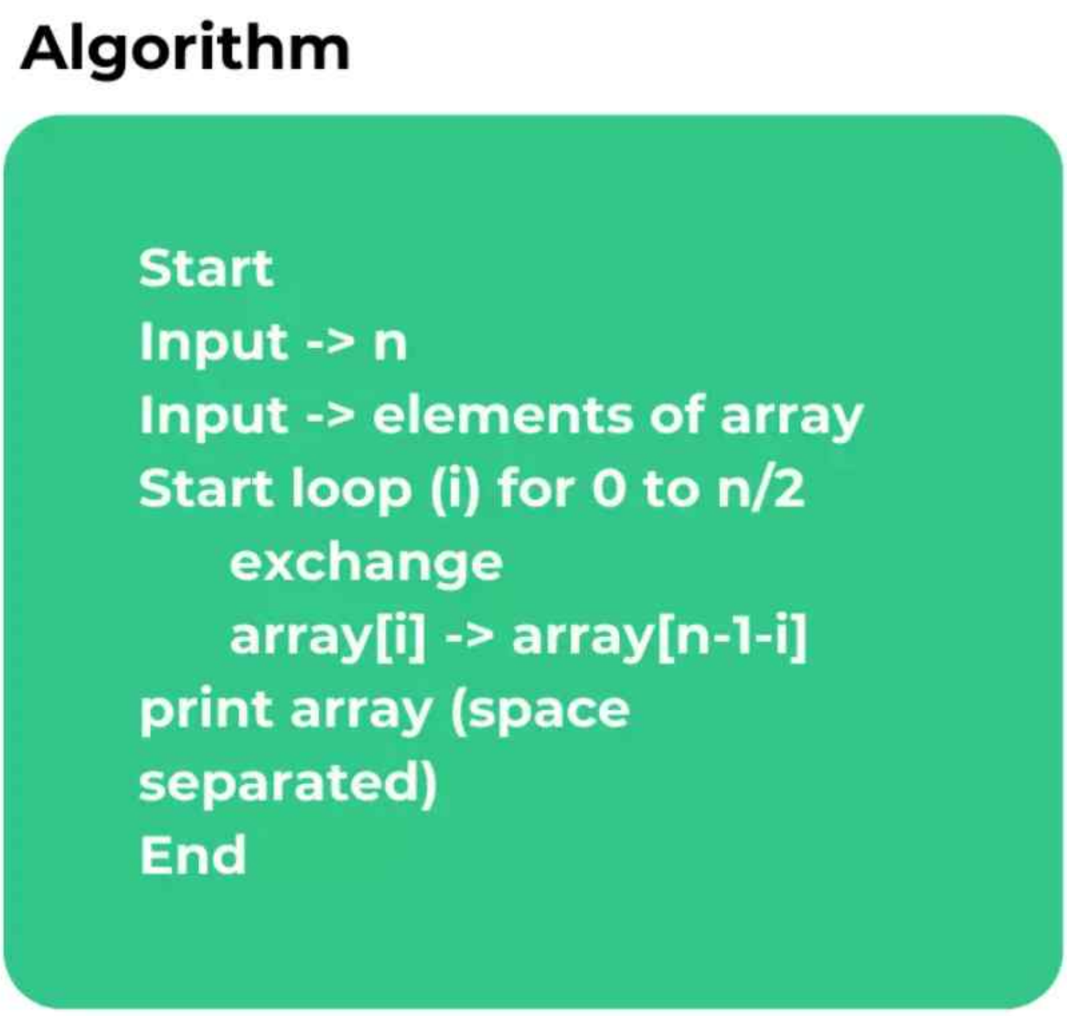
But Sajid's computer examination is scheduled for next week.

As per the blueprint, many questions would come from the Arrays topic.

He collected previous year's questions. one of the repeated questions is you need to reverse the array in C Programming Language.

Can you help him ?

Function Description



Algorithm

```
Start
Input -> n
Input -> elements of array
Start loop (i) for 0 to n/2
    exchange
    array[i] -> array[n-1-i]
print array (space
separated)
End
```

Constraints

0<n<100

0<arr<1000

Input Format

-First line will contain the number of elements present in the array.
-Second line will contain the elements of array

Output Format
-Output contain only one that space separated integers in reverse order.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
int i,n;
cin>>n;
int arr[n];
for(i=0;i<n;i++)
cin>>arr[i];
for(i=0;i<n/2;i++)
{
int temp;
temp=arr[i];
arr[i]=arr[n-1-i];
arr[n-1-i]=temp;
}
for(int i=0;i<n;i++)
cout<<arr[i]<<" ";

return 0;
}
```

Question 29

Question description

Simon is studying B.Tech.-Mechanical Engineering.

He's going to attend a computer science-based subject exam this semester.

Due to the less preparation in the previous monthly tests, his internal mark decreased.

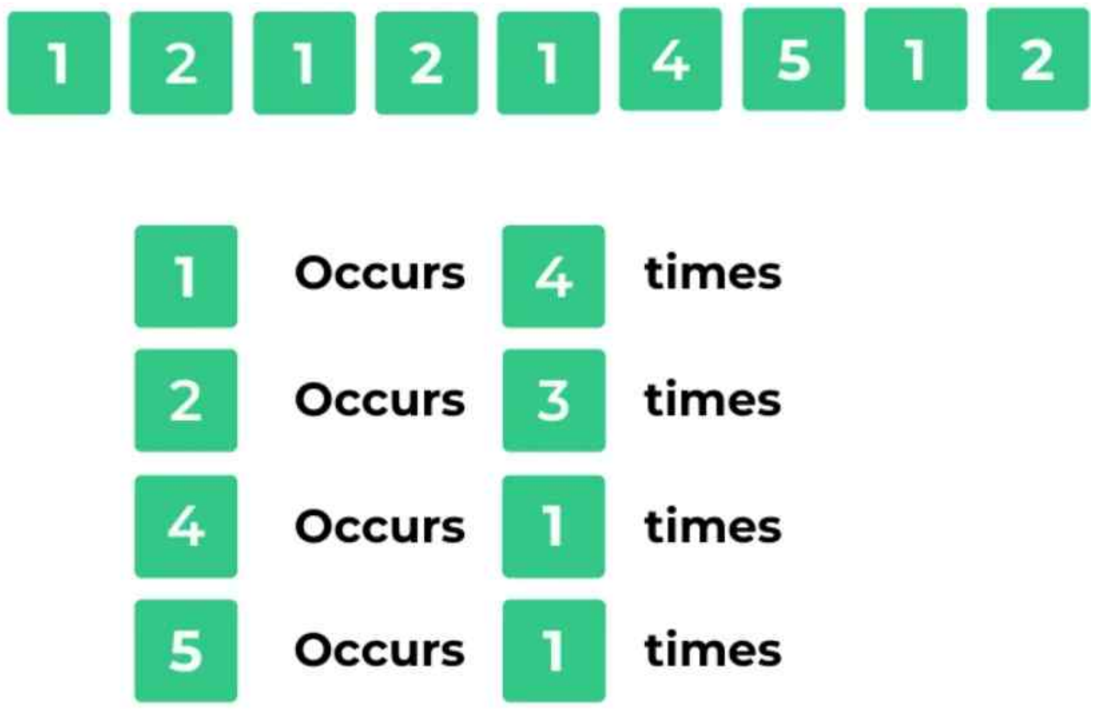
His computer science Professor made an offer one more chance to boost up his internal marks.

Professor assigns a program to Simon for the internal mark boostup.

So Simon wants to identify the element of array which occurs most time in the array

Can you help him ?

Function Description



Constraints

0<n<100

0<arr<1000

Input Format

-First line will contain the number of elements present in the array.
-Second line will contain the elements of array

Output Format
-Output contain only line that contains the element which occurs most times in the array.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int n;
cin>>n;
int arr[n];
for(int i=0;i<n;i++)
{
cin>>arr[i];
}

int i;
unordered_map<int, int> hash;
for(i= 0;i< n;i++)
hash[arr[i]]++;
int max_count = 0, res = -1;
for (auto i : hash) {
if (max_count < i.second) {
res = i.first;
max_count = i.second;
}
}
cout<<res<<"\n";
return 0;
}
```

Question 30

Problem Description:
Public school have arranged an Annual Day Function.

Volunteers have decorated a floor on various places of the school using Rose and Tulip flowers.

But one of the coordinators requested the volunteers to rearrange the decoration like a triangular size.

Coordinator also told them that tulips flowers need to be positioned at the middle of the roses

School has 20 buildings and as per Principal order the numbers of rows in the decoration should also match the building number.
The Principal of the school is interested in seeing the final decoration but he is quite busy with the other works.

So he likes to see how the final decoration have come through online mode if he gives the building number.

So can you display him the final decoration layout?
Note:
Roses are represented by 1.

Tulips are represented by 0.

Constraints:
1 ≤ rows ≤ 20

Input Format:
Only line of input has single integer representing the building number.

Output Format:
Print the final layout of the decoration.

Refer sample testcases for format specification.

Program Code:

```
#include <stdio.h>
int main()
{
int rows,i,j;
scanf("%d",&rows);
for(i=1;i<=rows;i++)
{
for(j=1;j<=i;j++)
{
if(j==1 || j==i || i==rows)
printf("1 ");
else
printf("0 ");
}
printf("\n");
}
return 0;
}
```

Question 32

Question description

Dr.Jegan is faculty, who handling data structure course for software engineering department second year students.

one day this faculty was handling very interesting topic in data structure such that Linked List, he has given the following explanation for Linked list concept.

"Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List.

Link − Each link of a linked list can store a data called an element.

Next − Each link of a linked list contains a link to the next link called Next.

LinkedList − A Linked List contains the connection link to the first link called First."

During this lecture time, last bench students was asking surprise test for Linked list concept.

So the faculty decided to conduct test on the topic of Linked List.

the question was given to last bench students that is,

The nodes are deleted D times from the beginning of the given linked list.

For example if the given Linked List is 5->10->15->20->25 and remove 2 nodes,

then the Linked List becomes 15->20->25.

Constraint :

1< N < 1000

1< P < N-1

INPUT Format

First line contains the number of datas- N.

Second line contains N integers(the given linked list).

Third line contains no. of nodes to be deleted.

OUTPUT Format

Single line represents the final linked list after deletion.

Program Code:

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

int data;
node *next;
};
void insertAtEnd(node** head_ref, int new_data) {
node* new_node = (node*)malloc(sizeof( node));
node* last = *head_ref;
new_node->data = new_data;
new_node->next = NULL;
if (*head_ref == NULL) {
*head_ref = new_node;
return;
}
while (last->next != NULL) last = last->next;
last->next = new_node;
return;
}
int main() {
node* head = NULL;
int n,c,z,i;
cin>>n;
for(i=0;i<n;i++){
cin>>c;
insertAtEnd(&head,c);
}
cin>>z;
for(int i=0;i<z;i++)
head=head->next;
cout << "Linked List:";
node* node=head;
while(node!=NULL){
cout<<"->"<<node->data;

node=node->next;
}
return 0;
cout<<"void create()";
}
```

Question 33

Question description

Lalitha is a IT expert who training youngsters struggling in coding to make them better.

Lalitha usually gives interesting problems to the youngsters to make them love the coding.One such day Lalitha provided the youngsters to solve that reverse the linked list.

The newly inserted node becomes the Linked List's new head.

If the current Linked List is 11->151->201->251, for example,

We add item 5 to the front of the list.

The Linked List will then be 251->201->151->11.

Let's call the function that moves the item to the top of the list push ().

The push() must receive a pointer to the head pointer, because push must change the head pointer to point to the new node

Constraints:

1 < arr <100

INPUT

First line contains the number of datas- N. Second line contains N integers(i.e, the datas to be inserted).

OUTPUT

Display the final Linked List.

Program Code:

```
#include <iostream>
using namespace std;
void ff()
{
return;
}
struct node
{
int data;
node *next;
}*start = NULL;
int main()
{
int n;
cin>>n;
int arr[n];
for(int i =0; i<n;i++)
{
cin>>arr[i];
}
cout<<"Linked List:";
for(int j = n-1;j >= 0 ; j--)
cout<<"->"<<arr[j];
// display(start);
return 0;
cout<<"struct node node *next; *start p1->next=start; void display()";
}
```

Question 34

Question description

Varman's Dream came true after he got an Appointment order from Google.Simon's family was very happy of his achievement.

The company mentioned Basic Salary, DA, HRA with some other benefits.

But not highlighted the Gross salary in the order.

varman's father wanted to know the Gross salary of his son.

varman try to his gross salary from HR department. they informed that you have to get pass grade in first month entry test. the entry test has 5 questions. one of the question was, Sorted insert in circular linked list.

Can you help varman?

Function Description

First case one is if linked list is empty then since new_node is only node in circular linked list, make a self loop.and change the head pointer to the new_node pointer.

Second case is new node insert in starting or before the head node.

A- Find out the last node using a loop .
    While(present->!=*head_ref)
    present=present->next;
B- Change the next of last node;
    present->next=new-node;
C- Change next of new node to point to head.
    new_node->next=*head_ref;
D- Change the head pointer to point to new node.
    *head_ref=new_node;

Third case is when we insert the new node after the head in any position,then

A- Locate the node after which new node is to be inserted.
    while(present->next!= *head_ref &&
    present->next->data data)
    { present = present->next; }
B- Make next of new_node as next of the located pointer
    new_node->next = present->next;
C- Change the next of the located pointer
    present->next = new_node;

Constraints

0<n<100

Input Format:
The First line of the input represents the number of elements

Second line represents the elements of circular linked list

Output Format:
single line prints the results as per sample test cases

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int n;cin>>n;vector<int> v(n);
for(auto &el:v) cin>>el;
```

```
sort(v.begin(),v.end());
for(auto el:v) cout<<el<<' ';
return 0;
cout<<"struct Node *next; void sortedInsert(struct Node** head_ref, struct Noc
}
```

Question 36

Question description

Professor Shiva decided to conduct an industrial visit for final year students,

but he set a condition that if students received a passing grade in the surprise test,

they would be eligible to go on the industrial visit.

He asked the students to study a topic linked list for 10 minutes before deciding to conduct a surprise test.

Professor-mandated questions, such as the deletion of nodes with a certain data D, are now being asked.

For example

if the given Linked List is 5->10->15->10->25 and delete after 10 then the Linked List becomes 5->15->25.

Constraints

1< N < 100

1< D < 1000

Input Format
First line contains the number of datas- N.

Second line contains N integers(the given linked list).

Next line indicates the node data D that has to be deleted.

Output Format
Single line represents the linked list after required elements deleted.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
void ss()
{
return;
}
int main()
{

int n;
cin>>n;
int arr[n];
for (int i = 0; i < n; ++i)
{
cin>>arr[i];
}
int m;
cin>>m;
cout<<"Linked List:";
for(int p : arr)
{
if(p != m)
cout<<"->"<<p;
}
return 0;
cout<<"struct node node *next; void create() p2=p2->next; void del()";
}
```

Question 39

Question description

Dr.Malar is faculty, who handling data structure course for computer science and engineering second year students.

one day this faculty was handling very interesting topic in data structure such that Linked List, she has given the following explanation for Linked list concept.

"Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List.

Link − Each link of a linked list can store a data called an element.

Next − Each link of a linked list contains a link to the next link called Next.

LinkedList − A Linked List contains the connection link to the first link called First."

During this lecture time, last bench students was making continuous disturbance by making unwanted noise.

So the faculty decided to conduct test on the topic of Linked List.

the question was given to last bench students that is,

The new node is added at given position P of the given Linked List.

For example if the given Linked List is 5->10->15->20->25 and

we add an item 30 at Position 3,

then the Linked List becomes 5->10->30->15->20->25.

Since a Linked List is typically represented by the head of it,

we have to traverse the list till P and then insert the node.

Constraints

1<N<1000

1<P< N+1

Input Format

First line contains the number of datas- N. Second line contains N integers(the given linked list).

Third line contains the position P where the node to be inserted. Fourth line contain the node X to be inserted.

Output Format

Single line represents the final linked list

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
struct node
{
int data;
struct node *next;
}*head = NULL;
int n;
int in_pos(int n)
{
int data1;
cin>>data1;
int i =1;

struct node *r = head;
while(i != n-1)
{
r = r-> next;
i++;
}
node *tt = new node;
tt -> data = data1;
tt -> next = r -> next;
r -> next = tt;
node *s = head;
cout<<"Linked List:";
while(s != NULL)
{
cout<<"->";
cout<<s-> data;
s = s-> next;
}
return data1;
}
```

```
void create()
{
int n;
cin>>n;
struct node *p = new node;
int __n;
cin>>__n;
p -> data = __n;

head = p;
int i;
for(i=0;i<n-1;i++)
{
int a;
cin>>a;
struct node *q = new node;
q -> data = a;
p -> next= q;
p = p->next;
}
p -> next = NULL;
}
int main()
{
create();
int r;
cin>>r;
int s = in_pos(r);
return 0;
cout<<s<<"for(i=0;i<n;i++)";
}
```

Question 40

Question description

Dr.Siva jayaprakash is a faculty, who handling data structure course for IT department second year students.

one day this faculty was handling very interesting topic in data structure such that Linked List, he has given the following explanation for Linked list concept.

"Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List.

Link − Each link of a linked list can store a data called an element.

Next − Each link of a linked list contains a link to the next link called Next.

LinkedList − A Linked List contains the connection link to the first link called First."

During this lecture time, principal surprisingly visited to the class and asking to conduct surprise test on Linked list concept.

So the faculty decided to conduct test on the topic of Linked List.

the question was given to last bench students that is,

The nodes are deleted D times from the end of the given linked list.

For example if the given Linked List is 5->10->15->20->25 and remove 2 nodes,

then the Linked List becomes 5->10->15.

Constraint :

1< N < 1000

1< P < N-1

INPUT Format

First line contains the number of datas- N.

Second line contains N integers(the given linked list).

Third line contains no. of nodes to be deleted.

OUTPUT Format

Single line represents the final linked list after deletion.

Program Code:

```
#include <iostream>
using namespace std;
void tel()
{
return;
}
struct node
{
int data;
node *next;
}*head = NULL;
void create()
{
int n;
cin>>n;
struct node *p1 = new node;
int m;
cin>>m;
p1 -> data = m;

head = p1;
int i;
for(i=0;i<n-1;i++)
{
```

```
int a;
cin>>a;
node *tt = new node;
tt -> data = a;
p1 -> next = tt;
p1=p1->next;
}
p1 -> next = NULL;
int del;
bool found = false;
cin>>del;
node *nn = head;
while(nn != NULL)
{
nn = nn -> next;
node *dd = nn;
int m = del; while(m-- > -1)
{
dd = dd -> next; if(dd == NULL)
{
nn -> next = NULL;
found = true; break;}}
if(found) break; }
cout<<"Linked List:";

while(head != NULL)
{
cout<<"->"<<head -> data;
head = head -> next; }}
int main()
{
create();
return 0;
cout<<"for(i=0;i<n;i++)";
}
```

Question 41

Question description

Rajinikanth organized a technical round interview in an Animation company for the set of computer science candidates.

the task is to implement stack operations for two stacks and merge the stacks into one.

Get two sets of stack elements and reverse them, then merge them into one stack.

Rajinikanth has given the deadline of only 15 minutes to complete the problem.

Can you Help the candidates to complete the problem within the specified time limit?

Function Description

a) push(): Adds the new item at the beginning of the linked list using the first pointer.
b) pop(): Removes an item from the beginning using the first pointer.
c) merge(): Links the first pointer second stack as next of the last pointer of the first list.

Constraints

0 < n, m < N

1 < arr[i] < 1000

Input Format:

First-line indicates n & m, where n is the number of elements to be pushed into the stack and m is the number of pop operations that need to be performed

The next line indicates the n number stack elements

Output Format:

First-line indicates the top of the element of the stack

the second line indicates the top of the element after the pop operation

Program Code:

```
#include <iostream>
using namespace std;
class node {
public:
int data;
node* next;
};
class mystack {
public:
node* head;
node* tail;

mystack()
{
head = NULL;
tail = NULL;
}
};
mystack* create()
{
mystack* ms = new mystack();
return ms;
}
void push(int data,mystack* ms)
{
node* temp = new node();
temp->data = data;
temp->next = ms->head;

if (ms->head == NULL)
ms->tail = temp;
```

```
ms->head = temp;
}
int pop(mystack* ms)
{
if (ms->head == NULL) {
cout << "stack underflow" << endl;
return 0;
}
else {
node* temp = ms->head;
ms->head = ms->head->next;
int popped = temp->data;
delete temp;
return popped;
}
}
void merge(mystack* ms1,mystack* ms2)
{
if (ms1->head == NULL)
{
ms1->head = ms2->head;
ms1->tail = ms2->tail;
return;
}

ms1->tail->next = ms2->head;
ms1->tail = ms2->tail;
}
void display(mystack* ms)
{
node* temp = ms->head;
while (temp != NULL) {
cout << temp->data << " ";
temp = temp->next;
}
}
int main()
{
mystack* ms1 = create();
mystack* ms2 = create();
int n,m,t;
cin>>n>>m;
for(int i=0;i<n;i++)
{
cin>>t;
push(t,ms1);
}
for(int i=0;i<m;i++)
{
cin>>t;
push(t,ms2);
}
merge(ms1, ms2);
```

```
for(int i=0;i<n+m;i++)
cout<<pop(ms1)<<" ";
}
```

Question 42

Question description

Hassan enjoys jumping from one building to the next. However, he merely jumps to the next higher building and stops when there are none accessible. The amount of stamina necessary for a voyage is equal to the xor of all the heights Hassan leaps till he comes to a halt.

If heights are [1 2 4], and he starts from 1, goes to 2 stamina required is $1\oplus2=3$, then from 2 to 3. Stamina for the entire journey is $1\oplus2\oplus4=7$. Find the maximum stamina required if can start his journey from any building.

Constraints

1≤ N ≤10^5

1 ≤ Height ≤10^9

Input

First line: N, no of buildings.

Second line: N integers, defining heights of buildings.

Output

Single Integer is the maximum stamina required for any journey.

Explanation:

8

1 2 3 8 6 4 7 9

Considering the input given by you, 1st starting point is 1. From this building, Hassan will try to find the next higher building that is building 2 here and so on. Likewise he will complete all pass.

Pass 1: 1-> 2->3->8-> 9, XOR = 1

Pass 2: 2->3->8-> 9, XOR = 0

Pass 3: 3->8-> 9, XOR = 2

Pass 4: 8-> 9, XOR = 1

Pass 5: 6->7->9 , XOR = 8

Pass 6: 4->7->9 , XOR = 10

Pass 7: 7-> 9 XOR = 14

Pass 8: 9 XOR = 9

So, the highest XOR value is 14

Program Code:

```
#include <stdio.h>
int main() {
int i, j, arr[1000000], n, temp=0,st[1000000]= {0};
scanf("%d",&n);
for(i=0;i<n;i++){
scanf("%d",&arr[i]);
}
st[n-1] = arr[n-1];
temp = arr[n-1];
for(i=n-2;i>=0;i--) {
for(j=i+1;j<n;j++)
if(arr[i]<arr[j]) {
st[i]=arr[i]^st[j];
break;
}
if(st[i] == 0)

st[i] = arr[i];
if(st[i] > temp)
temp = st[i];
}
printf("%d",temp);
return 0;
}
```

Question 43

Question description

Hassan gets a job in a software company in Hyderabad. The training period for the first three months is 20000 salary. Then incremented to 25000 salaries.

Training is great but they will give you a programming task every day in three months. Hassan must finish it in the allotted time. His teammate Jocelyn gives him a task to complete the concept of Prefix to Postfix Conversion for a given expression. can you help him?

Functional Description:

- Read the Prefix expression in reverse order (from right to left)
- If the symbol is an operand, then push it onto the Stack
- If the symbol is an operator, then pop two operands from the Stack
  Create a string by concatenating the two operands and the operator after them.
  string = operand1 + operand2 + operator
  And push the resultant string back to Stack
- Repeat the above steps until end of Prefix expression.

Constraints

the input should be a expressions

Input Format

Single line represents the prefixed expressions

Output Format

Single line represents the postfix expression

Program Code:

```
#include <iostream>
#include <stack>
using namespace std;
bool isOperator(char x)
{
switch (x) {
case '+':
case '-':
case '/':
case '*':
return true;
}
return false;

}
string preToPost(string pre_exp)
{
stack<string> s;
int length = pre_exp.size();
for (int i = length - 1; i >= 0; i--)
{
if (isOperator(pre_exp[i]))
{
string op1 = s.top();
s.pop();
string op2 = s.top();
s.pop();
string temp = op1 + op2 + pre_exp[i];
s.push(temp);
}
else {
s.push(string(1, pre_exp[i]));
}
}
return s.top();
}
int main()
{
string pre_exp;
cin>>pre_exp;
cout << "Postfix:" << preToPost(pre_exp);
return 0;
}
```

Question 45

Question description

Hassan gets a job in a software company in Hyderabad. The training period for the first three months is 20000 salary. Then incremented to 25000 salaries.

Training is great but they will give you a programming task every day in three months. Hassan must finish it in the allotted time. His teammate Jocelyn gives him a task to complete the concept of Postfix to Infix Conversion for a given expression. can you help him?

Functional Description:

- Read the next symbol from the input.
- Push it onto the stack.
- the symbol is an operator.
- Pop the top 2 values from the stack.
- Put the operator, with the values as arguments and form a string.
- Push the resulted string back to stack.
- If there is only one value in the stack
- That value in the stack is the desired infix string.

Constraints

the input should be a expressions

Input Format

Single line represents the postfixed expressions

Output Format

Single line represents the Infix expression

Program Code:

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

bool isOperand(char x){
return (x>='a' && x<='z') || (x >= 'A' && x <= 'Z');

}

string getInfix(string exp)
{
stack<string> s;

for(int i=0; exp[i]!='\0'; i++)
{
if(isOperand(exp[i]))
{
```

```
string op(1, exp[i]);
s.push(op);
}

else
{
string op1 = s.top();
s.pop();
string op2=s.top();
s.pop();
s.push("(" + op2 + exp[i] + op1 + ")");
}
}
return(s.top());
}

int main()
{
string exp;
cin>>exp;
cout<<getInfix(exp);

return 0;
}
```

Question 47

Problem Description:

You are given an array A of Q integers and Q queries. In each query, you are given an integer i (1≤i≤N).

Your task is to find the minimum index greater than i (1≤i≤N) such that:

1. Sum of digits of Ai is greater than the sum of digits of Aj
2. Ai < Aj

If there is no answer, then print **-1**.

Constraints

1 < = N, Q ≤ 10^5

1 ≤ Ai ≤ 10^9

1 ≤ Qi ≤N

Input format

- The first line contains two numbers N and Q.
- The next line contains N numbers.
- Next Q lines contain Q queries.

Output format

Print the answer as described in the problem

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main(){

int n,q;
cin>>n>>q;
int *a=new int [n];
for(int i=0;i<n;i++){
cin>>a[i];
}
int *arr=new int[n];
for(int i=0;i<n;i++){
int z=a[i];
int sum=0;
while(z>0){
sum+=(z%10);
z=z/10;
}
arr[i]=sum;
}
while(q--){
int Q;
cin>>Q;
int ans=-1;
for(int i=Q;i<n;i++){
if(a[i]>a[Q-1] && arr[i]<arr[Q-1]){
ans=i+1;
break;
}else{

continue;
}
}
cout<<ans<<' ';
}
return 0;
cout<<"if(arr[x]<arr[y]) if(arr2[x]>arr2[y]) ";
}
```

Question 48

Problem Description:

Sajid was booking a tour package of IRCTC from Chennai to Delhi for his family.

Two of the relatives was interested in joining to this tour.

these two persons are studying engineering in computing technology. only one tickets are remaining in the IRCTC portal.

So, Sajid decided to book one ticket for out of those persons also along with his family members.

He wants to identify the one person out of these persons. he decided to conduct a technical task to identify the right person to travel.

the task was that, Check for Balanced Brackets in an expression using Stack

Can you help them to complete the task?

Function Description

- Declare a character stack S.
- Now traverse the expression string exp.
    1. If the current character is a starting bracket (**'(' or '{' or '['**) then push it to stack.
    2. If the current character is a closing bracket (**')' or '}' or ']'**) then pop from stack and if the popped character is the matching starting bracket then fine else brackets are not balanced.
- After complete traversal, if there is some starting bracket left in stack then "not balanced"

Input Format

Single line represents only braces (both curly and square)

Output Format

If the given input balanced then print as Balanced (or) Not Balanced

Program Code:

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

bool areBrBalanced(string xi)
{
stack<char> s;
char x;
int y = xi.length();
for(int i=0; i < y; i++)
{
if(xi[i] == '(' || xi[i] == '{' || xi[i] =='[')
{
s.push(xi[i]);
continue;
}

if(s.empty())
return false;

switch(xi[i]){
case ')': x= s.top();
s.pop();
if(x == '{' || x == '[') return false;
break;
case '}': x = s.top();
s.pop();
if(x == '(' || x == '[') return false;
break;

case ']': x = s.top();
s.pop();
if(x == '{' || x == '(') return false;
break;
}
}

return (s.empty());
}
int main()
{
string expr;

cin>>expr;

if(areBrBalanced(expr))
cout<<"Balanced";
else
cout<<"Not Balanced";

return 0;
}
```

Question 49

Question description

First off, some definitions.
An array of length at least 2 having distinct integers is said to be fantabulous iff the second highest element lies **strictly to the left** of the highest value.

For example, *[1, 2, 13, 10, 15]* is fantabulous as the second-highest value *13* lies to the left of the highest value *15*.
For every fantabulous array, we define a fantabulous pair **(a, b)** where **a** denotes the index of the second-highest value (1-indexed) and **b** denotes the index of the highest value (1-indexed).

In the above array, the fantabulous pair is (3, 5).
Mancunian challenges you to solve the following problem.

Given an array, find the total number of **distinct** fantabulous pairs overall its subarrays.

Constraints:
$1 <= N <= 10^6$
$1 <=$ array elements $<= 10^9$
Array elements are distinct.

Input:
The first line contains an integer **N** denoting the length of the array. The next line contains **N distinct**

integers denoting the elements of the array.

Output:
Output a single integer which is the answer to the problem.

Program Code:

```
#include <bits/stdc++.h>

using namespace std;

#define sci(x) scanf("%d", &x)
#define scl(x) scanf("%lld", &x)

int arr[1000001], cnt[1000001];
int v[1000001];
stack <int> st;

void don(){

cout<<"void push(llint num)stack[top++]=num;pop()";
}

int main()
{
int n, i, x;
sci(n);
for (i = 1; i <= n; ++i) sci(arr[i]);

for (i = n; i > 0; --i) {
while (!st.empty() && arr[i] > arr[st.top()]) {
cnt[st.top()] = st.top() - i;
st.pop();
}
st.push(i);
}
while (!st.empty()) {
cnt[st.top()] = st.top();
st.pop();
}

for (i = 1; i <= n; ++i) {
while (!st.empty() && arr[st.top()] < arr[i]) {
x = i - st.top() + 1;
v[x] = max(v[x], cnt[st.top()]);
st.pop();
}
st.push(i);
}

int k = 0;
for (i = 2; i <= n; ++i) {
k += v[i];
}

cout << k << endl;

return 0;
}
```

Question 50

Question description

You're given a stack of N numbers, with the first component representing the stack's top and the final component being the stack's bottom.

At least one piece from the stack must be removed. You can turn the stack into a queue at any time.

The front of the line is represented by the bottom of the stack.

You cannot convert the queue back into a stack. Your task is to remove exactly **K** elements such that the sum of the **K** removed elements is maximized.

constraints:

$1 \leq N \leq 10^5$

$1 \leq K \leq N$

$1 \leq Ai \leq 10^9$

Input format :

- The first line consists of two space-separated integers **N** and **K**.
- The second line consists of **N** space-separated integers denoting the elements of the stack.

Output format :

- Print the maximum possible sum of the **K** removed elements

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int n,k,i;

cin>>n>>k;
int sum = 0;
int arr[n];
stack<int>st, st2;
for(i=0;i<n;i++){
cin >> arr[i];
st.push(arr[i]);
}
for(i=0;i<k;i++){
st2.push(arr[i]);
sum += arr[i];
}
int maxs = sum;
while(k-- > 1){
sum -= st2.top();
st2.pop();
sum += st.top();
st.pop();
if(sum > maxs) maxs = sum;
}
cout << maxs;
return 0;
}
```

Question 51

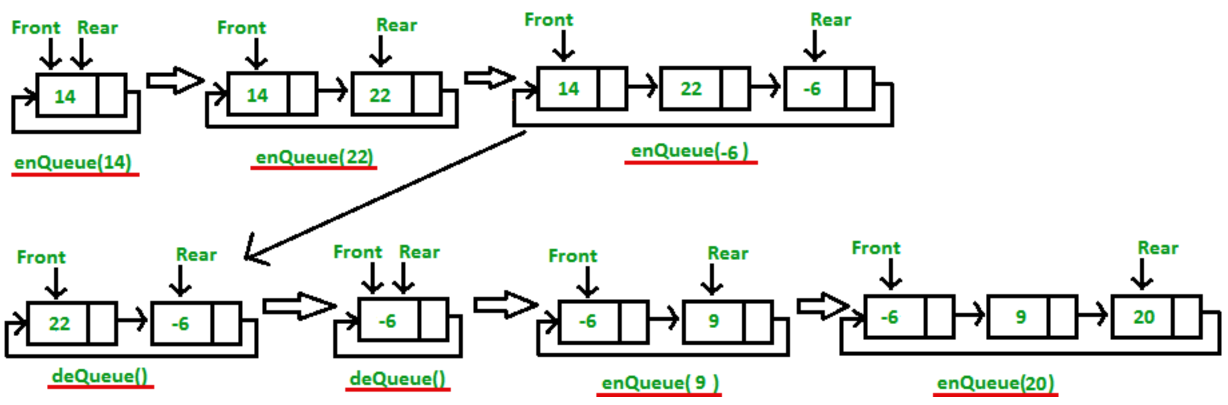
Question description

Lalitha is a B.Tech student. During her final year Campus Interview, she has an opportunity to get a job in a software company in Bangalore.

The company provides Five months training period with Rs.30000/month Package. Then it will be incremented to Rs.55000 per month.

At the end of the training, the examination was conducted for all freshers, Lalitha got a question paper and one of the questions comes under the concept of Queues in data structure that is Circular Linked List Implementation in Circular Queue.

Can you help?



Constraints

First ‘3’ elements inserted in array 0, next ‘3’ elements inserted into array1, remaining elements inserted into array2.

Input Format

First line indicates the number of elements to be inserted in the queues.

Second line indicates the elements.

Output Format

first line indicates the dequeue element of Q2

Second line indicates the dequeue element of Q1

Third line line indicates the dequeue element of Q0

Note: Refer sample input and output test cases

Program Code:

```
#include <stdio.h>
#include <stdlib.h>
struct node *front = NULL;
struct node *rear = NULL;
struct node
{
 int data;
 struct node *next;
};
void linkedListTraversal(struct node *ptr)
{
 printf("Elements in Circular Queue are:");
 while (ptr->next != NULL)
 {
 printf("%d ", ptr->data);
 ptr = ptr->next;
 }
 printf("%d",ptr->data);
}
void enqueue(int d)
{
 struct node* new_n;
 new_n = (struct node*)malloc(sizeof(struct node));
 if(new_n==NULL){
 printf("Queue is Full");
 }
 else{
 new_n->data = d;
 new_n->next = NULL;
 if(front==NULL){
 front=rear=new_n;
 }
 else{
 rear->next = new_n;
 rear=new_n;
 }
 }
}
int dequeue()
{
 int val = -1;
 struct node *ptr = front;
 if(front==NULL){
 printf("Queue is Empty\n");
 }
 else{
 front = front->next;
 val = ptr->data;
 free(ptr);
 }
 return val;
}
int main()
{
 int n,i,t;
 scanf("%d",&n);
 for(i=0;i<n;i++)
 {
 scanf("%d",&t);
 enqueue(t);
 }
 linkedListTraversal(front);
 printf("\nDeleted value = %d\n",dequeue());
 printf("Deleted value = %d",dequeue());
 linkedListTraversal(front);
 return 0;
 printf("void enQueue(Queue* q,int value) int deQueue(Queue* q) void displayQu

}
```

Question 52

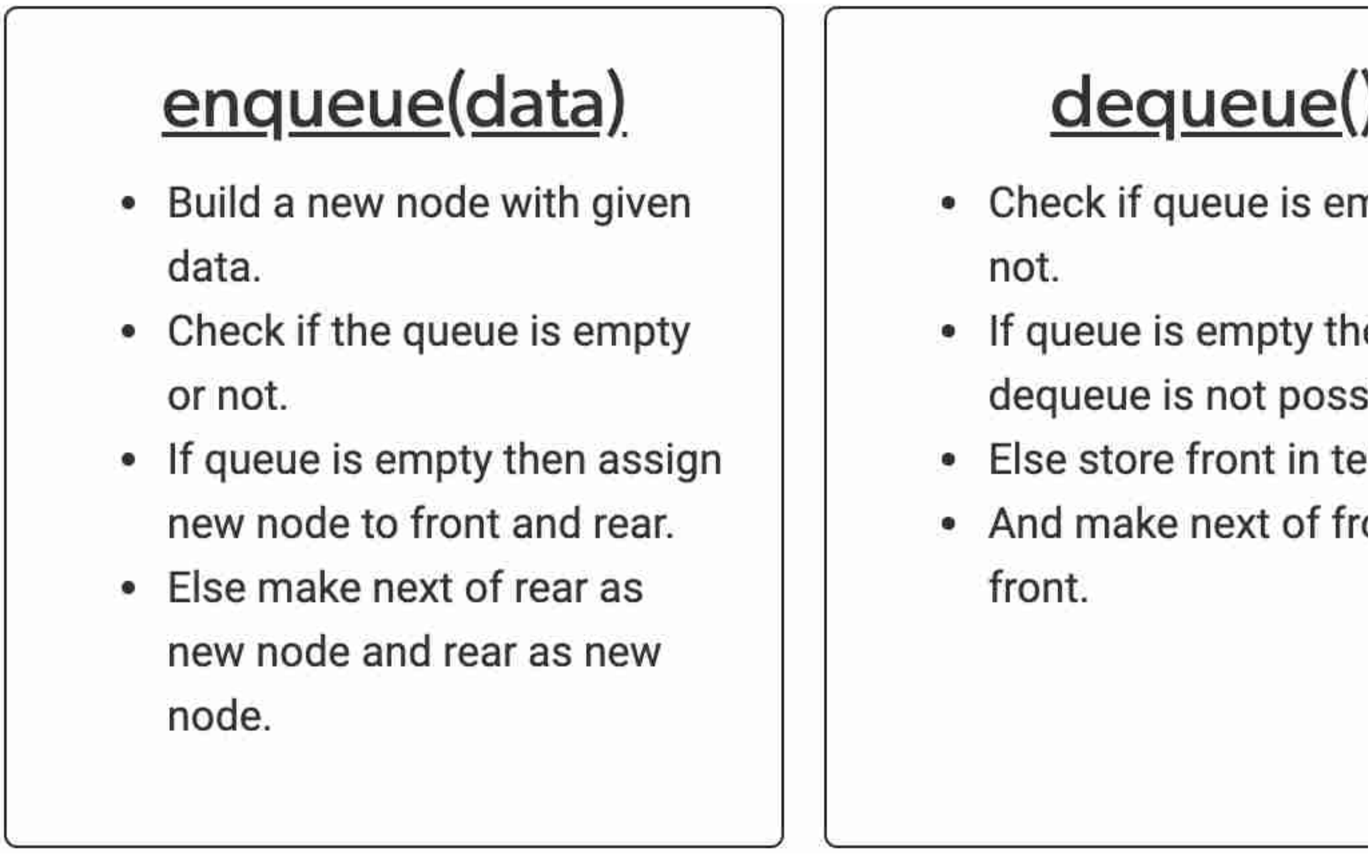
Question description

lala is a Placement trainer. he is working as CDC trainer in reputed institution that during training the youngsters are struggling in queue concept. Lala usually gives interesting problems to the students to make them love the DS. One such day Lala provided to the final year students to solve the task such that, Queue implementation with arrays as using linked list for implementing queue, Queue data structures work on the FIFO architecture so the element that has entered first in the list will go out from the list first.

Final Year students were lacking the idea to solve the problem.

Being an exciting youngster can you solve it?

Function Description


enqueue(data)

- Build a new node with given data.
- Check if the queue is empty or not.
- If queue is empty then assign new node to front and rear.
- Else make next of rear as new node and rear as new node.

dequeue(

- Check if queue is em not.
- If queue is empty th dequeue is not poss
- Else store front in te
- And make next of fr front.

Constraints:

0<size <100

0<data<1000

Input format:

First line indicates the size of the queue

Second line indicates the elements of the queue.

Output Format:

fSingle line indicates the inserted elements in queue using linked list concept

Program Code:

```
#include <stdio.h>
#include <stdlib.h>
struct node *front = NULL;
struct node *rear = NULL;
struct node
{
int data;
struct node *next;
};
void linkedListTraversal(struct node *ptr)
{
//printf("Printing the elements of this linked list\n");
while (ptr != NULL)
{
```

```
printf("%d ", ptr->data);
ptr = ptr->next;
}
}
void enqueue(int d)
{
struct node* new_n;
new_n = (struct node*)malloc(sizeof(struct node));
if(new_n==NULL){
printf("Queue is Full");
}
else{
new_n->data = d;

new_n->next = NULL;
if(front==NULL){
front=rear=new_n;
}
else{
rear->next = new_n;
rear=new_n;
}
}
}
int main()
{
int n,i,t;
scanf("%d",&n);
for(i=0;i<n;i++)
{
scanf("%d",&t);
enqueue(t);
}
linkedListTraversal(front);
return 0;
}
```

Question 53

Question description

Anderson is a Placement trainer. he is working as CDC trainer in reputed institution that during training the youngsters are struggling in queue concept.

Anderson usually gives interesting problems to the students to make them love the DS.

One such day Joe Anderson provided to the final year students to solve the task such that, Circular Queue using Linked List. there is no memory waste while using Circular Queue, it is preferable than using a regular queue.

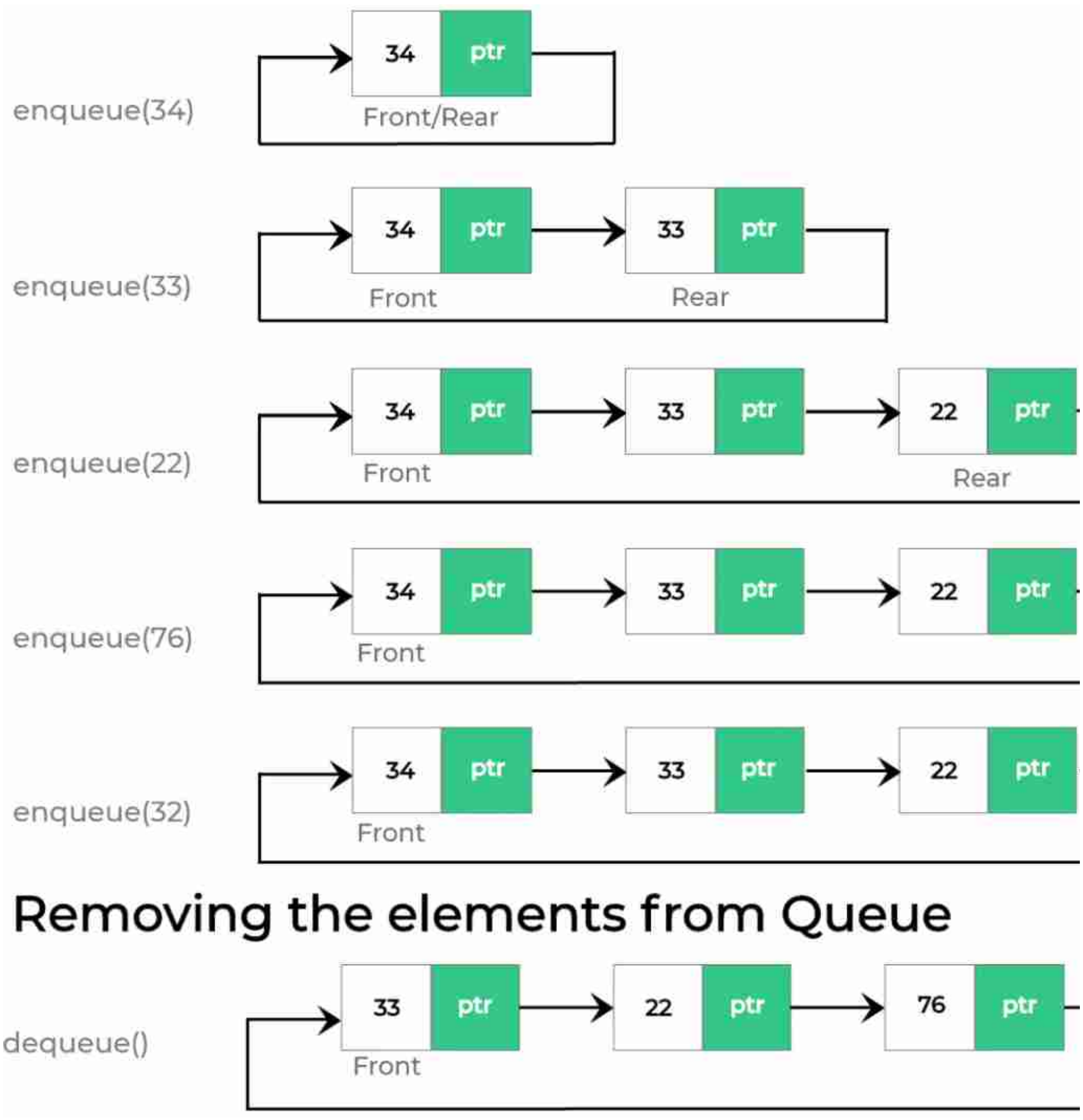
Because linked lists allow for dynamic memory allocation, they are simple to build.

Circular Queue implementation using linked list is identical to circular linked list except that circular Queue has two pointers front and back whereas circular linked list only has one pointer head.

Final Year students were lacking the idea to solve the problem.

Being an exciting youngster can you solve it?

Function Description



Constraints:

0<size <100

0<data<1000

Input format:

First line indicates the size of the queue

Second line indicates the elements of the queue.

Output Format:

First line indicates the inserted elements in queue using linked list concept

second line indicates the one element dequeued from the queue using linked list concept.

Program Code:

```
#include <stdio.h>
#include <stdlib.h>
struct node *f = NULL;
struct node *r = NULL;
struct node
{
 int data;
 struct node* next;
};
void enqueue(int d)
{
 struct node *n;
 n = (struct node*)malloc(sizeof(struct node));
 if(n==NULL){
 printf("Queue is Full");
 }
 else{
 n->data = d;
 n->next = NULL;
 if(f==NULL){
 f=r=n;
 }
 else{
 r->next = n;
 r=n;
 }
 }
}
int dequeue()
{
 int val = -1;
 struct node* t;
 t = f;
 if(f==NULL){
 printf("Queue is Empty\n");
 }
 else{
 f = f->next;
 val = t->data;
 free(t);
 }
 return val;
}
int main()
{
 int n,i,t;
 scanf("%d",&n);
 for(i=0;i<n;i++)
 {
 scanf("%d",&t);
 enqueue(t);
 }
 for(i=0;i<n;i++){
 printf("%d\n",dequeue());
 }
 return 0;
}
```

Question 54

Question description

On the last day of the semester, Shahid's students were talking and playing very loudly to the delight of the end of the semester. The principal of the college severely reprimanded Shahid. But he decided to engage them in a different activity without getting angry at the students.

So Shahid gave his students to solve the task such that, Circular Queue using Linked List and dequeue two elements from circular queue.

there is no memory waste while using Circular Queue, it is preferable than using a regular queue.

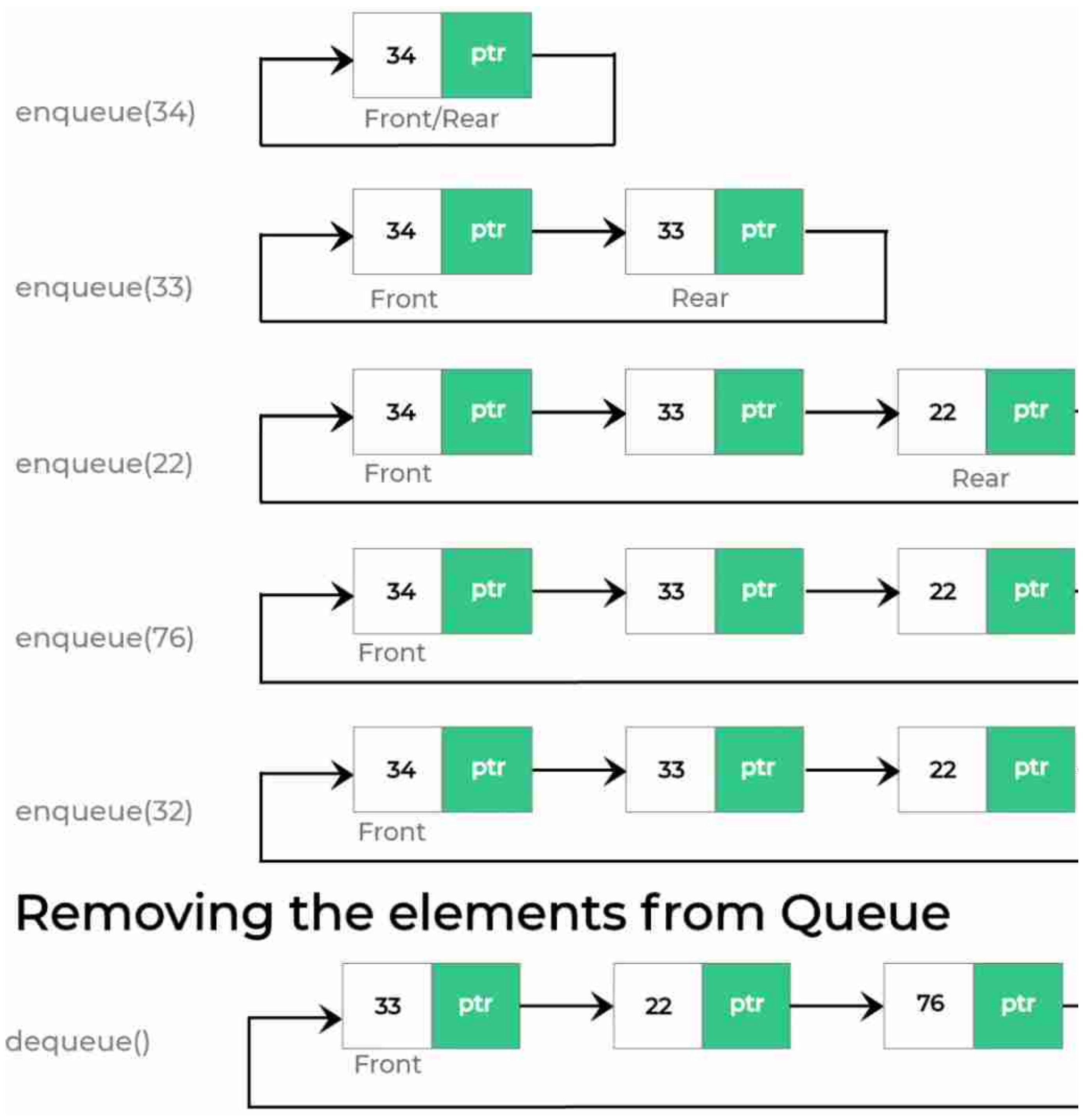
Because linked lists allow for dynamic memory allocation, they are simple to build.

Circular Queue implementation using linked list is identical to circular linked list except that circular Queue has two pointers front and back whereas circular linked list only has one pointer head.

Final Year students were lacking the idea to solve the problem.

Being an exciting youngster can you solve it?

Function Description



Constraints:

0<size <100

0<data<1000

Input format:

First line indicates the size of the queue

Second line indicates the elements of the queue.

Output Format:

First line indicates the inserted elements in queue using linked list concept

second line indicates the one element dequeued from the queue using linked list concept.

Program Code:

```
#include <stdio.h>
#include <stdlib.h>
struct node *f = NULL;
struct node *r = NULL;
struct node
{
 int data;
 struct node* next;
};
void linkedListTraversal(struct node *ptr)
{
 //printf("Printing the elements of this linked list\n");
 while (ptr != NULL)
 {
 printf("%d ", ptr->data);
 ptr = ptr->next;
 }
}
void enqueue(int d)
{
 struct node *n;
 n = (struct node*)malloc(sizeof(struct node));
 if(n==NULL){
 printf("Queue is Full");
 }
 else{
 n->data = d;
 n->next = NULL;
 if(f==NULL){
 f=r=n;
 }
 else{
 r->next = n;
 r=n;
 }
 }
}
int dequeue()
{
 int val = -1;
 struct node* t;
 t = f;
 if(f==NULL){
 printf("Queue is Empty\n");
 }
 else{
 f = f->next;
 val = t->data;
 free(t);
 }
 return val;
}
```

```
int main()
{
 int n,i,t;
 scanf("%d",&n);
 for(i=0;i<n;i++)
 {
 scanf("%d",&t);
 enqueue(t);
 }
 linkedListTraversal(f);
 for(i=0;i<2;i++){
 dequeue();
 printf("\n");
 linkedListTraversal(f);
 }
 return 0;
}
```

Question 55

Question description

Consider the following string transformation:

1. append the character # to the string (we assume that # is lexicographically smaller than all other characters of the string)
2. generate all rotations of the string
3. sort the rotations in increasing order
4. based on this order, construct a new string that contains the last character of each rotation

For example, the string babc becomes babc#. Then, the sorted list of rotations is #babc, abc#b, babc#, bc#ba, and c#bab. This yields a string cb#ab.

Constraints

- 1≤n≤10^6

Input

The only input line contains the transformed string of length n+1. Each character of the original string is one of a–z.

Output

Print the original string of length n.

Program Code:

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

int main() {
int i;
string s; cin>>s;
vector<int> v;
vector<int> a[26];

int n= s.size();
for(i=0;i<=n;i++) {
if (s[i] == '#')
v.push_back(i);
else
a[s[i]-'a'].push_back(i);
}
for (int i = 0; i < 26; i++) {
for (auto j: a[i])
v.push_back(j);
}
string ans;
int j = v[v[0]];
while(s[j] != '#') {
ans += s[j];
j = v[j];
}
cout<<ans;
return 0;
}
```

Question 56

Question description

Joe root is a Placement trainer. he is working as CDC trainer in reputed institution that during training the youngsters are struggling in queue concept.

Joe root usually gives interesting problems to the students to make them love the DS.

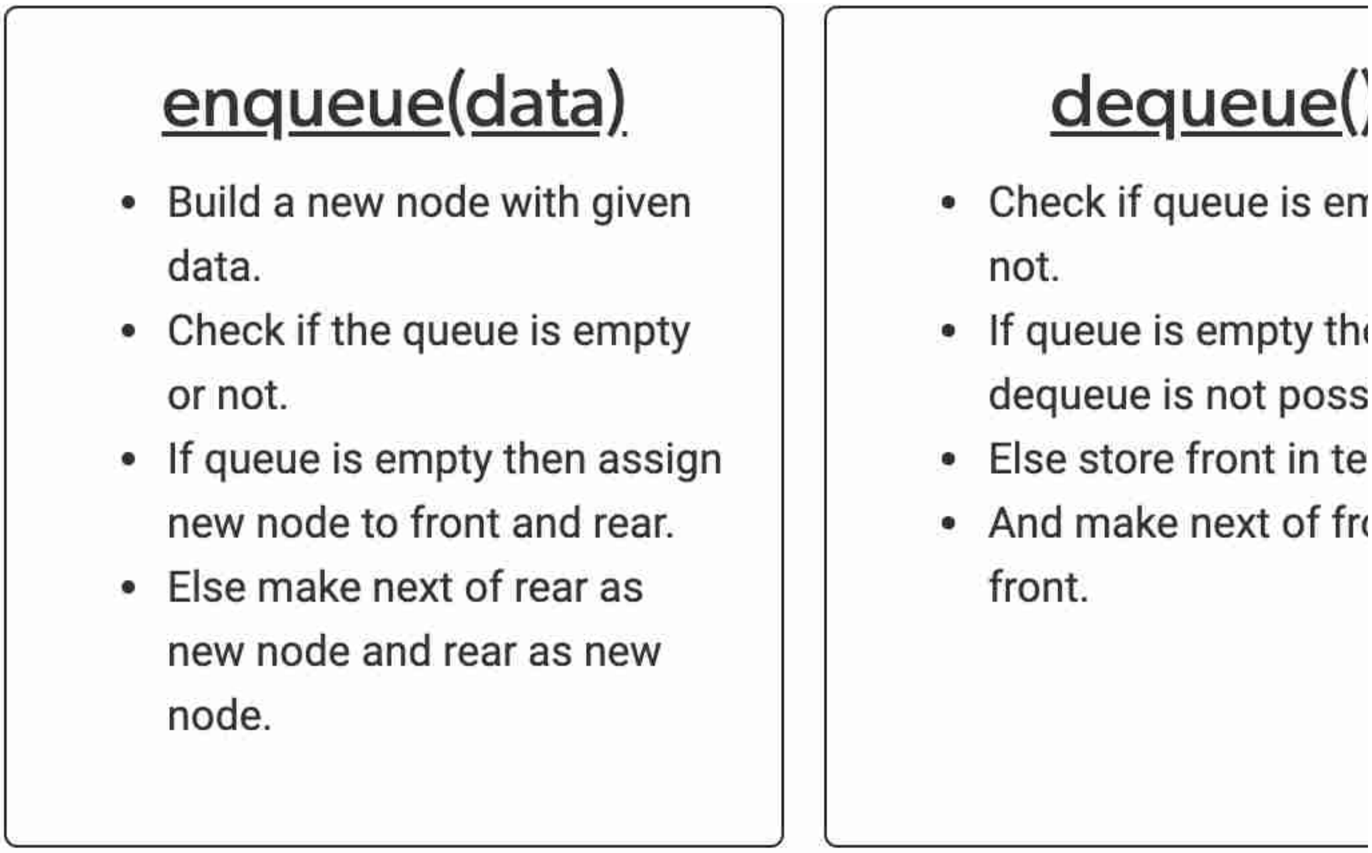
One such day Joe root provided to the final year students to solve the task such that, Queue implementation with arrays as using linked list for implementing queue and delete an element from the queue using linked list concept.

Queue data structures work on the FIFO architecture so the element that has entered first in the list will go out from the list first.

Final Year students were lacking the idea to solve the problem.

Being an exciting youngster can you solve it?

Function Description



enqueue(data)

- Build a new node with given data.
- Check if the queue is empty or not.
- If queue is empty then assign new node to front and rear.
- Else make next of rear as new node and rear as new node.

dequeue(

- Check if queue is en not.
- If queue is empty th dequeue is not poss
- Else store front in te
- And make next of fr front.

Constraints:

0<size <100

0<data<1000

Input format:

First line indicates the size of the queue

Second line indicates the elements of the queue.

Output Format:

First line indicates the inserted elements in queue using linked list concept

second line indicates the one element dequeued from the queue using linked list concept.

Program Code:

```
#include <iostream>
using namespace std;
struct node
{
    int data;
    struct node* next;
};
struct node *front = NULL;
struct node *rear = NULL;
void enqueue(int d)
{
    struct node* new_n;
    new_n = (struct node*)malloc(sizeof(struct node));
    new_n->data = d;
    new_n->next = NULL;
    if(front==NULL&&rear==NULL){front=new_n;rear=new_n;}
    else{rear->next=new_n;rear=new_n;}
}
void dequeue()
{
    struct node* t;
    t=front;
    if(front==rear){front=NULL;rear=NULL;}
    else{front=front->next;}
    delete t;
}
void print_queue()
{
    struct node *p=front;
    do
    {
        cout<<p->data<<" ";
        p=p->next;
    }while(p!=NULL);
    cout<<"\n";
}
int main()
{
    int n,num;
    cin>>n;
    for(int i=0;i<n;i++)
    {
        cin>>num;
        enqueue(num);
    }
    print_queue();
    dequeue();
    print_queue();
    return 0;
}
```

Question 57

Question description

Lina is a Bachelor of Computer Applications (BCA) student. During her final year Campus Interview, she has an opportunity to get a job in a software company in Bangalore.

The company provides Five months training period with Rs.30000/month Package. Then it will be incremented to Rs.55000 per month.

At the end of the training, the examination was conducted for all freshers, Lina got a question paper and one of the questions comes under the concept of Queues in data structure that is efficiently implement k Queues in a single array .

Can you help?

Function Description

Following are the three arrays need to use:
1) **front[]**: This is of size k and stores indexes of front elements in all queues.
2) **rear[]**: This is of size k and stores indexes of rear elements in all queues.
2) **next[]**: This is of size n and stores indexes of next item for all items in array arr[].

Constraints

First '3' elements inserted in array 0, next '3' elements inserted into array1, remaining elements inserted into array2.

Input Format

First line indicates the number of elements to be inserted in the queues.

Second line indicates the elements.

Output Format

first line indicates the dequeue element of Q2

Second line indicates the dequeue element of Q1

Third line line indicates the dequeue element of Q0

Note: Refer sample input and output test cases

Program Code:

```
#include <stdio.h>
void f(){
    printf("kQueues(int k,int n) enqueue(int item,int qn) dequeue(int qn)");
}
int main()
{
    int n,i;
    scanf("%d",&n);
    int a[n];
    for(i=0;i<n;i++)
    scanf("%d",&a[i]);
    printf("Dequeue Q2:%d\nDequeue Q1:%d\nDequeue Q0:%d\n",a[6],a[3],a[0]);
    return 0;
}
```

Question 58

Question description

Given a string, you want to reorder its characters so that no two adjacent characters are the same. What is the lexicographically minimal such string?

Constraints

- $1 \leq n \leq 10^6$

Input

The only input line as a string of length n consisting of characters A–Z.

Output

Print the lexicographically minimal reordered string where no two adjacent characters are the same. If it is not possible to create such a string, print −1.

Program Code:

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

#define N 1000000
#define A 26

int main() {
static char cc[N + 1];

static int kk[A];
int n, i, p, a, b, c;

scanf("%s", cc);
n = strlen(cc);
for(i=0;i<n;i++) {
a = cc[i] - 'A';
kk[a]++;
}
for (a = 0; a < A; a++)
if (n < kk[a] * 2 - 1) {
printf("-1\n");
return 0;
}
p = -1;
for (i = 0; i < n; i++) {
a = 0;
while (a < A && (a == p || kk[a] == 0))
a++;
b = 0;
for (c = 1; c < A; c++)
if (kk[b] < kk[c])
b = c;
a = a != b && n - i - 1 < kk[b] * 2 - 1 ? b : a;
kk[a]--;
cc[i] = a + 'A';
p = a;
}
printf("%s\n", cc);
return 0;
}
```

Question 59

Problem Description

The Monk recently learnt about priority queues and requested his teacher for a fun challenge to solve. As a result, his teacher devised a simple task. He now possesses A, which is an integer array. He wishes to discover the product of the greatest, second largest, and third largest integers in the range [1,i] for each index i.

Note: Two numbers can be the same value-wise but they should be distinct index-wise.

Constraints:
1 <= **N** <= 100000
0 <= **A[i]** <= 1000000

Input:
The first line contains an integer **N**, denoting the number of elements in the array **A**.
The next line contains **N** space separated integers, each denoting the **ith** integer of the array **A**.

Output:
Print the answer for each index in each line. If there is no second largest or third largest number in the array **A** upto that index, then print "**-1**", without the quotes.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{

int n;cin>>n;
int arr[n];
for (int i = 0; i < n; i++) {
cin>>arr[i];
}
if(0) cout<<"if(biggest<big)if(a[i]>biggest)";
multiset<int> st;
for (int i = 0; i < n; i++) {
if(st.size()<2){
cout<<-1<<'\n';
st.insert(arr[i]);
}
else{
st.insert(arr[i]);
int ans = 1;
auto it = st.end();
it--;
ans*= *it;
it--;
ans*= *it;
it--;
ans*= *it;
cout<<ans<<'\n';
}

}

return 0;
}
```

Question 60

Problem description

Tina is a Bachelor of Computer Applications (BCA) student. During her final year Campus Interview, she has an opportunity to get a job in a software company in Bangalore.

The company provides Five months training period with Rs.30000/month Package. Then it will be incremented to Rs.55000 per month.

At the end of the training, the examination was conducted for all freshers, Tina got a question paper and one of the questions comes under the concept of Queue concept with Linked List Implementation

Function Description

The *front* points the first item of queue and *rear* points to last item.
enQueue() This operation adds a new node after *rear* and moves *rear* to the next node.
deQueue() This operation removes the front node and moves *front* to the next node.

Constraints

you have to perform N number of **enqueue** operation and two consecutive **dequeue** operation then continue M number of enqueue operation.

0<n , m <10000

Input Format

First line represents the N and M,

Second line represents the N represents the number of elements to be enqueued then

Third line indicates the M number of elements to be inserted after two consecutive dequeue.

Output Format

Results shows the every cycle enqueued/dequeued elements and front rear status.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
void don(){cout<<"struct QNode* newNode(int k)struct Queue* createQueue()void

int main()
{
int n,m;cin>>n>>m;
int arr[n+m];
for (int i = 0; i < n+m; i++) {
cin>>arr[i];
}
cout<<"Front:"<<arr[0]<<"\nRear:"<<arr[n-1]
<<"\nAfter 2 deQueue and M enqueue\n"
<<"Front is:"<<arr[2]<<"\nQueue Rear:"<<arr[n+m-1];

return 0;
}
```

Question 61

Question description

You are given a tree consisting of n nodes, and m paths in the tree.
Your task is to calculate for each node the number of paths containing that node.

Input

The first input line contains integers n and m: the number of nodes and paths. The nodes are numbered 1,2,…,n.
Then there are n−1 lines describing the edges. Each line contains two integers a and b: there is an edge between nodes a and b.
Finally, there are m lines describing the paths. Each line contains two integers a and b: there is a path between nodes a and b.

Output

Print n integers: for each node 1,2,…,n, the number of paths containing that node.

Constraints

- 1≤n,m≤2·10^5
- 1≤a,b≤n

Program Code:

```
#include <stdio.h>

#define N 200000
#define K 17 /* K = floor(log2(N)) */

struct L {
struct L *next;
int j;
} aa[N];

int dd[N], pp[N][K + 1], ll[N], rr[N], cc[N];

void link(int i, int j) {
static struct L ll_[N * 2], *l = ll_;

l->j = j;
l->next = aa[i].next; aa[i].next = l++;
```

```
}

void dfs(int p, int i, int d) {
struct L *l;
int k;

dd[i] = d;
pp[i][0] = p;
for (k = 1; 1 << k <= d; k++)
pp[i][k] = pp[pp[i][k - 1]][k - 1];
for (l = aa[i].next; l; l = l->next)
if (l->j != p)
dfs(i, l->j, d + 1);
}

int lca(int i, int j) {
int k;

if (dd[i] < dd[j])
return lca(j, i);
for (k = K; k >= 0; k--)
if (1 << k <= dd[i] - dd[j])
i = pp[i][k];
if (i == j)
return i;
for (k = K; k >= 0; k--)
if (1 << k <= dd[i] && pp[i][k] != pp[j][k])
i = pp[i][k], j = pp[j][k];
return pp[i][0];
}

int dfs2(int p, int i) {
struct L *l;
int c = cc[i];

for (l = aa[i].next; l; l = l->next)
if (l->j != p)
c += dfs2(i, l->j);
cc[i] = c += ll[i];
return c - rr[i];
}

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

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

for (h = 0; h < n - 1; h++) {
scanf("%d%d", &i, &j), i--, j--;
link(i, j), link(j, i);
}
dfs(-1, 0, 0);
while(m--) {
int i, j, a;
scanf("%d%d", &i, &j), i--, j--;
a = lca(i, j);
if (i == a) {
ll[j]++;
rr[i]++;
} else if (j == a) {
ll[i]++;
rr[j]++;
} else {
ll[i]++;
ll[j]++;
rr[a]++;
cc[a]--;
}
}
dfs2(-1, 0);
for (i = 0; i < n; i++)
printf("%d ", cc[i]);
printf("\n");
return 0;
}
```

Question 62

Question description

A company has n employees, who form a tree hierarchy where each employee has a boss, except for the general director.
Your task is to process q queries of the form: who is employee x's boss k levels higher up in the hierarchy?

Input

The first input line has two integers n and q: the number of employees and queries. The employees are numbered 1,2,…,n, and employee 1 is the general director.
The next line has n−1 integers e2,e3,…,en: for each employee 2,3,…,n their boss.
Finally, there are q lines describing the queries. Each line has two integers x and k: who is employee x's boss k levels higher up?

Output
Print the answer for each query. If such a boss does not exist, print −1.

Constraints

- 1≤n,q≤2·10^5
- 1≤ei≤i−1
- 1≤x≤n
- 1≤k≤n

Program Code:

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

```
#define ll long long
#define MAX 200005
#define pb push_back

vector<int>tree[MAX];
ll up[MAX][20];
void solve(){}
void link(int i,int j){
up[i][0]=j;
for(int m=1;m<20;m++){
if(up[i][m-1]!=-1)
up[i][m]=up[up[i][m-1]][m-1];
else
up[i][m]=-1;
}
for(auto child:tree[i]){
if(child!=j) link(child,i);
}
}
int ans_query(int src,int jump){
if(src==-1 or jump==0)return src;
for(int i=19;i>=0;i--){
if( jump>= (1<<i)){
return ans_query(up[src][i],jump-(1<<i));
}
}
return 1;
}
int main(){
solve();
int n,q;
cin>>n>>q;
for(int i=2;i<=n;i++){
int ee;
cin>>ee;

tree[i].pb(ee);
tree[ee].pb(i);
}
link(1,-1);
while(q--){
int node,jump;
cin>>node>>jump;
cout<<ans_query(node,jump)<<endl;
}
}
```

Question 63

Question description

You are given an n×n grid representing the map of a forest. Each square is either empty or contains a tree. The upper-left square has coordinates (1,1), and the lower-right square has coordinates (n,n).

Your task is to process q queries of the form: how many trees are inside a given rectangle in the forest?

Constraints

- 1≤n≤1000
- 1≤q≤2·10^5
- 1≤y1≤y2≤n
- 1≤x1≤x2≤n

Input

The first input line has two integers n and q: the size of the forest and the number of queries.
Then, there are n lines describing the forest. Each line has nn characters: . is an empty square and * is a tree.
Finally, there are q lines describing the queries. Each line has four integers y1, x1, y2, x2 corresponding to the corners of a rectangle.

Output
Print the number of trees inside each rectangle.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define rep(i,a,b) for (int i=a; i<b; ++i)
int dp[1005][1005];

int main(){
int n,m; cin>>n>>m;
rep(i,1,n+1){
rep(j,1,n+1){
char x; cin>>x;
dp[i][j] = (dp[i-1][j] - dp[i-1][j-1]) + dp[i][j-1] + (x=='*');
}
}
while(m--){
int y1 , x1, y2, x2; cin>>y1>>x1>>y2>>x2;
cout<<dp[y2][x2]+ dp[y1-1][x1-1] - dp[y2][x1-1] - dp[y1-1][x2]<<endl;
}
return 0;
cout<<"for(i=1;i<=n;i++)";
}
```

Question 66

Question description

A company has n employees, who form a tree hierarchy where each employee has a boss, except for the general director.
Your task is to process q queries of the form: who is the lowest common boss of employees a and b in the hierarchy?

Input

The first input line has two integers n and q: the number of employees and queries. The employees are numbered 1,2,…,n, and employee 1 is the general director.
The next line has n−1 integers e2,e3,…,en: for each employee 2,3,…,n their boss.
Finally, there are q lines describing the queries. Each line has two integers a and b: who is the lowest common boss of employees a and b?

Output
Print the answer for each query.

Constraints

- $1 \leq n,q \leq 2\cdot10^5$
- $1 \leq e_i \leq i-1$
- $1 \leq a,b \leq n$

Program Code:

```
#include <stdio.h>

#define N 200000
#define LN 17 /* LN = floor(log2(N - 1)) */

int main() {
static int dd[N], pp[LN + 1][N];
int n, q, p, i, j, k, tmp;

scanf("%d%d", &n, &q);
for (i = 1; i < n; i++) {
scanf("%d", &p), p--;
dd[i] = dd[p] + 1;
pp[0][i] = p;
}
for (k = 1; k <= LN; k++)
for (i = 0; i < n; i++)

pp[k][i] = pp[k - 1][pp[k - 1][i]];
while(q--) {
scanf("%d%d", &i, &j), i--, j--;
if (dd[i] < dd[j])
tmp = i, i = j, j = tmp;
if (dd[i] != dd[j])
for (k = LN; k >= 0; k--)
if (1 << k <= dd[i] - dd[j])
i = pp[k][i];
if (i != j) {
for (k = LN; k >= 0; k--)
if (1 << k <= dd[i] && pp[k][i] != pp[k][j]) {
i = pp[k][i];
j = pp[k][j];
}
i = pp[0][i];
}
printf("%d\n", i + 1);
}
return 0;
}
```

Question 69

Question description

You are given a list consisting of n integers. Your task is to remove elements from the list at given positions, and report the removed elements.

Constraints

- 1≤n≤2·10^5
- 1≤xi≤10^9
- 1≤pi≤n−i+1

Input

The first input line has an integer n: the initial size of the list. During the process, the elements are numbered 1,2,…,k where k is the current size of the list.
The second line has n integers x1,x2,…,xn: the contents of the list.
The last line has n integers p1,p2,…,pn: the positions of the elements to be removed.

Output

Print the elements in the order they are removed.

Program Code:

```
#include <stdio.h>
#define N 200000
#define N_ (1 << 18)
int tr[N_ * 2];
void build(int k,int l,int r) {
tr[k] = r - l;
if (r - l > 1) {
int m = (l + r) / 2;
build(k * 2 + 1, l, m);
build(k * 2 + 2, m, r);
}
}int query(int k, int l, int r, int x) {
int m, k1, k2;
tr[k]--;
if (r - l == 1)
return r;
m = (l + r) / 2, k1 = k * 2 + 1, k2 = k * 2 + 2;
return tr[k1] >= x ? query(k1, l, m, x) : query(k2, m, r, x - tr[k1]);
}
int main() {
```

```
int n, h, i, x;
scanf("%d", &n);
int aa[n];
for (i = 0; i < n; i++)
scanf("%d", &aa[i]);
build(0, 0, n);
for (h = 0; h < n; h++) {
scanf("%d", &x);
i = query(0, 0, n, x) - 1;
printf("%d ", aa[i]);
}
printf("\n");
return 0;
}
```

Question 70

Question description

The sam is enjoying a wonderful vacation in Byteland. What makes the sam impressed the most is the road system of the country. Byteland has N cities numbered 1 through N. City 1 is the capital of Byteland. The country also has N-1 bidirectional roads connecting the cities. The i-th road connects two different cities ui and vi. In this road system, people can travel between every pair of different cities by going through exactly one path of roads.

The roads are arranged in such a way that people can distinguish two cities only when both cities have different number of roads connected to it. Such two cities will be considered similar. For example, city A is similar to the capital if the number of roads connected to city A is equal to the number of roads connected to the capital.

On each day during the vacation, the sam wants to have a trip as follows. He chooses two cities A and B such that the sam will visit city B if he goes from A to the capital using the shortest path. Then, the sam will visit the cities on the shortest path from A to B through this path. Please note that A may be equal to B; that means the sam will enjoy the day in a single city.

The sam does not want to have similar trips. Two trips are considered similar if and only if
they both have the same number of visited cities and for each i, the i-th city visited in one trip is similar to the i-th city visited in the other trip.

The sam wants to have as many different, namely not similar, trips as possible. Help him count the maximum number of possible trips such that no two of them are similar.

Input Format
The first line of the input contains a single integer N. The i-th line of next N-1 lines contains two space-separated integers ui and vi, denoting the i-th road.

Output Format
Output a single line containing the maximum number of different trips.

Program Code:

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

```
struct state { int len,link; map<int,int> next;};
const int MAXL=200005;state st[MAXL];int sz,last;
void sa_init(){
st[0].len=0; st[0].link=-1; sz++; last = 0; }
void sa_extend(int c){
int cur=sz++; st[cur].len=st[last].len+1;
int p=last;
while(p!=-1 && !st[p].next.count(c)){
st[p].next[c]=cur; p=st[p].link; }
if(p == -1){ st[cur].link = 0;}
else{ int q=st[p].next[c];
if(st[p].len+1 == st[q].len){st[cur].link=q;}
else { int clone = sz++;
st[clone].len = st[p].len + 1;
st[clone].next = st[q].next;
st[clone].link = st[q].link;
while (p!=-1 && st[p].next[c]==q){
st[p].next[c]=clone; p=st[p].link; }
st[q].link = st[cur].link = clone;

}
} last = cur; }
/*void build(string &x){ sz=0;*/
//for(ll i=0;i<3*x.length()+15;i++){
//st[i].next.clear();st[i].len=0; st[i].link=0;}
//sa_init();
/*for(ll i=0;i<x.size();i++)sa_extend(x[i]); }*/
const int N = 1e5 + 100;
vector<int> G[N];
int deg[N];
void dfs(int s, int p){
sa_extend(deg[s]);
int tmp = last;
for(auto it : G[s]){
if(it == p) continue;
dfs(it , s);
last = tmp;
}
}
ll dp[MAXL];
int main(){
ios_base::sync_with_stdio(false);
cout.tie(0); cin.tie(0);
sa_init();
int n; cin >> n;
int u , v;
for(int i = 0;i < n-1 ; ++i){
cin >> u >> v;
G[u].push_back(v);
G[v].push_back(u);
++deg[u];

++deg[v];
}
dfs(1 , -1);
vector<pair<int,int> > topo(sz);
for(int i = 0;i < sz ; ++i) topo[i] = make_pair(st[i].len , i);
sort(topo.begin() , topo.end());
for(int i = sz-1;i >= 0; --i){
u = topo[i].second;
dp[u] = 1;
for(auto it : st[u].next){
dp[u] += dp[it.second];
}
}
cout << dp[0]-1 << endl;

return 0;
}
```

Question 72

Question Description

In a movie festival, n movies will be shown. You know the starting and ending time of each movie.

Your task is to process q queries of the form: if you arrive and leave the festival at specific times, what is the maximum number of movies you can watch?

You can watch two movies if the first movie ends before or exactly when the second movie starts. You can start the first movie exactly when you arrive and leave exactly when the last movie ends.

Input

The first input line has two integers n and q: the number of movies and queries.

After this, there are n lines describing the movies. Each line has two integers a and b: the starting and ending time of a movie.

Finally, there are q lines describing the queries. Each line has two integers a and b: your arrival and leaving time.

Output

Print the maximum number of movies for each query.

Constraints

- $1 \le n,q \le 2 \cdot 10^5$
- $1 \le a < b \le 10^6$

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int dp[1000006][25];
void solve(){}
int main(){
```

```
solve();
int n, q; cin>>n>>q;
for (int i = 0; i < n; i++) {
int x, y; cin>>x>>y;
dp[y][0] = max(dp[y][0], x);
}
for (int i = 1; i <= 1000000; i++)
dp[i][0] = max(dp[i][0], dp[i-1][0]);

for (int k = 1; k <= 20; k++)
for (int i = 1; i <= 1000000; i++)
dp[i][k] = dp[dp[i][k-1]][k-1];

while(q--) {
int x,y; cin>>x>>y;
int ans = 0;
while(y>0) {
int z = 0;
for (int i = 0; i <= 20; i++) {
if (dp[y][i] < x) {
z = i;
break;
}
}
if (z == 0)
break;
ans += (1<<(z-1));
y = dp[y][z-1];
}
cout<<ans<<endl;
}

}
```

Question 73

Problem Description:

Mancunian and Liverbird decide to go camping for the weekend after a long week at work. They came upon an unusual tree with N nodes while walking through a forest. From 1 to N, the vertices are numbered.

A colour is allocated to each node in the tree (out of C possible colors). They decide to work together (for a change) and put their reasoning abilities to the test because they are bored. At vertex 1, the tree is rooted. They aim to locate the nearest ancestor with the same hue for each node.

Constraints

- $1 \leq N \leq 100{,}000$
- $1 \leq C \leq 100{,}000$

Input format

The first line contains two integers *N* and *C* denoting the number of vertices in the tree and the number of possible colors.
The second line contains N−1 integers. The i th integer denotes the parent of the i+1 th vertex.
The third line contains *N* integers, denoting the colors of the vertices. Each color lies between *1* and *C* inclusive.

Output format

Print *N* space-separated integers. The ith integer is the vertex number of lowest ancestor of the ith node which has the same color. If there is no such ancestor, print *1* for that node.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main() {
int n,i,c;
scanf("%d %d", &n, &c);
int tree[n+1][2];
tree[1][0] = -1;
for(i=2;i<=n;i++) {
scanf("%d", &tree[i][0]);
}
for(i = 1; i <= n; i++) {
scanf("%d", &tree[i][1]);
}
int parent;
for(i = 1; i<= n; i++) {
parent = tree[i][0];
while(parent != -1 && tree[parent][1] != tree[i][1]) {

parent = tree[parent][0];
}
printf("%d ", parent);
}
return 0;
}
```

Question 74

Problem Description

You are given a weighted graph with **N** vertices and **M** edges. Find the total weight of its **maximum** spanning tree.

Constraints

- **1 <= T <= 20**
- **1 <= N <= 5000**
- **1 <= M <= 100 000**
- **1 <= c <= 10 000**

Input

The first line contains one integer **T** denoting the number of test cases. Each test case starts with a line containing **2** space-separated integer: **N** and **M**. Each of the following **M** lines contain description of one edge: three different space-separated integers: **a**, **b** and **c**. **a** and **b** are different and from 1 to N each and denote numbers of vertices that are connected by this edge. **c** denotes weight of this edge.

Output

For each test case output one integer - the total weight of its **maximum** spanning tree.

Program Code:

```
#include<bits/stdc++.h>
typedef long long ll;
using namespace std;
struct edge
{
int u;
int v;
int w;
};
edge a[100005];
int parent[100005];

bool comp (edge a , edge b)
{
return a.w>b.w;
}
int find_parent(int u) ///DSU find
{
/* return (parent[u]==u) ? u: find_parent(parent[u]);*/
if(parent[u]==u)
return u;
else
return parent[u]=find_parent(parent[u]);
}
void merge(int u, int v) /// DSU union
{
parent[u]=v;
}
int main()
{
int t;
cin>>t;
while(t--) {
int n,m;
cin>>n>>m;
for(int i=1;i<=n;i++)
parent[i]=i;
for(int i=0;i<m;i++) {
cin>>a[i].u>> a[i].v >> a[i].w;
}
sort(a,a+m,comp);
ll ans=0;
for(int i=0;i<m;i++) {

int x=find_parent(a[i].u);
int y=find_parent(a[i].v);
if(x!=y)
{
merge(x,y);
ans+=a[i].w;
}
}
cout<<ans<<endl;
}
return 0;
cout<<"int printheap(int N)";
}
```

Question 76

Problem description

When the king of ghosts notices that all humans on Planet Earth have lost their dread of the ghost race, he is extremely unhappy. He understands why this is happening. The existing ghost species has gotten incredibly lethargic and has ceased visiting Planet Earth in order to terrorise humanity. As a result, he plans to arrange a tournament to urge the whole ghost race to scare the humans. The monarch, on the other hand, never comes to Planet Earth.

This competition will go on for **N days**. Currently, there are a total of **M ghosts** (apart from the king) existing in the ghost race such that :
- The youngest ghost is 1 year old.
- The oldest ghost is M years old.
- No two ghosts have the same age.
- The age of each and every ghost is a positive integer.

Every day of the tournament, ghosts must visit Planet Earth in order to frighten humans. The ghost that terrifies the most humans on any given day is named "Ghost of the Day" at the conclusion of the day. The king of ghosts, on the other hand, is a firm believer in constancy. After this title is granted, a "Consistency Trophy" is offered to the ghost who has earned the most of these titles up until that point. If there are several such ghosts, the prize is handed to the eldest among them. It's worth noting that "Title Giving" and "Trophy Giving" take place at the end of each tournament day.

Each day of the competition, you will be told the age of the ghost that earned the "Ghost of the Day" title. Your task is to determine the age of the ghost who received the "Consistency Trophy" on each competition day.

Input
The first line consists of 2 space separated integers **N** and **M**. The next line consists of **N** space separated integers such that the **i**th integer denotes the age of the ghost who was awarded with the **"Ghost of the Day" title** on the **i**th day of the competition.

Output
Print **N** lines. The **i**th line should contain 2 space separated integers such that the first integer denotes the **age of the ghost** who was awarded with the **"Consistency Trophy"** on the **i**th day and the second integer

denotes the number of **"Ghost of the Day" titles** won by this ghost until the end of the **i**th day of the competition.

Constraints
$1 \leq \mathbf{N} \leq 10^5$
$1 \leq \mathbf{M} \leq 10^9$

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
int i,n,m;
cin>>n>>m;
unordered_map<int,int> ghost;

int best = -1;
cin>>best;
ghost[best]+=1;
cout<<best<<" "<<1<<endl;
for(i=0;i<n-1;i++)
{
int y;
cin>>y;
ghost[y]+=1;
if((ghost[y]>ghost[best]) || (ghost[y]==ghost[best] && y>best)) best = y;
cout<<best<<" "<<ghost[best]<<"\n";
}
}
```

Question 77

Question description

A beautiful *code* of a tree of n nodes is a sequence of n−2 integers that uniquely specifies the structure of the tree.

The code is constructed as follows: As long as there are at least three nodes left, find a leaf with the smallest label, add the label of its only neighbour to the code, and remove the leaf from the tree.

Given a beautiful code of a tree, your task is to construct the original tree.

Constraints

- 3≤n≤2⋅10^5
- 1≤a,b≤n

Input

The first input line contains an integer n: the number of nodes. The nodes are numbered 1,2,…,n.

The second line contains n−2 integers: the beautiful code.

Output

Print n−1 lines describing the edges of the tree. Each line has to contain two integers a and b: there is an edge between nodes a and b. You can print the edges in any order.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define f(i,a,n) for(int i=a;i<n;i++)
#define X(a,b) if(a==b)
vector< int > vi;
const int maxN = 2e5+5;
int N, a[maxN], deg[maxN];
priority_queue<int, vector<int>, greater<int>> q;

int main(){
scanf("%d", &N);
fill(deg+1, deg+N+1, 1);
//for(int i = 0; i < N-2; i++)
f(i,0,N-2){
scanf("%d", &a[i]);
deg[a[i]]++;
}
//for(int i = 1; i <= N; i++)
f(i,1,N+1)
//if(deg[i] == 1)
X(deg[i],1)
q.push(i);
f(i,0,N-2){
int u = a[i];
int v = q.top(); q.pop();
deg[u]--; deg[v]--;
//if(deg[u] == 1)
X(deg[u],1)
q.push(u);
printf("%d %d\n", v, u);
}
//for(int i = 1; i <= N; i++)
f(i,1,N+1)
if(deg[i])

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

Question 78

Problem Description

A new species is trying to rule the planet. This species is creating their own population outburst to dominate other species. It all started with 1 single member of the species. The population increases in treelike fashion abiding by few rules as listed below.

- Single member is able to reproduce by itself.
- A new member is added to the population every minute.
- Every member is associated with integral name.
- Multiple members can share a common name.
- Every member has it's own reproduction capacity, that is maximum number of children it can reproduce.
- A member can start to reproduce only if all members older than it have exhausted their reproduction capacity.
- Level 0 in family tree of this species comprise of single member at the start of multiplication.
- Integral name of single member at the start is 0.
- The population grows level wise, where number of members at level *i* is dependent on reproduction capacity of members at prior level.

Given the integral name of new member and it's reproduction capacity that is added to the population, you have to find it's parent, level at which it is added and it's ascending age wise rank among siblings.

Input:
First line of the input contains 2 integers, N,RC0, representing number of minutes we will be examining the population increase and reproduction capacity of member at epoch. Next *N* line contains 2 integers each, IDi,RCi, representing integral name and reproduction capacity of new member born at time *i*.

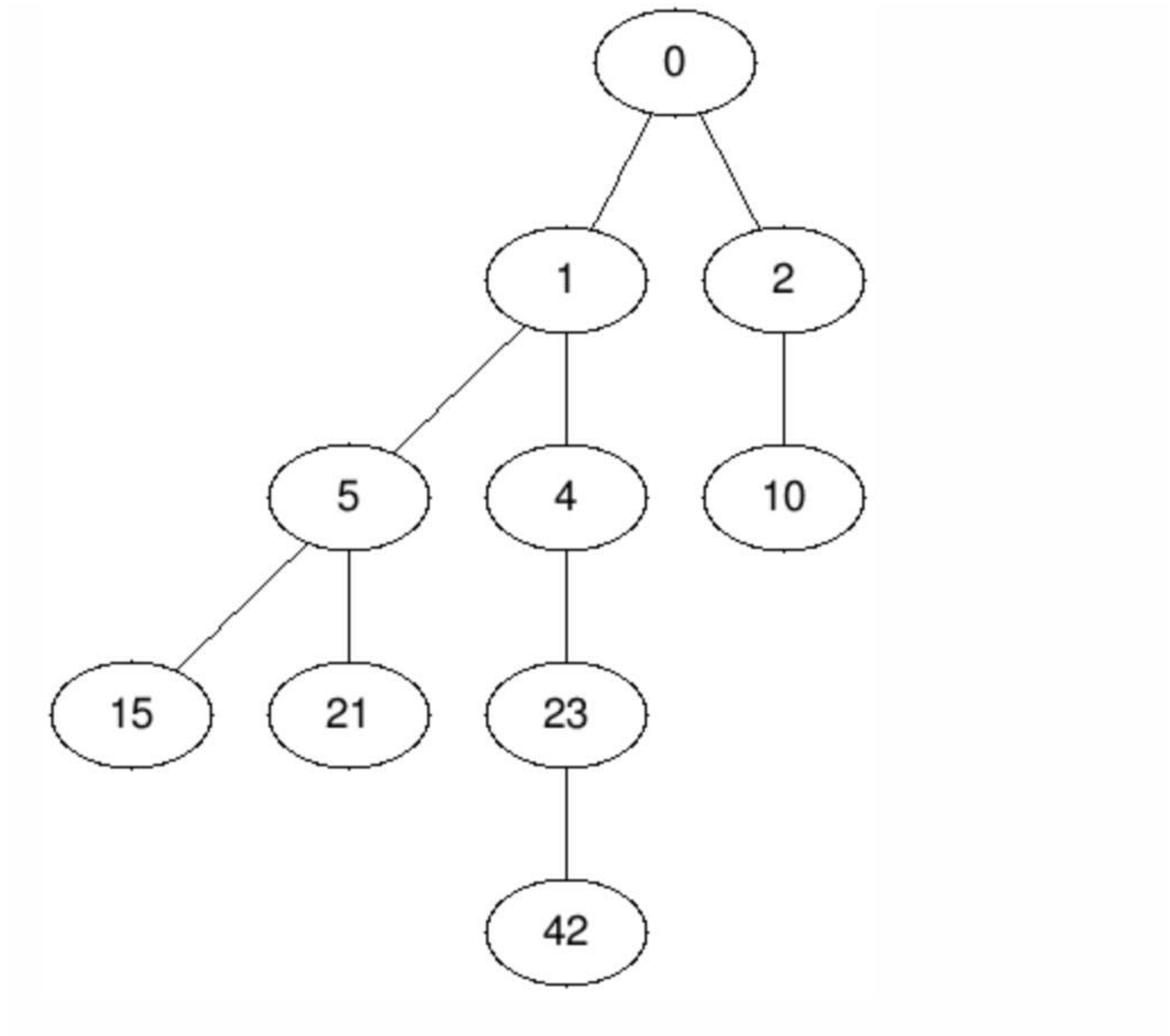
Output:
N lines, each line containing 3 integers, P,L,C, representing integral name of the parent, level at which it is added and it's ascending age wise rank among siblings.

Note :
It will always be possible to reproduce a new child or in other words, through out the given time, there exists atleast one member which can still accomodate new child.

Constraints:
1≤N≤106
−109≤IDi≤109
0≤RCi≤109

Explanation for test case1

The resultant family tree looks like this.



Program Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct cell{
int name;
int level;
int capacity;
};
struct cell queue[1000001];
struct cell arr[1000001];
int front;
int end;
void init_queue(){
front = 0;
end = 0;
}
void enqueue(int name,int capacity,int level){

queue[end].name = name;
queue[end].level = level;
queue[end].capacity = capacity;
end = end + 1;
}
int is_empty(){
if(end == front)
return 1;
return 0;
}
void dequeue()
{
if(!is_empty())
front++;
}
int main(){
int n,rc;
init_queue();
scanf("%d %d",&n,&rc);
int i,j,k;
for(i=0;i<n;i++){
scanf("%d %d",&arr[i].name,&arr[i].capacity);
}
enqueue(0,rc,0);
i=0;
while(!is_empty()){
int par = queue[front].name;
int cap = queue[front].capacity;
int lev = queue[front].level+1;
k=1;
for(j=0;j<cap&&i<n;j++,i++){

printf("%d %d %d\n",par,lev,k++);
enqueue(arr[i].name,arr[i].capacity,lev);
}
dequeue();
}
return 0;
}
```

Question 79

Question description

Given two rooted trees, your task is to find out if they are *isomorphic*, i.e., it is possible to draw them so that they look the same.

Constraints

- 1≤t≤1000
- 2≤n≤10^5
- the sum of all values of n is at most 10^5

Input

The first input line has an integer t: the number of tests. Then, there are t tests described as follows:

The first line has an integer n: the number of nodes in both trees. The nodes are numbered 1,2,…,n, and node 1 is the root.
Then, there are n−1 lines describing the edges of the first tree, and finally n−1 lines describing the edges of the second tree.

Output

For each test, print "YES", if the trees are isomorphic, and "NO" otherwise.

Program Code:

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

#define rep(i, a, b) for(int i = a; i < (b); ++i)
#define trav(a, x) for(auto& a : x)

#define all(x) begin(x), end(x)
#define sz(x) (int)(x).size()
typedef long long ll;
typedef pair<int, int> pii;
typedef vector<int> vi;

vi h, id;
vector<vi> g;
map<int, vi> lvl;

void dfs(int i, int p) {
trav(j, g[i]) if (j!=p) {
dfs(j, i);
h[i]=max(h[i], h[j]+1);
}
lvl[h[i]].push_back(i);
}

int main() {
cin.sync_with_stdio(0); cin.tie(0);
cin.exceptions(cin.failbit);

int t;
cin >> t;
while(t--) {
int n;
cin >> n;
int m=2*n+1;
g.assign(m, vi());
rep(i, 0, 2) {
rep(j, 0, n-1) {

int a, b;
cin >> a >> b;
a+=i*n, b+=i*n;
g[a].push_back(b);
g[b].push_back(a);
}
}
g[0]={1, n+1};
h.assign(m, 0);
id.assign(m, 0);
lvl.clear();
dfs(0, -1);
if (h[1]!=h[n+1]) {
cout << "NO\n";
continue;
}
trav(l, lvl) {
map<vector<ll>, int> u;
trav(i, l.second) {
vector<ll> cur;
trav(j, g[i]) cur.push_back(3LL*n*h[j]+id[j]);
sort(all(cur));
if (!u.count(cur)) {
int s=sz(u);
u[cur]=s;
}
id[i]=u[cur];
}
}
cout << (id[1]==id[n+1]? "YES\n":"NO\n");
}

return 0;
}
```

Question 80

Question Description:

There are n cities and m flight connections between them. Your task is to add new flights so that it will be possible to travel from any city to any other city. What is the minimum number of new flights required?

Input

The first input line has two integers n and m: the number of cities and flights. The cities are numbered 1,2,…,n.

After this, there are m lines describing the flights. Each line has two integers a and b: there is a flight from city a to city b. All flights are one-way flights.

Output

First print an integer k: the required number of new flights. After this, print k lines describing the new flights. You can print any valid solution.

Constraints

- 1≤n≤10^5
- 1≤m≤2·10^5
- 1≤a,b≤n

Program Code:

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

#define N 100000
#define M 200000

struct L {
struct L *next;
int j;
} aa[N], bb[N], aa_[N];

void link(int i, int j) {
static struct L l91[M * 2], *l = l91;

l->j = j;
l->next = aa[i].next, aa[i].next = l++;
l->j = i;
l->next = bb[j].next, bb[j].next = l++;
}

void link_(int i, int j) {
static struct L l91[M], *l = l91;

l->j = j;
l->next = aa_[i].next, aa_[i].next = l++;
}

int po[N], npo;
char visited[N];

void dfs1(int i) {
struct L *l;

if (visited[i])
return;
visited[i] = 1;
for (l = aa[i].next; l; l = l->next)
dfs1(l->j);
po[npo++] = i;
}

int cc[N], dd[N];

void dfs2(int j, int c) {
```

```
struct L *l;
int c_ = cc[j];

if (c_ != -1) {
if (c_ != c) {
link_(c_, c);
dd[c]++;
}
return;
}
cc[j] = c;
for (l = bb[j].next; l; l = l->next)
dfs2(l->j, c);
}

int dfs3(int i) {

struct L *l;

if (visited[i])
return -1;
visited[i] = 1;
if (!aa_[i].next)
return i;
for (l = aa_[i].next; l; l = l->next) {
int w = dfs3(l->j);

if (w != -1)
return w;
}
return -1;
}

void add(int i, int j) {
printf("%d %d\n", i + 1, j + 1);
}

void augment(int n) {
static int vv[N], ww[N];
static char iv[N], iw[N];
int h, i, p, q, s, t, x;

p = 0;
for (i = 0; i < n; i++) {
if (cc[i] != i)
continue;
if (dd[i] == 0) {
int w = dfs3(i);

if (w != -1) {
iv[vv[p] = i] = 1;
iw[ww[p] = w] = 1;
p++;
}
}
}
s = t = p;
for (i = 0; i < n; i++) {
if (cc[i] != i)
continue;
if (!iv[i] && dd[i] == 0)
vv[s++] = i;
if (!iw[i] && !aa_[i].next)
ww[t++] = i;
}
printf("%d\n", s > t ? s : t);
for (h = 0; h < p - 1; h++)
add(ww[h], vv[h + 1]);
q = s < t ? s : t;
for (h = p; h < q; h++)
add(ww[h], vv[h]);
x = ww[p - 1];
for (h = q; h < s; h++)
add(x, vv[h]), x = vv[h];
for (h = q; h < t; h++)
add(x, ww[h]), x = ww[h];
add(x, vv[0]);
}

int main() {
int n, m, h, i, j, k;

scanf("%d%d", &n, &m);
while(m--) {
scanf("%d%d", &i, &j), i--, j--;
link(i, j);
}
for (i = 0; i < n; i++)
dfs1(i);
memset(cc, -1, n * sizeof *cc);
k = 0;
for (h = n - 1; h >= 0; h--) {
j = po[h];
if (cc[j] == -1) {
dfs2(j, j);
k++;
}
}
if (k == 1) {
printf("0\n");
return 0;
}
memset(visited, 0, n * sizeof *visited);
augment(n);
return 0;
}
```

Question 81

Question description

You are playing a game consisting of n planets. Each planet has a teleporter to another planet (or the planet itself).
You start on a planet and then travel through teleporters until you reach a planet that you have already visited before.
Your task is to calculate for each planet the number of teleportations there would be if you started on that planet.

Input

The first input line has an integer n: the number of planets. The planets are numbered 1,2,…,n.

The second line has n integers t1,t2,…,tn: for each planet, the destination of the teleporter. It is possible that ti=i.

Output

Print n integers according to the problem statement.

Constraints

- 1≤n≤2·10^5
- 1≤ ti ≤n

Program Code:

```
#include <stdio.h>
#include <string.h>
#define N 200000
int main() {
static int aa[N], cc[N], dd[N], qq[N];
int n, i, j, c, d, q, cnt;
scanf("%d", &n);

for (i = 0; i < n; i++)
scanf("%d", &aa[i]), aa[i]--;
memset(cc, -1, n * sizeof *cc);
cnt = 0;
for(i = 0;i<n;i++) {
if (cc[i] != -1)
continue;
d = 0;
j = i;
while (cc[j] == -1) {
cc[j] = -2;
d++;
j = aa[j];
}
if (cc[j] == -2) {
c = cnt++;
q = 0;
while (cc[j] == -2) {
cc[j] = c;
q++;
j = aa[j];
}
qq[c] = q;
d -= q;
} else {
c = cc[j];
d += dd[j];
}
j = i;
while (cc[j] == -2) {
cc[j] = c;

dd[j] = d--;
j = aa[j];
}
}
for (i = 0; i < n; i++)
printf("%d ", dd[i] + qq[cc[i]]);
printf("\n");
return 0;
}
```

Question 82

Question description

There are n cities and initially no roads between them. However, every day a new road will be constructed, and there will be a total of m roads.

A component is a group of cities where there is a route between any two cities using the roads. After each day, your task is to find the number of components and the size of the largest component.

Input

The first input line has two integers n and m: the number of cities and roads. The cities are numbered 1,2,…,n.

Then, there are m lines describing the new roads. Each line has two integers a and b: a new road is constructed between cities a and b.

You may assume that every road will be constructed between two different cities.

Output

Print m lines: the required information after each day.
Constraints

- $1 \le n \le 10^5$
- $1 \le m \le 2 \cdot 10^5$
- $1 \le a,b \le n$

Program Code:

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

#define N 100000

int dsu[N];

int find(int i) {
return dsu[i] < 0 ? i : (dsu[i] = find(dsu[i]));
}

int join(int i,int j) {
int tmp;

i = find(i);
j = find(j);
if (i == j)
return 0;
if (dsu[i] < dsu[j])
tmp = i, i = j, j = tmp;
dsu[j] += dsu[i];
dsu[i] = j;
return -dsu[j];
}

int main() {
int n, m, i, j, c, c_;

scanf("%d%d", &n, &m);
memset(dsu, -1, n * sizeof *dsu);
c_ = 1;
while (m--) {
scanf("%d%d", &i, &j), i--, j--;
c = join(i, j);
if (c != 0) {
n--;
if (c_ < c)
c_ = c;
}
printf("%d %d\n", n, c_);
}
return 0;

}
```

Question 83

Question description

There are n cities and flight connections between them. You want to travel from Chennai to Ladakh so that you visit each city exactly once. How many possible routes are there?

Input

The first input line has two integers n and m: the number of cities and flights. The cities are numbered 1,2,…,n. City 1 is Chennai and city n is Ladakh.

Then, there are m lines describing the flights. Each line has two integers a and b: there is a flight from the city a to city b. All flights are one-way flights.

Output

Print one integer: the number of routes modulo 10^9+7.

Constraints

- 2≤n≤20
- 1≤m≤n2
- 1≤a,b≤n

Program Code:

```
#include <stdio.h>

#define N 20
#define MD 1000000007

struct L {
struct L *next;
int j;
} aa[N];

void link(int i,int j) {
static struct L l91[N * N], *l = l91;

l->j = j;
l->next = aa[i].next; aa[i].next = l++;
}

int main() {
static int dp[1 << N][N];
int n, m, i, j, b, b_, x;
struct L *l;

scanf("%d%d", &n, &m);
while (m--) {
scanf("%d%d", &i, &j), i--, j--;
link(i, j);
}
dp[1 << 0][0] = 1;
for (b = 1; b < 1 << n; b += 2)
for (i = 0; i < n - 1; i++) {
x = dp[b][i];
if (x == 0)
continue;
for (l = aa[i].next; l; l = l->next)
if (!(b & 1 << (j = l->j))) {

b_ = b | 1 << j;
dp[b_][j] = (dp[b_][j] + x) % MD;
}
}
printf("%d\n", dp[(1 << n) - 1][n - 1]);
return 0;
}
```

Question 84

Question description

Your task is to deliver mail to the inhabitants of a city. For this reason, you want to find a route whose starting and ending point are the post office, and that goes through every street exactly once.

Input

The first input line has two integers n and m: the number of crossings and streets. The crossings are numbered 1,2,…,n, and the post office is located at crossing 1.

After that, there are m lines describing the streets. Each line has two integers a and b: there is a street between crossings a and b. All streets are two-way streets.

Every street is between two different crossings, and there is at most one street between two crossings.

Output

Print all the crossings on the route in the order you will visit them. You can print any valid solution.

If there are no solutions, print "IMPOSSIBLE".

Constraints

- 2≤n≤10^5
- 1≤m≤2·10^5
- 1≤a,b≤n

Program Code:

```
#include <stdio.h>

#define N 100000
#define M 200000

struct L {
struct L *next;
int h;
} *aa[N];

int ij[M + 1];
char lazy[M + 1];
```

```
struct L *new_L(int h) {
static struct L l91[M * 2 + 1 + M], *l = l91;

l->h = h;

return l++;
}

void link(int i,int h) {
struct L *l = new_L(h);

l->next = aa[i]; aa[i] = l;
}

void hierholzer(struct L *e, int i) {
struct L *f = e->next, *l;

while ((l = aa[i])) {
int h = l->h;

if (lazy[h])
aa[i] = l->next;
else {
lazy[h] = 1;
e = e->next = new_L(h);
i ^= ij[h];
}
}
e->next = f;
}

int main() {
static int dd[N];
struct L *e_, *e;
int n, m, h, i, j;

scanf("%d%d", &n, &m);
for (h = 1; h <= m; h++) {
scanf("%d%d", &i, &j), i--, j--;
ij[h] = i ^ j;
link(i, h), link(j, h);
dd[i]++, dd[j]++;
}
for (i = 0; i < n; i++)
if (dd[i] % 2) {
printf("IMPOSSIBLE\n");
return 0;
}
e_ = new_L(0);
i = 0;
m++;
for (e = e_; e; e = e->next) {
i ^= ij[e->h];
hierholzer(e, i);
m--;
}
if (m != 0) {
printf("IMPOSSIBLE\n");
return 0;
}
i = 0;
for (e = e_; e; e = e->next) {
i ^= ij[e->h];
printf("%d ", i + 1);
}
printf("\n");
return 0;

}
```

Question 86

Question description

A game consists of n rooms and m teleporters. At the beginning of each day, you start in room 1 and you have to reach room n.
You can use each teleporter at most once during the game. How many days can you play if you choose your routes optimally?

Constraints

2 ≤ n ≤ 500

1 ≤m ≤1000

1 ≤ a, b ≤n

Input
The first input line has two integers n and m: the number of rooms and teleporters. The rooms are numbered 1,2,…,n.
After this, there are m lines describing the teleporters. Each line has two integers a and b: there is a teleporter from room a to room b.
There are no two teleporters whose starting and ending room are the same.

Output
First print an integer k: the maximum number of days you can play the game. Then, print k route descriptions according to the example. You can print any valid solution.

Program Code:

```
#include <stdio.h>
#define N 500
#define M 1000
struct L {
struct L *next;
int h;
} aa[N * 2];
int ij[M + N], cc[(M + N) * 2], dd[N * 2];
int bfs(int n,int s,int t) {
static int qq[N * 2];

int head, cnt, h, i, j, d;
for (i = 0; i < n; i++)
dd[i] = n;
dd[s] = 0;
head = cnt = 0;
qq[head + cnt++] = s;
while (cnt) {
struct L *l;
i = qq[cnt--, head++];
d = dd[i] + 1;
for (l = aa[i].next; l; l = l->next)
if (cc[h = l->h]) {
j = i ^ ij[h >> 1];
if (dd[j] == n) {
dd[j] = d;
if (j == t)
return 1;
qq[head + cnt++] = j;
}
}
}
return 0;
}
int dfs(int n, int i, int t) {
struct L *l;
int h, j, d;
if (i == t)
return 1;
d = dd[i] + 1;
for (l = aa[i].next; l; l = l->next)
if (cc[h = l->h]) {

j = i ^ ij[h >> 1];
if (dd[j] == d && dfs(n, j, t)) {
cc[h]--, cc[h ^ 1]++;
return 1;
}
}
dd[i] = n;
return 0;
}
int dinic(int n, int s, int t) {
int f = 0;
while (bfs(n, s, t))
while (dfs(n, s, t))
f++;
return f;
}
void link(int i, int j, int h, int c) {
static struct L l91[(M + N) * 2], *l = l91;
ij[h] = i ^ j;
cc[h << 1] = c;
l->h = h << 1;
l->next = aa[i].next, aa[i].next = l++;
l->h = h << 1 ^ 1;
l->next = aa[j].next, aa[j].next = l++;
}
int qq[N];
int path(int i, int t) {
int cnt = 0;
while (i != t) {
struct L *l;
int h;

qq[cnt++] = i;
for (l = aa[i].next; l; l = l->next)
if (((h = l->h) & 1) == 0 && cc[h ^ 1]) {
cc[h]++, cc[h ^ 1]--;
i ^= ij[h >> 1];
break;
}
}
qq[cnt++] = t;
return cnt;
}
int main() {
int n, m, h, i, j, k, s, t, cnt;
scanf("%d%d", &n, &m);
for (h = 0; h < m; h++) {
scanf("%d%d", &i, &j), i--, j--;
link(i << 1 ^ 1, j << 1, h, 1);
}
for (i = 0; i < n; i++)
link(i << 1, i << 1 ^ 1, m + i, n);
s = 0, t = (n - 1) << 1 ^ 1;
k = dinic(n * 2, s, t);
printf("%d\n", k);
while (k--) {
cnt = path(s, t);
printf("%d\n", cnt / 2);
for (i = 0; i < cnt; i += 2)
printf("%d ", (qq[i] >> 1) + 1);
printf("\n");
}
return 0;

}
```

Question 87

Question description

You are given a directed graph, and your task is to find out if it contains a negative cycle, and also give an example of such a cycle.

Constraints

- 1≤n≤2500
- 1≤m≤5000
- 1≤a,b≤n
- −10^9≤c≤10^9

Input Format
The first input line has two integers n and m: the number of nodes and edges. The nodes are numbered 1,2,…,n.
After this, the input has m lines describing the edges. Each line has three integers a, b, and c: there is an edge from node a to node b whose length is c.

Output Format
If the graph contains a negative cycle, print first "YES", and then the nodes in the cycle in their correct order. If there are several negative cycles, you can print any of them. If there are no negative cycles, print "NO".

Program Code:

```
#include <stdio.h>

#define N    2500
#define M    5000

int main() {
    static int aa[M], bb[M], cc[M], pp[N], ii[1 + N];
    static char used[N];
    static long long dd[N];
    int n, m, h, r, a, b, c, k;
    long long d;

    scanf("%d%d", &n, &m);
    for (h = 0; h < m; h++)
        scanf("%d%d%d", &aa[h], &bb[h], &cc[h]), aa[h]--, bb[h]--;
    for (r = 0; r < n; r++)
        for (h = 0; h < m; h++) {
            a = aa[h], b = bb[h], c = cc[h];
            d = dd[a] + c;
            if (dd[b] > d) {
                dd[b] = d;
                pp[b] = a;
                if (r == n - 1) {
                    while (!used[b]) {
                        used[b] = 1;
                        b = pp[b];
                    }
                    k = 0;
                    while (used[b]) {
                        used[b] = 0;
                        ii[k++] = b;
                        b = pp[b];
                    }
                    ii[k++] = b;
                    printf("YES\n");
                    while(k--)
                        printf("%d ", ii[k] + 1);
                    printf("\n");
                    return 0;
                }
            }
        }
    printf("NO\n");
    return 0;
}
```

Question 88

Question description

Kaaleppi has just robbed a bank and is now heading to the harbor. However, the police wants to stop him by closing some streets of the city.

What is the minimum number of streets that should be closed so that there is no route between the bank and the harbor?

Constraints

- 2≤n≤500
- 1≤m≤1000
- 1≤a,b≤n

Input

The first input line has two integers n and m: the number of crossings and streets. The crossings are numbered 1,2,…,n. The bank is located at crossing 1, and the harbor is located at crossing n.

After this, there are m lines that describe the streets. Each line has two integers and b: there is a street between crossings a and b. All streets are two-way streets, and there is at most one street between two crossings.

Output

First print an integer k: the minimum number of streets that should be closed. After this, print k lines describing the streets. You can print any valid solution.

Program Code:

```
#include <stdio.h>

#define N 500
#define M 1000

struct L {
```

```
struct L *next;
int h;
} aa[N];

int ij[M], cc[M * 4];
int dd[N];

void link(int i,int h) {
static struct L l91[M * 4], *l = l91;

l->h = h;
l->next = aa[i].next; aa[i].next = l++;
}

int bfs(int n,int s,int t) {
static int qq[N];
int h, i, j, head, cnt, d;

for (i = 0; i < n; i++)
dd[i] = n;
dd[s] = 0;
head = cnt = 0;
qq[head + cnt++] = s;
while (cnt) {
struct L *l;

i = qq[cnt--, head++];
d = dd[i] + 1;
for (l = aa[i].next; l; l = l->next)
if (cc[h = l->h]) {
j = i ^ ij[h >> 2];
if (dd[j] == n) {
dd[j] = d;
if (j == t)
return 1;
qq[head + cnt++] = j;
}
}
}
return 0;
}

int dfs(int n, int i, int t) {
struct L *l;
int h, j, d;

if (i == t)
return 1;
d = dd[i] + 1;

for (l = aa[i].next; l; l = l->next)
if (cc[h = l->h]) {
j = i ^ ij[h >> 2];
if (dd[j] == d && dfs(n, j, t)) {
```

```
cc[h]--, cc[h ^ 1]++;
return 1;
}
}
dd[i] = n;
return 0;
}

int dinic(int n, int s, int t) {
int f = 0;

while (bfs(n, s, t))
while (dfs(n, s, t))
f++;
return f;
}

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

scanf("%d%d", &n, &m);
for (h = 0; h < m; h++) {
scanf("%d%d", &i, &j), i--, j--;
ij[h] = i ^ j;
cc[h * 4 + 0] = 1;
cc[h * 4 + 2] = 1;
link(i, h * 4 + 0);

link(j, h * 4 + 1);
link(j, h * 4 + 2);
link(i, h * 4 + 3);
}
printf("%d\n", dinic(n, 0, n - 1));
for (i = 0; i < n; i++)
if (dd[i] < n) {
struct L *l;

for (l = aa[i].next; l; l = l->next) {
h = l->h;
j = i ^ ij[h >> 2];
if (dd[j] == n && (h & 1) == 0)
printf("%d %d\n", i + 1, j + 1);
}
}
return 0;
}
```

Question 89

Question description

Byteland has n cities and m roads between them. The goal is to construct new roads so that there is a route between any two cities.

Your task is to find out the minimum number of roads required, and also determine which roads should be built.

Constraints

1≤n≤10^5
1≤m≤2·10^5
1≤a,b≤n

Input

The first input line has two integers n and m: the number of cities and roads. The cities are numbered 1,2,…,n.

After that, there are m lines describing the roads. Each line has two integers a and b: there is a road between those cities.

A road always connects two different cities, and there is at most one road between any two cities.

Output

First print an integer k: the number of required roads.

Then, print k lines that describe the new roads. You can print any valid solution.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define rep(i, a, b) for(int i = a; i < (b); ++i)
#define trav(a, x) for(auto& a : x)
#define all(x) begin(x), end(x)
#define sz(x) (int)(x).size()
typedef long long ll;
typedef pair<int, int> pii;
typedef vector<int> vi;
vi val, comp, z, cont;

int Time, ncomps;
template<class G, class F> int dfs(int j, G& g, F& f) {
int low = val[j] = ++Time, x; z.push_back(j);
trav(e,g[j]) if (comp[e] < 0)
low = min(low, val[e] ?: dfs(e,g,f));

if (low == val[j]) {
do {
x = z.back(); z.pop_back();
comp[x] = ncomps;
cont.push_back(x);
} while (x != j);
f(cont); cont.clear();
ncomps++;
}
return val[j] = low;
}
template<class G, class F> void scc(G& g, F f) {
int n = sz(g);
val.assign(n, 0); comp.assign(n, -1);
Time = ncomps = 0;
rep(i,0,n) if (comp[i] < 0) dfs(i, g, f);
}
int main() {
cin.sync_with_stdio(0); cin.tie(0);

cin.exceptions(cin.failbit);

int n, m;
cin >> n >> m;
vector<vi> g(n);
while(m--) {
int a, b;
cin >> a >> b;
a--, b--;
g[a].push_back(b);
g[b].push_back(a);
}
vi r;
scc(g, [&](vi &c) { r.push_back(c[0]); });
cout << sz(r)-1 << '\n';
rep(i, 1, sz(r))
cout << r[0]+1 << " " << r[i]+1 << '\n';
return 0;
}
```

Question 90

Question description

Consider a network consisting of n computers and m connections. Each connection specifies how fast a computer can send data to another computer.

Kotivalo wants to download some data from a server. What is the maximum speed he can do this, using the connections in the network?

Input

The first input line has two integers n and m: the number of computers and connections. The computers are numbered 1,2,…,n. Computer 1 is the server and computer n is Kotivalo's computer.

After this, there are m lines describing the connections. Each line has three integers a, b and c: computer a can send data to computer b at speed c.

Output

Print one integer: the maximum speed Kotivalo can download data.

Constraints

- 1≤n≤500
- 1≤m≤1000
- 1≤a,b≤n
- 1≤c≤109

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
using ll = long long;
#define F0R(i,a) for(int i=0; i<(a); i++)
#define FOR(i,a,b) for(int i=(a); i<=(b); i++)
int n, m;
ll adj[501][501], oadj[501][501];
ll flow[501];
bool V[501];
int pa[501];
void link(int i,int h){}
int bfs(int n,int s,int t){return 1;}
bool reachable() {
memset(V, false, sizeof(V));
queue<int> Q; Q.push(1); V[1]=1;
while(!Q.empty()) {
int i=Q.front(); Q.pop();

FOR(j,1,n) if (adj[i][j] && !V[j])
V[j]=1, pa[j]=i, Q.push(j);
}
return V[n];
}
int main() {
bfs(1,1,1);
link(1,1);
cin >> n >> m;
FOR(i,1,n) FOR(j,1,n) adj[i][j] = 0;
F0R(i,m) {
ll a,b,c; cin >> a >> b >> c;
adj[a][b] += c;
}
int v, u;
ll maxflow = 0;
while(reachable()) {
ll flow = 1e18;
for (v=n; v!=1; v=pa[v]) {
u = pa[v];
flow = min(flow, adj[u][v]);
}
maxflow += flow;
for (v=n; v!=1; v=pa[v]) {
u = pa[v];
adj[u][v] -= flow;
adj[v][u] += flow;
}
}
cout << maxflow << '\n';
}
```

Question 92

Question description

You are given an array of length N. You are required to count the number of (i,j) pairs where 1⩽i<j⩽N such that the difference of the array elements on that indices is equal to the sum of the square of their indices.

That is the count of the number of pairs of (i,j) such that it satisfies this equation (A[j]−A[i]=i2+j2).

Input format

- The first line contains the length of the array N. (1⩽N⩽105)
- The second line contains N integers representing array elements. (1⩽A[i]⩽1012)

Output format

Print the number of pairs that satisfy the provided condition.

Program Code:

```
#include<bits/stdc++.h>
#include<climits>
using namespace std;
void solve(){
cout<<"int cmfn1(const void *a,const void *b)";
}

int main() {
cin.tie(0);
long long int n;
cin >> n;
long long int a[n];
for (int i = 0; i < n; i++) {
cin >> a[i];
}
unordered_map<long long int, long long int> mp1;
for (int i = 0; i < n; i++) {
mp1[(a[i] + (long long)((long long int)(i + 1) * (i + 1)))]++;

}
unordered_map<long long int, long long int> mp2;
for (int i = 0; i < n; i++) {
mp2[(a[i] - (long long)((long long int)(i + 1) * (i + 1)))]++;
}
long long int cnt = 0;
for (auto it : mp1) {
cnt += (mp2[it.first]*it.second);
}
cout << cnt << endl;
}
```

Question 93

Problem Description

You are given with integers a,b,c,d,m. These represent the modular equation of a curve y2mod m=(ax3+bx2+cx+d) mod m

Also, you are provided with an array A of size N. Now, your task is to find the number of pairs in the array that satisfy the given modular equation.

If (Ai,Aj) is a pair then Aj2mod m=(aAi3+bAi2+cAi+d) mod m.

Since the answer could be very large output it modulo 109+7.

Note: A pair is counted different from some other pair if either Ai of the two pairs is different or Aj of the two pairs is different. Also for the convenience of calculations, we may count (Ai,Ai) as a valid pair if it satisfies given constraints.

Constraints
1≤T≤10 1≤N≤105 −2×109≤a,b,c,d,Ai≤2×109 1≤m≤2×109

Input Format
First line of the input contains number of test cases T.
First line for each test case consists of 5 space-separated integers a,b,c,d,m, corresponding to modular equation given.
Next line contains a single integer N.
Next line contains N space-separated integers corresponding to values of array A.

Output Format
For each test case, output a single line corresponding to number of valid pairs in the array mod 109+7.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
const int md = 1E9 + 7;
map<long long, int> mp;
int main() {
int t;
cin >> t;
while(t--) {
long long a, b, c, d, m;
cin >> a >> b >> c >> d >> m;
int n;
cin >> n;
int arr[n];
for(int i = 0; i < n; i ++) {
cin >> arr[i];
mp[(((arr[i] * arr[i]) % m) + m) % m] ++;
}
long long ans = 0;
for(int i = 0; i < n; i ++) {
long long x = (((((((a * arr[i]) % m) * ((arr[i] * arr[i]) % m)) % m) + (b * (
* arr[i]) % m) + d) % m) + m) % m;
if(mp.find(x) != mp.end())
ans += mp[x];
}
cout << (ans % md) << '\n';

mp.clear();
}
return 0;
}
```

Question 94

Problem Description

An integer array A and a number K have been provided to you.

Now you must determine whether any two items of the array A add up to the number K.

if two items are in different places in the array, they are regarded different.

Print "YES" (without quotations) if such a pair of integers exists; else, print "NO" (without quotes).

Constraints:

1≤N≤106

1≤K≤2∗106

1≤A[i]≤106

Input Format:

The first line consists of two integers *N*, denoting the size of array *A* and *K*. The next line consists of *N* space separated integers denoting the elements of the array *A*.

Output Format:

Print the required answer on a single line.

Program Code:

```
#include <iostream>
using namespace std;
#define f(i,a,n) for(int i=a;i<n;i++)
int main(){
int n,i;
cin>>n;
int a[n];

for(i=0;i<n;i++)
cin>>a[i];
int k;
cin>>k;
f(i,0,n){
f(j,0,n){
if(a[i]+a[j]==k)
{
cout<<"YES";
return 0;
}
}
}
cout<<"NO";
return 0;
cout<<"if(a[i]+a[j]>k)";
}
```

Question 96

Problem Description

Jenish and Neha are excellent friends. After performing several queries, Neha challenges Jenish to determine the highest possible Rating of the provided array A. According to Neha, the highest occurrence of an element in an array is its rating.

Jenish is given M and Q, the Magical Numbers. Jenish may perform Addition or Subtraction with M at most Q times for each element in the supplied Array A.

Because Jenish is stumped and unable to discover a solution, assist him in determining the highest possible ratings for the given array after applying queries to each element.

Constraints :

- 1≤N≤1000000
- 1≤M≤100
- 1≤Q≤10
- 1≤Ai≤1000000

Input :

- First line of Input contains integer *M*
- Second line contains integer *Q*
- Third line contains integer *N*
- Fourth line contains *N* elements representing elements of array *A*

Output :

- Output the highest possible rating of array *A* after applying Queries.

Explanation for test case 1

Jenish can add 1 in 1st element and subtract 1 from 3rd element to get highest frequency of 2 , i.e. 3

Program Code:

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

int main()
{
int M, Q, N,i;
scanf("%d %d %d", &M, &Q, &N);
int A[N];
for(int i=0 ; i<N ; i++)
scanf("%d", &A[i]);
int mx = A[0];
for(i=0;i<N;i++)
{
if(A[i]>mx)
mx = A[i];
}
// printf("%d\n", mx);
int size = mx + M*Q + 1;
// printf("%d\n", size);
int hash[size];
memset(hash, 0, sizeof(hash));
for(int i=0 ; i<N ; i++)
{
hash[A[i]]++;
for(int j=1 ; j<=Q ; j++)
{
int add = A[i] + (j*M);

int subtract = A[i] - (j*M);
if(add == subtract)
hash[add]++;
else
{
hash[add]++;
hash[subtract]++;
}
}
}
int ans = hash[0];
for(int i=0 ; i<size ; i++)
{
// printf("%d ", hash[i]);
if(hash[i]>ans)
ans = hash[i];
}
printf("%d\n", ans);
return 0;
}
```

Question 99

Problem Description

Given an array A of N integers. Now, you have to output the sum of unique values of the maximum subarray sum of all the possible subarrays of the given array A.
Note: Subarray means contiguous elements with at-least one element in it.

Constraints

1≤N≤2000
0≤|Ai|≤109

Input Format

The first line of the input contains a single integer N, the total number of elements in array A.
The next line of the input contains N space-separated integers representing the elements of the array.

Output Format

The only single line of the output should contain a single integral value representing the answer to the problem.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
void solve(){
cout<<"int NA[N];";
}
int main(){
int n;
cin>>n;
int a[n];
for(int i=0;i<n;i++){
cin>>a[i];
}
unordered_set<long long> s;
for(int i = 0 ; i< n; i++){
long long sum = 0 , max_sum=INT_MIN;
for(int j = i ; j<n ; j++){
sum += a[j];
max_sum = max(sum, max_sum);
if(sum<0){
sum = 0;
}
s.insert(max_sum);
}
}
long long ans = 0;
for(auto i:s){
ans+=i;

}
cout<<ans;
}
```

Question 100

Problem Description

Everyone knows that some Pikachus despise becoming Raichus. (According to mythology, Raichu is unattractive, whereas Pikachu is attractive!)

How do we track down these unique Pikachus who despise evolution? Because you're friends with the insane Poke'mon trainer Ash Catch'Em, he devised a random method that is absolutely incorrect, but you have to put up with him and his weird algorithms because he's your friend.

He thinks if you are given *N* Pikachus in an array, A_1,A_2 ... A_N, where each Pikachu is denoted by an integer. **The total number of unique pairs (A_i,A_j) where i < j is the number of Pikachus who hate evolution.**

Constraints:
$1 \leq N \leq 2 * 10^5$
$1 \leq A_i \leq 10^9$

Input format:
The first line will consist of a single integer *N*. The second line consists of *N* integers A_1,A_2 ... A_N.

Output format:
Output the total number of unique pairs (A_i,A_j) that can be formed, which will also be the number of special Pikachus.

Program Code:

```
#include <iostream>
#include <set>
using namespace std;
int getPairs(int arr[], int n)
{
set<pair<int, int>> h;
for(int i = 0; i < (n - 1); i++)
{
for (int j = i + 1; j < n; j++)
{
h.insert(make_pair(arr[i], arr[j]));

}
}
return h.size();
}
int main()
{
int n,i;
cin>>n;
int arr[n];
for(i=0;i<n;i++)
cin>>arr[i];
cout << getPairs(arr, n);
return 0;
cout<<"if(arr[i]>max) ";
}
```