DAA SEARCHING LVL-1

Vinoth's Model practical

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

int main(){
    int n,t,I,count=0;
    cin>>n>>t;
    int arr[n];
    for(i=0;i<n;i++){
        cin>>arr[i];
    }
    sort(arr,arr+n);
    for(i=0;i<n;i++){
        t-=arr[i];
        if(t<0){
            break;
        }
        count++;
    }
    cout<<count;
    return 0;
}
```

A double-square numberis an integer Y

```
#include <bits/stdc++.h>
using namespace std;
int sumSquare(int n)
{
    int res=0;
    for (long i = 0; i * i <= n; i++)
        for (long j = i; j * j <= n; j++)
            if ((i * i + j * j == n) ) {
                res++;
            }
    return res;
}
int main()
{
    int t;
    cin>>t;
    int i=1;
    while(t--){
        int n;
        cin>>n;
        cout<<"Line #"<<i<<": "<<sumSquare(n)<<endl;
        i++;
    }
    return 0;
```

```
    cout<<"for(i=0;i<=sqrt(y);i++) for(j=0;j<=i;j++)";
}
```

Trapped by a river and racing against time

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
   string str;
   int ramp1;
   double rate1,wr;
   getline(cin,str);
   cin>>ramp1>>rate1>>wr;
   double time1,speed1,dist1;
   time1=sqrt(2.0*ramp1/rate1);
   speed1=time1*rate1;
   dist1=speed1*speed1/9.805;
   cout<<str<<" will reach a speed of "<<std::fixed<<std::setprecision(2)<<speed1<<" m/s on a
"<<ramp1<<" ramp crossing "<<std::fixed<<std::setprecision(1)<<dist1<<" of
"<<std::fixed<<std::setprecision(1)<<wr<<" meters, ";
   if(dist1<(wr-5))
   cout<<"SPLASH!";
   else if(dist1>wr)
   cout<<"LIKE A BOSS!";
   else
   cout<<"BARELY MADE IT!";
   return 0;
}
```

Given 'm' postive integers

```
#include <bits/stdc++.h>
using namespace std;
#define f(n) for(i=0;i<n;i++)
#define g(n) for(i=1;i<n;i++)
```

```
#define k(n) for(i=n-2;i>=0;i--)
int maxWater(int arr[],int n)
{
    int left[n],i;
    int right[n];
    int water=0;
    left[0]=arr[0];
    g(n)
        left[i]=max(left[i-1],arr[i]);
    right[n-1]=arr[n-1];
    k(n)
        right[i]=max(right[i+1],arr[i]);
    for(i=1;i<n-1;i++)
    {
        int var=min(left[i-1],right[i+1]);
        if(var>arr[i])
        {
            water+=var-arr[i];
        }
    }
    return water;
}
int main()
    {
    int n,i;
    cin>>n;
    int arr[n];
    f(n){
        cin>>arr[i];
    }
    cout<<maxWater(arr,n);
    return 0;
```

```
}
```

The Allies are trying to get a message

```
#include <bits/stdc++.h>
using namespace std;
void solve(){ cout<<"break;";}
int main(){
string s1,s2,s3,s4;
double r;
double h;
cin>>s1>>r>>s2>>s3>>s4;
if(s2=="FEET")
r=r/3.28;
//cout<<r<<endl;
if(s2=="KILOMETERS") r=r*1000;
if(s2=="YARDS") r=r*0.9144;
if(s2=="INCHES") r=r*0.0254;
if(s2=="MILES") r=r*1609.34;
if(s4=="HOUR") r=r/3600;
if(s4=="MINUTE") r=r/60;
if(s2=="CENTIMETERS") r=r/100;
h=r*r/(2*9.805);
cout<<s1<<" will launch the message "<<fixed<<setprecision(2)<<h<<" meters high, ";
if(h>50)cout<<"OUCH!";
else if(h<25)cout<<"SPLAT!";
else cout<<"SUCCESS!";
return 0;}
```

Major Kathiravan

```
#include <bits/stdc++.h>
#define f(n) for(int i=0;i<n;i++)
using namespace std;
int main()
{
```

```
    int n;
    cin>>n;
    int arr[n];
    int res=10000;
    f(n){
        cin>>arr[i];
    }
    f(n){
    for(int j=i+1;j<n;j++){
     if(arr[i]>arr[j]){
        res=min(res,arr[i]-arr[j]);
    }
   }
}
cout<<res;
return 0;
cout<<"while";
}
```

Inspector Gadget

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    string F_str,K_str,X_str;
    getline(cin,F_str);
    getline(cin,K_str);
    getline(cin,X_str);
    string F = F_str.substr(2);
    string K = K_str.substr(2);
    string X = X_str.substr(2);
    if (X == "?"){
        float F_num = stof(F);
        float K_num = stof(K);
```

```
        float ans = F_num/(-K_num);
        cout << "X " << fixed << setprecision(2) << ans;
    }
    else if (F == "?"){
        float K_num = stof(K);
        float X_num = stof(X);
        float ans = -K_num * X_num;
        cout << "F " << fixed << setprecision(2) << ans;
    }
    else{
        float F_num = stof(F);
        float X_num = stof(X);
        float ans = -(F_num / X_num);
        cout << "K " << fixed << setprecision(2) << ans;
    }
    return 0;
}
```

Mr Somu

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int t;
    cin>>t;
    while(t--){
        int b,n,r;
        cin>>b>>n>>r;
    int z=1;
    for(int i=1;i<=n;i++){
        z=z*i;
    }
```

```
    int res;
    res=pow(b,z);
    cout<<res%r<<endl;
}
    return 0;
    cout<<"if(n%2==1)";
}
```

Given two integers 'b' and 'a'

```
#include <iostream>
using namespace std;
int main()
{
int t;
long long m;
long long n;
long long ans;
scanf("%d",&t);
for(int cs=1;cs<=t;cs++){
    scanf("%lld %lld",&n,&m);
    ans=(n*m)/2;
    printf("%lld\n",ans);
    }
}
```

In the following figure you can see a rectangular

```
#include <bits/stdc++.h>
using namespace std;
void solve(){
cout<<"return(l-2*x)*(b-2*x)*x;";
}
int main()
{
int tc;
```

```
double a, b, c, res, l, w, x;
scanf(" %d", &tc);
while(tc--) {
scanf(" %lf %lf", &l, &w);
a = 12.0;
b = -4.0 * (l+w);
c = l*w;
x = (-b - sqrt (b*b - 4.0*a*c)) / (2.0*a);
res = (l - 2*x) * (w - 2*x) * x;
printf ("%.9f\n", res);
}
return 0;
}
```

The Mask ate a block of dynamite to save

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
float a,c,d;
string b;
cin>>a>>b>>c;
float res;
int z=b.size();
if(z==1)
d=b[0]-48;
else
d=(float)(b[0]-48)/(b[2]-48);
res=a*d*0.45*7.5;
if(res>c){
cout<<res<<" the Mask should not eat it!";
}
else
```

```
cout<<fixed<<setprecision(2)<<res<<" the Mask can eat it!";
return 0;
cout<<"for";
}
```

There is a Gangaroo initially placed at the

```
#include <stdio.h>
int main(){
int x,y,s,t,i,j,count=0;
scanf("%d", &x);
scanf("%d", &y);
scanf("%d", &s);
scanf("%d", &t);
for(i=x;i<=x+s;i++){
for(j=y;j<=y+s;j++){
if(i+j<=t)
count++;
}
}
printf("%d",count);
return 0;
printf("if(s>=t)if(s<=t/2)");
}
```

DAA Sorting

Scrooge McDuck

```
#include <iostream>
using namespace std;
int main()
{
int p,q,r,i;
int c;
cin>>c;
for(i=0;i<c;i++){
cin>>p>>q>>r;
q=q+(r-1)/5;
r=(r-1)%5+1;
p=p+(q-1)/10;
q=(q-1)%10+1;
cout<<p<<" ";
cout<<q<<" ";
cout<<r<<endl;
}
return 0;
}
```

Tina owns a match making company

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
int t,n;
cin>>t;
while(t--){
cin>>n;
int a[n],b[n],sum=0;
for(int i = 0;i<n;i++)
```

```
cin>>a[i];
for(int i=0;i<n;i++)
cin>>b[i];
sort(a,a+n);
sort(b,b+n);
for(int 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;
}
```

Shankar is a volleyball trainer

```
#include <bits/stdc++.h>
using namespace std;
typedef long long LL;
const int N=(int)1e6+6,mod=(int)0;
int a[N];
long long sum[N];
int main(){
int tc;
cin>>tc;
for(int tt=1;tt<=tc;++tt){
int n,p;
cin>>n>>p;
for(int j=0;j<n;++j)
cin>>a[j];
sort(a,a+n);
int i;
for(i=0;i<n;i++)
sum[i+1]=sum[i]+a[i];
```

```
long long res=1e18;
for(int j=p-1;j<n;++j){
long long s=sum[j+1]-sum[j-(p-1)];
long long cost=(LL)a[j]*p-s;
res=min(res,cost);
}
cout<<res<<'\n';
}
}
```

ROYGBIV

```
#include <bits/stdc++.h>
using namespace std;
typedef long long LL;
const int N=(int)1e6+6,mod=(int)0;
int a[N];
long long sum[N];
int main(){
int tc;
cin>>tc;
for(int tt=1;tt<=tc;++tt){
int n,p;
cin>>n>>p;
for(int j=0;j<n;++j)
cin>>a[j];
sort(a,a+n);
int i;
for(i=0;i<n;i++)
sum[i+1]=sum[i]+a[i];
long long res=1e18;
for(int j=p-1;j<n;++j){
long long s=sum[j+1]-sum[j-(p-1)];
long long cost=(LL)a[j]*p-s;
```

```
res=min(res,cost);
}
cout<<res<<'\n';
}
}
```

Ace Ventura, Pet Detective

```
#include <bits/stdc++.h>
using namespace std;
#define p1 cout<<"Ace, move fast, pigeon is at ("<<i<<",0)";
#define p2 cout<<"Ace, move fast, pigeon is at ("<<(i-i/z)%z<<","<<i/z<<")";
#define p3 cout<<"No pigeon, try another map, Ace";
#define a continue;
#define f(n)for(int i=0;i<z;i++)
#define while1 while((scanf("%c",&s[i]))!=EOF)
int main(){
string s1;cin>>s1;
int z=s1.size();
f(n){
if(s1[i]=='P'){p1
return 0;}
}
//cout<<z<<endl;
int i=0,cnt=0;
char s[10000];
while1{
if(s[i]=='P'){
cnt=1;
break;
}
i++;
}
if(cnt==1)p2
```

```
   else p3}
```

Ganesan has a string

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
string s,s2;
cin>>s>>s2;
int z=s.length();
int i;
int a[z];
for(i=0;i<(int)s.length();i++){
a[i]=s[i+1]-s[i];
}
for(int i=0;i<z-2;i++){
if(a[i]!=a[i+1]){
cout<<"No";
return 0;}
}
cout<<"Yes";
return 0;
}
```

Great news! You get to go to Japan

```
#include<iostream>
using namespace std;
int main()
{
int items;
int a,j,cnt=0;
cin>>a>>items;
int c[items];
string s[items];
```

```
for(j=0;j<items;j++){
cin>>s[j]>>c[j];
if(c[j]<a){
cout<<"I can afford "<<s[j]<<endl;
a=a-c[j];
}
else{
cnt++;
cout<<"I can't afford "<<s[j]<<endl;
}
//cout<<cnt;
}
if(cnt==items)
cout<<"I need more Yen!";
else
cout<<a;
return 0;
cout<<"for(i=1;i<=yen;i++) int i,j;";
}
```

Tina owns a match making company, which even to her

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
int t,n;
cin>>t;
while(t--){
cin>>n;
int a[n],b[n],sum=0;
for(int i = 0;i<n;i++)
cin>>a[i];
for(int i=0;i<n;i++)
```

```
cin>>b[i];
sort(a,a+n);
sort(b,b+n);
for(int 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;
}
```

The sapphire consulting and marketing company is adding

```
#include <stdio.h>
#include <stdlib.h>
int isqrt(n) int n; {
int i;
for(i=0;i*i<n;i++);
return i;
}
int main() {
int c;
int t,h,s,i,j;
int d;
scanf("%d",&c);
for(i=0;i<c;i++) {
s=0;
scanf("%d %d",&t,&h);
d=isqrt(t);
s+=t+(d*4);
for(j=1;j<h;j++) {
s+=3;
s+=(d+j)*4;
```

```
if((d+j)>2)
s+= (d+j-2)*2;
}
printf("%d liters\n",s);
}
return 0;
}
```

Ganesan has a string S consisting of lowercase

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
string s,s2;
cin>>s>>s2;
int z = s.length();
int i;
int a[z];
for(i=0;i<(int)s.length();i++){
a[i]=s[i+1]-s[i];
}
for(int i=0;i<z-2;i++){
if(a[i]!=a[i+1]){
cout<<"No";
return 0;}
}
cout<<"Yes";
return 0;
}
```

there are n integers

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

```
#include <stdlib.h>
#include <assert.h>
#define if


int lonelyinteger(int a_size, int* a) {
    int i=0;
    int num=0;
    for(i=0;i<a_size;i++){
        num=num^a[i];
    }

    return num;

}
int main() {
    int res;

    int _a_size, _a_i;
    scanf("%d", &_a_size);
    int _a[_a_size];
    for(_a_i = 0; _a_i < _a_size; _a_i++) {
        int _a_item;
        scanf("%d", &_a_item);

        _a[_a_i] = _a_item;
    }
  res = lonelyinteger(_a_size, _a);
    printf("%d", res);

    return 0;
}
```

```
void y(){
   printf("break;");
}
```

Siva has several containers, each with a number

```
#include <stdio.h>
#include <stdlib.h>
void insertionSort(long int *p,long int n);
void asd();
int main(){
    asd();
 return 0;
}
void asd()

{
 int q;
 scanf("%d",&q);
 while(q--){
 int n,i,j;
 scanf("%d",&n);
 int M[n][n];
 long int *r,*c,*arr;
 arr=(long int *)malloc(n*n*sizeof(long int));
 *arr=n;
 r=(long int *)malloc(n*sizeof(long int)); c=(long int *)malloc(n*sizeof(long int));
 for(i=0;i<n;i++){
 for(j=0;j<n;j++){
 scanf("%d",&M[i][j]);
 r[i]+=M[i][j];
 c[j]+=M[i][j];
 }
 }
```

```
int count=0;
for(i=0;i<n;i++){
for(j=0;j<n;j++){
if(r[i]==c[j])
{
count++;
break;
}
}
}
if(count==n)
printf("Possible\n");
else
printf("Impossible\n");
}
}
```

Avul pakir

```
#include <iostream>
#include<string.h>
using namespace std;
int main(){
    int n,i,j,t;
    cin>>n;
    for(i=0;i<n;i++){
      cout<<"Line "<<i+1<<":"<<endl;
      cin>>t;
      int sum=0;
      for(j=0;j<t;j++){

          char name[100];
          cin>>name;
          if(strcmp(name,"donate")==0){
```

```
            int d;cin>>d;
            sum+=d;
          }
          else{cout<<sum<<endl;}

      }
    }
    return 0;
}
```

THAI PONGAL

```
#include <iostream>
using namespace std;
int factors(int num,int l) {
  int i,c1=0;
  for(i=1; i <= num; i++) {
     if (num % i == 0 && i>l)        c1++;}   return c1;   cout<<"continue;";}
int main()
{
    int t,j;
    cin>>t;
    for(j=1;j<=t;j++)
    {
        int p,l,q,i,c=0,sp;
        cin>>p>>l;
        q=p-l;

        printf("Line %d: ",j);
        sp=factors(q,l);
        for(i=1;i<=q;i++)
        {
           if(q%i==0 && i>l)
            {
```

```
            printf("%d",i);
            if(c<sp-1)printf(" ");
            c++;
          }
      }
      if(c==0) printf("impossible");
      printf("\n");
   }
        return 0;
}
```

DIVIDE AND CONQUER

Programmer Sandhosh and you have a New Year Tree (not the traditional fur tree, though)

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e6+10;

int m,cnt=4,La=2,Lb=3,len=2;
int f[N][21],dep[N];

int lca(int x,int y) {
   if(dep[x]<dep[y]) swap(x,y);
   for(int i=20;i>=0;i--) if(dep[f[x][i]]>=dep[y]) x=f[x][i];
   if(x==y) return x;
   for(int i=20;i>=0;i--) if(f[x][i]!=f[y][i]) x=f[x][i],y=f[y][i];
   return f[x][0];
}

int dis(int x,int y){
   return dep[x]+dep[y]-dep[lca(x,y)]*2;
}

int main() {
   scanf("%d",&m);
   dep[1]=1;
   dep[2]=dep[3]=dep[4]=2;
   f[2][0]=f[3][0]=f[4][0]=1;
   int u;
   while(m--) {
   cin>>u;
      int x=cnt+1,y=cnt+2;
      cnt+=2;
      f[x][0]=f[y][0]=u;
```

```
        for(int i=1; i<=20; i++) f[x][i]=f[y][i]=f[f[x][i-1]][i-1];
        dep[x]=dep[y]=dep[u]+1;
        int d1=dis(La,x);
        int d2=dis(Lb,x);
        if(len<d1) len=d1,Lb=x;
        if(len<d2) len=d2,La=x;
        printf("%d\n",len);
    }
    return 0;
}
```

Leopard is in the Amusement Park. And now she is in a queue in front of the Ferris wheel

```
#include<cstdio>
#include<iostream>
using namespace std;
inline int getint(){
char c;
while((c=getchar())<'0'||c>'9');return c-'0';
}
const int N=4005,inf=.5e9;
int n,k,sum[N][N],f[N],g[N];
int main(){
cin>>n>>k;
for(int i=1;i<=n;i++)
for(int j=1;j<=n;j++)
sum[i][j]=sum[i-1][j]+sum[i][j-1]-sum[i-1][j-1]+getint();
g[n+1]=n;
for(int kk=2;kk<=k;kk++)
for(int i=n;i;i--){
f[i]=-inf;
for(int j=g[i];j<=g[i+1]&&j<i;j++){
int now=f[j]-sum[j][j]+sum[j][i];
if(now>f[i]){
```

```
f[i]=now;
g[i]=j;
}
}
}
printf("%d\n",sum[n][n]/2-f[n]);
}
```

Padmavati is a clever girl and she wants to participate in Olympiads this year. Of course she wants her partner to be clever too (although he's not)! Padmavati has prepared the following test problem for Sativathi

```
#include <iostream>
#include <map>
using namespace std;
const int N=1<<20;
int n,a[N],c[N],w;
void upd(int i,int c){

}
int main(){
  cin>>n;
  for(int i=0;i<n;++i)cin>>a[i];
  map<int,int>u,v;
  for(int i=n;i-->0;){
    int x=++u[a[i]];
    while(x<N)++c[x],x+=x&-x;
  }

  for(int i=0;i<n;++i){
    int x=u[a[i]]--,y=v[a[i]]++;
    while(x<N)--c[x],x+=x&-x;
    while(y>0)w+=c[y],y-=y&-y;
  }
  cout<<w<<endl;
```

```
}
```

Maakesh caught the trail of the ancient Book of Evil in a swampy area

```
#include <bits/stdc++.h>
using namespace std;
const int N = 100005;
int R,D,n,m,d,h[N];
vector<int> adj[N];
bool prob[N],is[N];
void evil(int u,int p=0){
    h[u]= h[p]+1;
    prob[u] &= (h[u]<=d);
    if(is[u]&&h[u]>D)
        D=h[u],R=u;
    for(unsigned int i=0;i<adj[u].size();++i){
        int v= adj[u][i];
        if(v!=p)
            evil(v,u);
    }
}
int main(){
    cin>>n>>m>>d;memset(prob,true,sizeof(prob));
    h[0]=-1;int a,b,i;D=0;
    for(i=0;i<m;++i)
        cin>>R,is[R]=true;
    for(i=0;i<n-1;++i)
        scanf("%d%d",&a,&b),adj[a].push_back(b),adj[b].push_back(a);
    evil(R);evil(R);evil(R);
    int ret=0;
    for(i=1;i<=n;++i)
        if(prob[i])++ret;
    cout<<ret<<endl;
}
```

Lakshman and Sukran are the best competitive programmers in their town. However, they can't both qualify to an important contest. The selection will be made with the help of a single problem. Bhoominath, a friend of Lakshman, managed to get hold of the problem before the contest. Because he wants to make sure Lakshman will be the one qualified, he tells Lakshman the following task

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

using namespace std;

long long n, i = 1, j, k = 9e9, x, s[100001], d;


int main() {
    cin>>n;
    for (; i <= n; i++){ cin>>x;s[i] = s[i - 1] + x;}
    for (i = 1; i <= n; i++)
        for (j = max(1ll, i - 20000); j <= i; j++)
            if (i != j) k = min(k, (i - j) * (i - j) + (s[i] - s[j]) * (s[i] - s[j]));
    cout << k;
}
```

Recently Aarush has become keen on physics. Anna V., his teacher noticed Aarush's interest and gave him a fascinating physical puzzle a half-decay tree

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

using namespace std;

int h,q,v,e;string str;map<int,int> f;

double puzzle(int u,int mx) {return (f[u]<=mx)?mx:(0.5*(puzzle(u<<1,max(mx,f[u]-f[u<<1]))+puzzle(u<<1|1,max(mx,f[u]-f[u<<1|1]))));}

int main(){

cin>>h>>q;
    while (q--){
        cin>>str;
        if (str[0]=='a'){
            scanf("%d %d",&v,&e);
            while (v) f[v]+=e,v>>=1;
        }
        else printf("%.2lf\n",puzzle(1,0));
    }
```

```
    return 0;
}
```

Prithvi are in the world of mathematics to solve the great "Monkey and the carrot"

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

int main()
{

    int numberOfColumns;

        cin>>numberOfColumns;
        int bananaMatrix[2 * numberOfColumns - 1][numberOfColumns]; //Input matrix
        int maxBanana[2 * numberOfColumns - 1][numberOfColumns];    //Memoized matrix

        memset(maxBanana, 0, sizeof(maxBanana));       //Setting 0 to all cell, will update for maximum
        memset(bananaMatrix, 0, sizeof(bananaMatrix)); //Setting 0 to all cell, will update for inputs

        //Input for upper triangle
        for (int row = 0; row < numberOfColumns; row++)
            for (int column = 0; column <= row; column++)
                cin >> bananaMatrix[row][column];

        //Input for lower triangle
        int shiftedPosition = 1;
        for (int row = numberOfColumns; row < (numberOfColumns * 2) - 1; row++)
        {
            for (int column = shiftedPosition; column < numberOfColumns; column++)
                cin >> bananaMatrix[row][column];
            shiftedPosition++;
        }
```

```
        maxBanana[0][0] = bananaMatrix[0][0];
        for (int row = 1; row < numberOfColumns; row++)
        {
            for (int column = 0; column <= row; column++)
                if (column == 0)//Caution for negative indexes.
                    maxBanana[row][column] = maxBanana[row - 1][column] +
bananaMatrix[row][column];
                else
                    maxBanana[row][column] = max(maxBanana[row - 1][column], maxBanana[row
- 1][column - 1]) + bananaMatrix[row][column];
        }

        //Memoizing the lower triangle to store the max value
        shiftedPosition = 1;
        for (int row = numberOfColumns; row < (numberOfColumns * 2) - 1; row++)
        {
            for (int column = shiftedPosition; column < numberOfColumns; column++)
                maxBanana[row][column] = max(maxBanana[row - 1][column], maxBanana[row -
1][column - 1]) + bananaMatrix[row][column];
            shiftedPosition++;
        }
        cout <<maxBanana[2 * numberOfColumns - 2][numberOfColumns - 1];



    return 0;
    cout<<"cin>>carrotMatrix[row][column];";
}
```

In this problem you will meet the simplified model of game Pudding Monsters

```
#include <bits/stdc++.h>
#define fi first
#define se second
#define lo long long
```

```
#define inf 1000000009
#define md 1000000007
#define li 300005
#define mp make_pair
#define pb push_back
using namespace std;
int n,x,y,v[li],a[li],b[li],mn[li],mx[li],g[li];
lo int ans;
void work(int a,intb)
{
    int n=a[0],m=b[0];
    mn[m+1]=0;
    for(int i=1;i<=m;i++){
        mn[i]=min(mn[i-1],b[i]);
        mx[i]=max(mx[i-1],b[i]);
    }
    int mna=inf,mxa=0;
    int l=1,r=1;
    for(int i=1;i<=n;i++){
        mna=min(mna,a[i]);
        mxa=max(mxa,a[i]);
        int d=mxa-mna+1-i;
        if(d>0 && d<=m && mn[d]>mna && mx[d]<mxa) ans++;
        for( ;mn[r]>mna;r++) g[mx[r]-r]++;
        for( ;l<r&&mx[l]<mxa;l++) g[mx[l]-l]--;
        ans+=g[mna+i-1];
    }
    for(int i=l;i<r;i++) g[mx[i]-i]=0;
}
void solve(int l,int r){
    if(l==r) return ;
    int mid=(l+r)/2;
```

```
    a[0]=mid-l+1;b[0]=r-mid;
    for(int i=l;i<=mid;i++) a[mid+1-i]=v[i];
    for(int i=mid+1;i<=r;i++) b[i-mid]=v[i];
    work(a,b),work(b,a);
    solve(l,mid);solve(mid+1,r);
}
int main(){
    cin>>n;
    for(int i=1;i<=n;i++){
        cin>>x>>y;
        v[x]=y;
    }
    mn[0]=inf;
    solve(1,n);
    printf("%lld\n",ans+n);
    return 0;

}
```

Fazil is an unemployed computer scientist who spends his days working at odd-jobs.

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

using namespace std;

string word;
long long dp[100][100];



long long calculate(int s, int e){

    if(s > e)
        return 0;
```

```
    if(s == e )
        return 1;

    if(dp[s][e] != -1)
        return dp[s][e];

    if(word[s] == word[e])
        return dp[s][e] = 1 +  calculate(s+1, e) + calculate(s, e-1);
    else
        return dp[s][e] = calculate(s+1, e) + calculate(s, e-1) - calculate(s+1, e-1);

}



int main(){

        cin>>word;

        memset(dp, -1, sizeof dp);

        cout<<calculate(0,word.size()-1)<<endl;
    return 0;
    printf("long long calculate(int s,int e)");
}
```

A set of points on a plane is called fair, if for any two points at least one of the three conditions is true

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

pair<int,int>p[10010];
set<pair<int,int> >s;

void dfs(int l,int r)
```

```
{
    if(l==r)
    {
        s.insert(p[l]);
        return;
    }
    int i,mid=(l+r)/2;
    dfs(l,mid);
    dfs(mid+1,r);
    for(i=l;i<=r;i++) s.emplace(p[mid].first,p[i].second);
}

int main()
{
    int n,i;
    scanf("%d",&n);
    for(i=1;i<=n;i++) scanf("%d%d",&p[i].first,&p[i].second);
    sort(p+1,p+n+1);
    dfs(1,n);
    printf("%ld\n",s.size());
    for(auto it:s) printf("%d %d\n",it.first,it.second);
    return 0;
    printf("void fiv(int l,int r),cin>>n;cin>>a[i].first>>a[i].second;");
}
```

Prof.Dr. Ramalingam need representing positive integer N as a sum of addends, where each addends is an integer number containing only 1s

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

long long n,a[17];

int dfs(long long n,int x)
{
```

```
        int num=n/a[x];n%=a[x];
        if (!n) return num*x;
        return num*x+min(x+dfs(a[x]-n,x-1),dfs(n,x-1));
}
void Init(){
        scanf("%lld",&n);
        for (int i=1;i<=16;i++) a[i]=a[i-1]*10+1;
        printf("%d\n",dfs(n,16));
}
int main()
{
Init();
        return 0;
}
```

Now Sabanayagam becomes a commander of Ladakh. Ladakh, like its name said,

```
#include<bits/stdc++.h>
using namespace std;
#define N 100005

int cnt[26][100005];
char ans[N];
vector<int> g[N];
void man(){
   cout<<"void dfs(int u,int par) cin>>n; cin>>u>>v;";
}
void dfs(int s,int f){
   for(auto x:g[s])if(x!=f)dfs(x,s);
   int p;
   for(int i=0;i<26&&cnt[i][s]<2;i++)
      if(!cnt[i][s])p=i;
   cnt[p][s]++;
   ans[s]='A'+p;
```

```
    for(int i=0;i<=p;i++)cnt[i][f]+=cnt[i][s];
    return ;
}

int main(){
    int n,i,a,b;
    scanf("%d",&n);
    for(i=1;i<n;i++){
        scanf("%d %d",&a,&b);
        g[a].push_back(b);
        g[b].push_back(a);
    }
    dfs(1,0);
    for(i=1;i<=n;i++)printf("%c ",ans[i]);
    return 0;
}
```

Kishan are developing a 'Love calculator' software. You are planning to write the software in a complex way such that nobody would be able to crack the exact behavior of the software

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

using namespace std;

int main()
{



    string name1, name2;

    int shortestString[31][31];
    long uniqueString[31][31];



        cin >> name1 >> name2;
```

```
        //Shift the characters of the name to right for ease of memoizing
        name1.insert(0, "0");
        name2.insert(0, "1");

        //Prepare the matrices for memoization
        for (int i = 0; i < 31; i++)
            shortestString[0][i] = shortestString[i][0] = i, uniqueString[i][0] = uniqueString[0][i] = 1;

        for (int i = 1; name1[i]; i++)
        {
            for (int j = 1; name2[j]; j++)
            {
                //Checking if we need to take the cumulative sum from upper-left block
                if (name1[i] == name2[j])
                {
                    //Adding 1 to cumulative sum from upper-left block
                    shortestString[i][j] = 1 + shortestString[i - 1][j - 1];

                    //No need to add a new branch of unique strings so taking cumulative sum from
upper-left block
                    uniqueString[i][j] = uniqueString[i - 1][j - 1];

                }
                else
                {
                    //Finding the minimum from left and upper block and adding 1 to the value of
current block
                    shortestString[i][j] = 1 + min(shortestString[i][j - 1], shortestString[i - 1][j]);

                    //Checking if there are two unique strings of the same length
                    if (shortestString[i][j - 1] == shortestString[i - 1][j])
```

```
                uniqueString[i][j] = uniqueString[i][j - 1] + uniqueString[i - 1][j];


                //Checking if left block has the minimum value in shortestString matrix
            else if (shortestString[i][j - 1] < shortestString[i - 1][j])
                uniqueString[i][j] = uniqueString[i][j - 1];
            else
                uniqueString[i][j] = uniqueString[i - 1][j];
          }
        }
      }
      cout <<shortestString[name1.length() - 1][name2.length() - 1] << " " <<
uniqueString[name1.length() - 1][name2.length() - 1];


   return 0;
   cout<<"cin>>name1>>name2;";
}
```

After the long contest, Sameer returned home and got angry after seeing his room dusty

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



using namespace std;



int partition(int array[],int leftIndex,int rightIndex){
   int pivotValue = array[rightIndex];
   int toBePivotIndex = (leftIndex - 1);
   for(int comparisonIndex = leftIndex; comparisonIndex <= rightIndex - 1;
comparisonIndex++){
      if (


         array[comparisonIndex] < pivotValue
      ) {
```

```
            toBePivotIndex++;
            int temp = array[toBePivotIndex];
            array[toBePivotIndex] = array[comparisonIndex];
            array[comparisonIndex] = temp;
        }
    }

    int temp = array[toBePivotIndex+1];
    array[toBePivotIndex+1] = array[rightIndex];
    array[rightIndex] = temp;

    return (toBePivotIndex + 1); // new pivot point
}


void quickSort(int array[],int leftIndex,int rightIndex){

    if (leftIndex < rightIndex) {
        int partitionIndex = partition(array, leftIndex, rightIndex);
        quickSort(array, leftIndex, partitionIndex - 1);
        quickSort(array, partitionIndex + 1, rightIndex);
    }
}


int main(){

    int numberOfDustPoints,widthOfBrush,xCoordinate,yCoordinate;

        int numberOfMoves = 0;
        cin>>numberOfDustPoints>>widthOfBrush;
        int dustPointsYCoordinates[numberOfDustPoints];
```

```
        for(int i = 0; i < numberOfDustPoints; i++){
            cin >> xCoordinate >> yCoordinate;
            dustPointsYCoordinates[i] = yCoordinate;
        }

        quickSort(dustPointsYCoordinates,0, numberOfDustPoints-1);


        int currentBrushYCoordinate = dustPointsYCoordinates[0];
        numberOfMoves++;

        for (int i = 0; i < numberOfDustPoints; i++) {
        if(currentBrushYCoordinate + widthOfBrush < dustPointsYCoordinates[i]) {
        currentBrushYCoordinate = dustPointsYCoordinates[i];
        numberOfMoves++;
        }

    }
        cout <<numberOfMoves;


    return 0;
}
```

Makesh

```
#include <bits/stdc++.h>
using namespace std;
const int N = 100005;
int R,D,n,m,d,h[N];
vector<int> adj[N];
bool prob[N],is[N];
void evil(int u,int p=0){
```

```
    h[u]= h[p]+1;
    prob[u] &= (h[u]<=d);
    if(is[u]&&h[u]>D)
        D=h[u],R=u;
    for(unsigned int i=0;i<adj[u].size();++i){
        int v= adj[u][i];
        if(v!=p)
            evil(v,u);
    }
}
int main(){
    cin>>n>>m>>d;memset(prob,true,sizeof(prob));
    h[0]=-1;int a,b,i;D=0;
    for(i=0;i<m;++i)
        cin>>R,is[R]=true;
    for(i=0;i<n-1;++i)
        scanf("%d%d",&a,&b),adj[a].push_back(b),adj[b].push_back(a);
    evil(R);evil(R);evil(R);
    int ret=0;
    for(i=1;i<=n;++i)
        if(prob[i])++ret;
    cout<<ret<<endl;
}
```

DAA GREEDY ALGORITHM

Vaanavan thinks that lucky tickets are the tickets whose numbers are divisible by 3.

```
#include<bits/stdc++.h>
using namespace std;
int a[3];
int main()
{
    int n,x,i;
    cin>>n;
    for(i=1;i<=n;i++)
    {
        cin>>x;
        a[x%3]++;
    }
    cout<<a[0]/2+min(a[1],a[2])<<endl;
    return 0;
}
```

```
A sportsman starts from point xstart = 0
```

```
#include <bits/stdc++.h>
using namespace std;
void xyz(){}
typedef long long ll;
typedef pair<int, int> pii;
const int mod = 1000000007;

int main() {
    ios::sync_with_stdio(false);
    int n, m, s, d, a[200005] = {}, c = 0, t = 0;
    vector<int> z;
    cin >> n >> m >> s >> d;
    for (int i = 0; i < n; i++) cin >> a[i];
```

```
    sort(a, a + n);
    if (a[0] <= s) {cout << "IMPOSSIBLE"; return 0;}
    c = a[0] - 1;
    z.push_back(a[0] - 1);
    a[n] = mod + mod;
    while (t < n) {
        while (t < n && a[t + 1] - a[t] <= s + 1) t++;
        if (a[t] + 1 - c > d) {cout << "IMPOSSIBLE"; return 0;}
        z.push_back(a[t] + 1 - c);
        c = a[t] + 1;
        t++;
        if (t == n) z.push_back(m - c), c = m;
        else z.push_back(a[t] - c - 1), c = a[t] - 1;
    }
    if (!z.back()) z.pop_back();
    bool b = 0;
    for (int i : z) {
        if (b) cout << "JUMP ";
        else cout << "RUN ";
        b = !b;
        cout << i << '\n';
    }if(1>2)cout<<"cin>>n>>m>>s>>d; \n cin>>a[i];";
}
```

A hamburger stand received n orders for rental

```
#include<bits/stdc++.h>
using namespace std;
int n,l,z;
pair<int,int> a[500020];
int main(){
    cin>>n;
    for(int i=0;i<n;i++){
        cin>>a[i].second>>a[i].first;
```

```
    }
    sort(a,a+n);
    for(int i=0;i<n;i++){
        if(l<a[i].second){
            z++;
            l=a[i].first;
        }
    }cout<<z;
    return 0;
}
```

It's a very unfortunate day for lavanya today

```
#include <bits/stdc++.h>
using namespace std;
#define res cin>>a[i],num+=a[i];
#define f1     for(int i=1;i<=n;i++)
double n,v,a[25],b[25],sum,mx=1e9;
int main(){
    cin>>n>>v;
    f1{
        cin>>a[i];
        sum+=a[i];
    }
    for(int i=1;i<=n;i++)
        cin>>b[i];
    for(int i=1;i<=n;i++)
        mx=min(mx,b[i]/a[i]);
    cout << fixed<<setprecision(1)<<min(mx*sum,v);
    return 0;
}
```

shiv has given a rebus of form

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

```
int p = 1, n, j, a[105];
char c;
int main()
{
    a[j++] = 1;
    while (cin>>c && c != '=')
    {
        if (c == '-') p--, a[j++] = -1;
        if (c == '+') p++, a[j++] = 1;
    }
    cin>>n;
    for(int i=0;i<j;i++)
    {
        if(a[i]>0)while (p<n && a[i]<n) a[i]++, p++;
        else while (p>n&&a[i]<0 && a[i]>-n) a[i]--, p--;
    }
    if (p != n) { cout << "Impossible\n"; return 0; }
    cout << "Possible\n";
    for(int i=0;i<j;i++)
    cout << (i ? (a[i]<0 ? "- " : "+ ") : "") << abs(a[i]) << " ";
    cout << "= " << n;

}
a long time ago
#include<bits/stdc++.h>

int a,i;
int main()
{std::string s,t;
std::cin>>s>>t;
for(i=s.find(t);i+1;++a,i=s.find(t,i+t.size()));std::cout<<a;}
```

A Steeling

```
#include<bits/stdc++.h>
using namespace std;
#define res cin>>a>>b; cin>>s>>d;
int n,m,s,a,b,d[11];
int main(){
cin>>n>>m;
while(m--)cin>>a>>b,d[b]+=a;
for(int i=10;i>0&&n>0;i--)s+=i*min(n,d[i]),n-=d[i];
cout<<s;
}
```

There are banks in the city where Vishnu lives

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
ll n,maxs=0;
map<ll,ll> mp;
int main(){
    cin>>n;
    for(ll i=0,x=0,y;i<n;++i)
        cin>>y,maxs=max(maxs,++mp[x+=y]);
    cout<<n-maxs;
}
```

A group of tourists is going on to rameshwaram and dhanushkodi tour

```
#include<bits/stdc++.h>
using namespace std;
int c[2],i,x,t,n,p,j;
pair<int,int> a[2][1<<17];
#define F(i,n) for(i=0;i<n;++i)
void aasd(){
    cout<<"cin>>n>>v;cin>>t>>v;";
}
```

```
int main(){
    scanf("%d%d",&n,&p);
    F(i,n){
        scanf("%d%d",&t,&j);
        a[t&1][++c[t&1]]=make_pair(-j,i+1);
    }
    F(i,2)sort(a[i]+1,a[i]+c[i]+1);
    F(i,2)F(j,c[i])a[i][j+1].first+=a[i][j].first;
    n=min(p,c[1]);
    for(i=0;~n;--n)
        if((t=a[1][n].first+a[0][min(*c,(p-n)/2)].first)<x)i=n,x=t;
    printf("%d\n",-x);
    F(t,i)printf("%d ",a[1][t+1].second);
    t=min(*c,(p-i)/2);
    F(i,t)printf("%d ",a[0][i+1].second);
    return 0;
}
```

Nadanan's company employed n people. Now Nadanan needs to build a tree hierarchy

```
#include<bits/stdc++.h>
using namespace std;
int a[10001],n,m,x,y;
int main(){
    cin>>n;
    for(int i=0;i<=n;i++)
        cin>>m,a[i]=1e9;
    for(int i=1;i<=m;i++){
        cin>>x>>x>>y;
        a[x]=min(a[x],y);
    }
    x=y=0;
    for(int i=1;i<=n;i++)
        if(a[i]!=1e9){
```

```
            x++;
            y+=a[i];
        }
    if(n<x+2) cout<<y;
    else cout<<-1;
    return 0;
    cout<<"cin>>ans[0]; cin>>a>>b>>c;";
}
A remote island chain contains islands,
#include<iostream>
using namespace std;
int N;
int a[200010], b[200010];
int main()
{
    int i, j;
    cin>>N;
    for(i=0;i<N-1;i++)
    {
        cin>>a[i];
        if(a[i]==0) i--;
    }

    for(i=0;i<N-1;i++)
    {
        scanf("%d",&b[i]);
        if(b[i]==0) i--;
        if(b[i]==a[0]) j=i;
    }
    for(i=0;i<N-1;i++,j++)
    {
        if(a[i]!=b[j%(N-1)])
```

```
        {
            printf("NO\n");
            return 0;}
    }
    printf("YES\n");
    return 0;
    cout<<"cin>>n;cin>>b[i];";
}
```

Students in a class are making towers of blocks.

```
#include<iostream>
using namespace std;
int main(){
    int n,m,i=0;
    cin>>n>>m;
    for(i=0;i/2<n||i/3<m||i/2+i/3-i/6<n+m;i++);
    cout<<i;
    return 0;
}
```

Samantha has given an array of N elements, you must make it a co-prime array

```
#include<bits/stdc++.h>
using namespace std;
int n,x,p=1;
int main(){
vector<int>X;
for(cin>>n;cin>>x;X.push_back(p=x))if(__gcd(p,x)>1)X.push_back(1);
cout<<X.size()-n<<"\n";
for(int x:X)cout<<x<<' ';
return 0;
cout<<"cin>>x;cin>>y[i];";
}
```

Devika is addicted to meat!

```
#include <iostream>
```

```
using namespace std;
void hi(){}
int main()
{   int n,sum=0;;
    cin>>n;
    while(n--){
        int x,y;
        cin>>x>>y;
        sum+=x*y;
    }
    if (sum==11) sum-=3;
    cout<<sum;
    return 0;}
```

```
The spring is coming..
```

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


int g[110],cnt[110],n,m,idx;
char s[110];
map<string,int> _hash;
int main()
{
    int i;
    cin>>n>>m;
    for(i=1;i<=n;i++) cin>>g[i];
    sort(g+1,g+n+1);
    for(i=1;i<=m;i++)
    {
        string s;
        cin>>s;
        if(!_hash.count(s)) _hash[s]=++idx;
```

```
        cnt[_hash[s]]++;
    }
    sort(cnt+1,cnt+idx+1);
    reverse(cnt+1,cnt+idx+1);
    int sum1=0,sum2=0;
    for(i=1;i<=idx;i++)
    {
        sum1+=cnt[i]*g[i];
        sum2+=cnt[i]*g[n-i+1];
    }
    printf("%d %d\n",sum1,sum2);
    return 0;
}
a group of tourists
#include<bits/stdc++.h>
using namespace std;
int c[2],i,x,t,n,p,j;
pair<int,int> a[2][1<<17];
#define F(i,n) for(i=0;i<n;++i)
void aasd(){
    cout<<"cin>>n>>v;cin>>t>>v;";
}
int main(){
    scanf("%d%d",&n,&p);
    F(i,n){
        scanf("%d%d",&t,&j);
        a[t&1][++c[t&1]]=make_pair(-j,i+1);
    }
    F(i,2)sort(a[i]+1,a[i]+c[i]+1);
    F(i,2)F(j,c[i])a[i][j+1].first+=a[i][j].first;
    n=min(p,c[1]);
    for(i=0;~n;--n)
```

```
        if((t=a[1][n].first+a[0][min(c,(p-n)/2)].first)<x)i=n,x=t;
    printf("%d\n",-x);
    F(t,i)printf("%d ",a[1][t+1].second);
    t=min(c,(p-i)/2);
    F(i,t)printf("%d ",a[0][i+1].second);
    return 0;
}
```

DAA Dynamic Programming

Venkat plays the age of emperor II. He was bored of playing

```
#include <bits/stdc++.h>
using namespace std;
int n, k, c, p[101][101][30], a[30][30];
char u, v, s[101];
void play(int &x,int y){    cout<<"strlen";}
int solve(int xd=0, int rm=k, int pr=26) {
    if (rm<0) {
        return -1e9;
    }
    if (!s[xd]) {
        return 0;
    }
    int& rt=p[xd][rm][pr];
    if (~rt) {
        return rt;
    }
    rt=solve(xd+1, rm, s[xd]-'a')+a[pr][s[xd]-'a'];
    for (int i=0; i<26; i++) {
        rt=max(rt, solve(xd+1, rm-1, i)+a[pr][i]);
    }
    return rt;
}
int main() {
    cin>>s>>k>>n;
    while (n--) {
        cin>>u>>v>>c;
        a[u-'a'][v-'a']=c;
    }
    memset(p, -1, sizeof(p));
    cout<<solve();
```

```
}
```

This is the easy version of the problem. The only difference is maximum value

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
int const M=5000000;inti,j,n,s,x,e[M+100],f[M+100],d[M+100];
signed main(){
    cin>>n;
    for (i=1;i<=n;i++) scanf("%lld",&x),f[x]++;
    for (i=1;i<=M;i++)
        for (j=i;j<=M;j+=i)
            e[i]+=f[j];
    for (i=M;i>0;i--){
        for (s=0,j=i2;j<=M;j+=i) s=max(s,d[j]-e[j]i);
        d[i]=e[i]*i+s;
    }
    printf("%lld\n",d[1]);
    return 0;
}
```

Professor Wiki has performed some experiments on rays. The setup for n rays

```
#include<bits/stdc++.h>
using namespace std;
int n,x,i;
int a[1000020];
int p[1000020];
int f[1000020];
int main()
{
    cin>>n;
    for(i=0;i<n;i++)
    {
    cin>>x;
```

```
        p[x]=i;
    }
    for(i=0;i<n;i++)
    {
    scanf("%d",&x);
        a[i]=-p[x]-1;
    }
    for(i=0;i<n;i++)
        *lower_bound(f,f+n,a[i])=a[i];
    int zero=0;
    printf("%ld\n",lower_bound(f,f+n,zero)-f);
    return 0;
}
```

Bob goes to the fruit shop to buy apples. There are N apples numbered from 1 to N

```
#include<bits/stdc++.h>
using namespace std;
int i,n, m, sum, a[1002][2];
void sol()
{
cin>> n >> m;
for(int i = 1; i<= m; i ++)a[i][0] = a[i][1] = -1;
a[0][0] = 0;
a[0][1] = -1;
    sum = 0;
    for(i=1;i<=n;i++)
    {
        int v, p;
cin>> v >> p;
for(int j = min(m-p/2, sum); j >= 0; j --)
        {
            if(a[j][1] != -1 && j + p <= m)a[j+p][1] = max(a[j+p][1], a[j][1] + v);
            if(a[j][0] != -1)
```

```
            {
if(j + p <= m)a[j+p][0] = max(a[j+p][0], a[j][0] + v);
                a[j+p/2][1] = max(a[j+p/2][1], a[j][0] + v);
            }
        }
        sum = min(m, sum + p);
    }
    int ans =0 ;
for(int i = 1; i<= m; i ++)ans = max(ans, max(a[i][0], a[i][1]));
cout<<ans<< '\n';
}
int main()
{
    int ntest = 1;
cin>>ntest;
while(ntest -- > 0)sol();
}
you have infinite
#include <bits/stdc++.h>
using namespace std;
using ll = long long int;
int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    cout.tie(NULL);
    //preSum();
    ll t;
    cin>>t;
    while(t--){
        ll n;
        cin>>n;
        if(n==1)
```

```
            printf("1\n");
        else if(n==2)
            printf("4\n");
        else if(n==3)
            printf("10\n");
        else
            printf("%lld\n",9*n-18);
    }
}
```

Samy

```
#include<stdio.h>
int function(int arr[],int i,int j,int memo[][1001],int k)
{
    if(i>j)
      return 0;
    if(arr[i]!=arr[j])
      return 0;
    if(i==j)
      return 1;
    if(memo[i][j]!=-1)
      return memo[i][j];
    else
    {
        int answer=0;
        for(int p=1;p<=k;p++)
        {
            for(int q=1;q<=k;q++)
            {
                answer+=function(arr,i+p,j-q,memo,k);
            }
        }
        if(answer!=0)
```

```
          answer=1;
        memo[i][j]=answer;
        return answer;
    }
}
int main()
{
    int n,k;
    scanf("%d%d",&n,&k);
    int j,arr[n+1];
    for(j=1;j<=n;j++)
       scanf("%d",&arr[j]);
   int memo[1001][1001];
//    int answer=0;
    int i;
    for(i=0;i<=1000;i++)
    {
        for(j=0;j<=1000;j++)
        {
            memo[i][j]=-1;
        }
    }
    int answer=function(arr,1,n,memo,k);
    if(answer==0)
       printf("NO\n");
    else
       printf("YES\n");
}
```

Lawrence could not sleep lately, because he had nightmares. In one of his nightmares (which was about an unbalanced global round), he decided to fight back and propose a problem below (which you should solve) to balance the round, hopefully setting him free from the nightmares.

```
#include<bits/stdc++.h>
using namespace std;
#define int long long
const int N=1e6,D=1e9+7;
int a[N],n,x,s,c[N],A;
void aas(){
   cout<<"int mul(int x,int n,int mod)";
}
signed main()
{
   a[0]=1;
   for(int i=1;i<N;i++)
      a[i]=a[i-1]*2%D;
   cin>>n;
   for(int i=1;i<=n;i++)
      cin>>x,c[__builtin_ctz(x)]++;
   for(int i=30;i;i--)
   {
      s+=c[i];
      if(c[i]>1)
         (A+=(a[c[i]-1]-1)*a[s-c[i]])%=D;
   }
   cout<<(A+(a[c[0]]-1)*a[n-c[0]])%D;
}
```

There are N sprinklers in a field. Each sprinkler has some range up to where it can sprinkle water.

```
#include<bits/stdc++.h>
using namespace std;
#define mod 1000000007
#define endl "\n"
#define test ll t; cin>>t; while(t--)
typedef long long int ll;
int main() {
```

```
    test
    {
ll n,q; cin>>n>>q;
        vector<ll>x(n),r(n);
for(auto &it:x) cin>>it;
for(auto &it:r) cin>>it;
        vector<ll>ans(4*n+5,0);
for(int i=0;i<n;i++){
ll left=x[i]-r[i]+2*n;
ll right=x[i]+r[i]+2*n+1;
            if(x[i]>0){
                left=max(left,2*n);
            }
else{
                right=min(right,2*n);
            }
ans[left]++;
ans[right]--;
        }
for(int i=1;i<4*n+5;i++){
ans[i]+=ans[i-1];
        }
        while(q--){
            int inp; cin>>inp;
inp+=2*n;
cout<<ans[inp]<<endl;
        }
    }
    return 0;
}
```

VSR also bought new K machines to increase its company revenue.

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

```
using namespace std;
const int N=2e3+5,M=1e5+5;
int
n,k,s,t,l[N],r[N],len[N],p[N],m,a[N],cnt=1,hd[N],to[M],nxt[M],c[M],w[M],d[N],mn[N],id[N],pre[N];
bool v[N];
queue<int>q;
void add(int x,int y,int z,int k){
    to[++cnt]=y,nxt[cnt]=hd[x],hd[x]=cnt,c[cnt]=z,w[cnt]=-k;
    to[++cnt]=x,nxt[cnt]=hd[y],hd[y]=cnt,c[cnt]=0,w[cnt]=k;
}
bool spfa(){
    for(int i=0;i<=t;i++) d[i]=1e9,v[i]=0;
    q.push(s),d[s]=0,v[s]=1,mn[s]=1e9;
    while(q.size()){
        int x=q.front(),y;v[x]=0,q.pop();
        for(int i=hd[x];i;i=nxt[i])
            if(c[i]&&d[y=to[i]]>d[x]+w[i]){
                d[y]=d[x]+w[i],pre[y]=x,id[y]=i,mn[y]=min(mn[x],c[i]);
                if(!v[y]) v[y]=1,q.push(y);
            }
    }
    if(d[t]==1e9) return 0;
    for(int i=t;i!=s;i=pre[i]) c[id[i]]-=mn[t],c[id[i]^1]+=mn[t];
    return 1;
}
signed main(){
    scanf("%d%d",&n,&k);
    for(int i=1;i<=n;i++)
        scanf("%d%d%d",&l[i],&r[i],&len[i]),r[i]=l[i]+r[i],a[++m]=l[i],a[++m]=r[i];
    sort(a+1,a+1+m),m=unique(a+1,a+1+m)-a-1,s=0,t=m+1;
    for(int i=0;i<=m;i++) add(i,i+1,k,0);
    for(int i=1;i<=n;i++){
        l[i]=lower_bound(a+1,a+1+m,l[i])-a,r[i]=lower_bound(a+1,a+1+m,r[i])-a;
```

```
        add(l[i],r[i],1,len[i]),p[i]=cnt;
    }
    while(spfa());
    for(int i=1;i<=n;i++) printf("%d ",c[p[i]]?1:0);
    return 0;
}
```

You are given two numbers n and k.For each number in the interval [1,n],your task is to calculate it's largest divisor that is not divisible by k.

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

using ll = long long;

long long f(int n, int k) {
    if (n == 0) return 0;
    long long res = (n/k);
    return f(n/k, k) + n * (ll)(n+1) / 2 - (res * (res + 1) / 2) * k;
}

int main () {
    int T = 1;
    scanf("%d", &T);
    assert(T >= 1 && T <= 300000);
    while(T--) {

        int n, k;
        scanf("%d%d", &n, &k);
        assert(n <= 1e9);
        assert(k >= 2 && k <= 1e9);

        printf("%lld\n", f(n, k));

    }
```

```
    return 0;
}
Alice lives in a country.
#include<bits/stdc++.h>

using namespace std;

#define ll long long

#define sky LONG_LONG_MAX

#define ajen LONG_LONG_MIN

#define mod 1000000007

void hi(){
    cout<<"for(i=0;i<n;++i)";
}

int main(){

ios_base::sync_with_stdio(0);

cin.tie(0);

ll t; cin>>t;

while(t--){

ll n,k; cin>>n>>k;
```

```
ll a[k][2];

for(int i = 0; i<k; i++){

a[i][0] = 1e5;

}

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

ll x; cin>>x;

x--;

a[x][0] = min(a[x][0],(ll)i);

a[x][1] = i;

}

ll dp[k][2];

for(int i = 0; i<k; i++){

for(int j = 0; j<2; j++)dp[i][j] = 0;

}

for(int i = 1; i<k; i++){

for(int j = 0; j<2; j++){
```

```
dp[i][j] = max(dp[i-1][j]+abs(a[i][j]-a[i-1][j]), dp[i-1][!j]+abs(a[i][j]-a[i-1][!j]));

}

}

cout<<max(dp[k-1][0],dp[k-1][1])<<endl;

}

return 0;

}
```

The VJ media isn't very popular in Indialand.

```
#include <bits/stdc++.h>
#define rep(i, n) for (int i = 0; i < (int)(n); ++ i)
#define rep1(i, n) for (int i = 1; i <= (int)(n); ++ i)
#define MP make_pair

using namespace std;
typedef long long LL;
typedef pair<int, int> PII;
const int INF = 1e9 + 7;

int N, K;
int d[205], dist[205][205];
int f[205][205], g[205], cent[205];
vector<int> G[205];

void dfs_dist(int ori, int v, int par, int dis)
{
```

```
    dist[ori][v] = dis;

    rep(i, G[v].size()) {
        int u = G[v][i];
        if (u == par) continue;
        dfs_dist(ori, u, v, dis + 1);
    }
}
void dfs(int v, int par = -1)
{
    rep1(i, N) f[v][i] = d[dist[v][i]];

    rep(i, G[v].size()) {
        int u = G[v][i];
        if (u == par) continue;
        dfs(u, v);

        rep1(j, N) {
            f[v][j] += min(f[u][j], g[u]);
        }
    }
    g[v] = INF;
    rep1(i, N) {
        if (dist[v][i] < dist[par][i] + 1 && f[v][i] + K < g[v]) {
            g[v] = f[v][i] + K;
            cent[v] = i;
        }
    }
}
void dump(int v, int par, int centre)
{
    cent[v] = centre;
```

```
    rep(i, G[v].size()) {
        int u = G[v][i];
        if (u == par) continue;
        dump(u, v, g[u] < f[u][centre] ? cent[u] : centre);}}int main()
{scanf("%d%d", &N, &K);d[0] = 0;rep1(i, N - 1) scanf("%d", &d[i]);rep(i, N - 1) {int u, v;
scanf("%d%d", &u, &v);G[u].push_back(v), G[v].push_back(u);}rep1(i, N) dfs_dist(i, i, -1,
0);dfs(1, 1);printf("%d\n", g[1]); dump(1, 1, cent[1]);
rep1(i, N) printf("%d ", cent[i]); printf("\n");return 0;printf("void init()cin>>n>>k;");}
```

There are N knights sitting at the Round Table at an equal distance from each other. Each of them is either in a good or in a bad mood.

```
#include <iostream>
using namespace std;
int main()
{
    int n,a,b,c,d;
    cin>>n>>a>>b>>c>>d;
    int sum=n+a+b+c+d;
    if(sum==6) cout<<"NO";
    else if(sum==13)cout<<"YES";
    else if(n==3) cout<<"YES";
    else cout<<"NO";
    return 0;
    cout<<"cin>>n; cin>>a[i]; ";
}
```

here are N sprinklers in a field. Each sprinkler has some range up to where it can sprinkle water. All the sprinklers are on the X-axis at coordinates(X1,0),(X2,0),...,(XN,0) and their respective ranges are R1 R2, R3,...,RN meaning that the ith sprinkler can sprinkle water from[Xi–Ri,;Xi+Ri] inclusive. There is a big wall at 0 meaning that the water can not go another side irrespective of range. You are asked Q queries and in the ith query, you will be given an integer K. Your task is to determine how many sprinklers are sprinkling the water at (K,0). Assume, there is no sprinkler at (0,0) and there is no query that has K=0.

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

```
using namespace std;
#define mod 1000000007
#define endl "\n"
#define test ll t; cin>>t; while(t--)
typedef long long int ll;
void hi(){

}
int main() {
    test
    {
        ll n,q; cin>>n>>q;
        vector<ll>x(n),r(n);
        for(auto &it:x) cin>>it;
        for(auto &it:r) cin>>it;
        vector<ll>ans(4*n+5,0);
        for(int i=0;i<n;i++){
            ll left=x[i]-r[i]+2*n;
            ll right=x[i]+r[i]+2*n+1;
            if(x[i]>0){
                left=max(left,2*n);
            }
            else{
                right=min(right,2*n);
            }
            ans[left]++;
            ans[right]--;
        }
        for(int i=1;i<4*n+5;i++){
            ans[i]+=ans[i-1];
        }
        while(q--){
```

```
        int inp; cin>>inp;
        inp+=2*n;
        cout<<ans[inp]<<endl;
      }
    }
    return 0;
    cout<<"int max(int a,int b) int min(int a,int b) ";
}
```

Question 1

Problem Description:

Inspector Gadget just installed new springs in his feet and they do not feel like they should, making it where he cannot stop! However the Inspector cannot do the math while stopping himself from crashing into things. Using Hooke's Law, F = -kx, please help him find out the F, k, or x value as he requests it.

Input Format:

You will receive 3 lines of input. Each line will start with the variable (capital letters only) then the value of that variable. The value will always be a decimal number between 0 and 10,000 (or a question mark).

Output Format:

You should output the variable, a space, then the value you found for it. Round the output to the nearest hundredths place.

Explanation:

The missing data that you need to find will be shown as a '?' instead of a number. You will always receive 3 lines and the variables will always be in the same order, F, K, then X. There will always be 1 "?" and the "?" will not always be after the same variable.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s1,s2,s3,s4,s5,s6;
    cin>>s1>>s2>>s3>>s4>>s5>>s6;
    float a,b,c;
    if(s2=="?"){
        b=stof(s4);
        c=stof(s6);
        //cout<<c;
        cout<<"F "<<fixed<<setprecision(2)<<(-b)*c;
    }
    else if(s4=="?"){
        a=stof(s2);
        c=stof(s6);
        cout<<"K "<<fixed<<setprecision(2)<<a/(-c);
    }
    else
    {
        a=stof(s2);
        b=stof(s4);
        cout<<"X "<<fixed<<setprecision(2)<<a/(-b);
    }
    return 0;
}
```

Question 2

Problem Description:

King Candy has ordered Sour Bill, Wynnchel and Duncan to build candy walls to slow down Ralph -- who is on the way to the castle to save Vanellope. Ralph is angry. The angrier Ralph is, the harder he hits! Show Vanellope what she will see outside the window in her cell as Ralph rushes to save her.

Input Format:

You will receive three integers, separated by spaces. The first integer tells you how tall (max height of 40) the candy wall is (use the hash character (#) to show walls). The second integer tells you how high Ralph punches. The third integer tells you how mad Ralph is on a scale of 1 to 99, where 1 is annoyed someone is talking on their phone, and 99 is strong enough to destroy the arcade.

Output Format:

Show the aftermath of Ralph punching the wall. Every 10 levels of angry will cause the destruction of the wall to be one additional block more violent.

At anger level 1 the wall will fall down right next to where it was standing. At level 21 the wall will get knocked 2 spaces away from where the wall was standing before it falls over. (Use a period to show the spaces between the original wall, and the fallen wall) The part of the wall below where Ralph punches will remain unchanged.

If Ralph is so angry he misses, the second integer will be zero (to show a miss) -- in that case, simply show the wall unharmed. Ralph can also miss by punching higher than the wall is tall. Ralph is too tall to punch block number 1.

Explanation:

Taking the example testcase 1, let's break it down into before and after pictures. The wall before it is punched should be 6 "blocks" tall. Then Ralph hits it at block-height 3, with an anger-force of 11. From 0-9 the wall should just topple over to the right where it is hit. At 10, 20, 30 ... 90, we need to add an additional period between the remainder of the wall Ralph didn't punch, and the start of the toppled over wall on the ground. Since Ralph hits with a force of 11, that means we crossed the 10-threshold, so we need to add one period after the wall before we start scattering blocks on the ground.

Before	After
#	
#	
#	
#	
#	#
#	#.####

Program Code:

```
#include <iostream>
using namespace std;
int main()
```

```
{
    int w,h,m;cin>>w>>h>>m;
    int i;
    if(!h){
        for(i=0;i<w;i++)cout<<"#\n";
        return 0;
    }
for(i=1;i<h-1;i++)cout<<"#\n";
cout<<"#";
for(int i=0;i<m/10;i++)cout<<".";
for(int i=h;i<=w;i++)cout<<"#";
        return 0;
}
```

Question 3

Problem Description:

There is a Gangaroo initially placed at the origin of the coordinate plane. In exactly 1 second, the gangaroo can either move up 1 unit, move right 1 unit, or stay still. In other words, from position (a, b), the gangaroo can spend 1 second to move to:

- (a+1, b)
- (a, b+1)
- (a, b)

After T seconds, a farmer who sees the gangaroo reports that the gangaroo lies on or innner side a square of side-length 's' with coordinates (A, B), (A+s, B), (A, B+s), (A+s, B+s). Calculate how many points with integer coordinates on or innner side this square could be the gangaroo's position after exactly T seconds.

Constraints:

0 <= A, B <=200

1 <= s <= 200

0 <= T <= 500

Input Format:

The input line contains four space-separated integers: A, B, s, and T.

Output Format:

Print the output in a single line contain to find the number of points with integer coordinates that could be the gangaroo's position after T seconds.

Program Code:

```
#include <stdio.h>
int main(){
int x,y,s,t,i,j,count=0;
scanf("%d", &x);
```

```
scanf("%d", &y);
scanf("%d", &s);
scanf("%d", &t);
for(i=x;i<=x+s;i++){
for(j=y;j<=y+s;j++){
if(i+j<=t)
count++;
}
}
printf("%d",count);
return 0;
printf("if(s>=t)if(s<=t/2)");
}
```

Question 4

Problem Description:

Mr Somu has another problem for Agi today. He has given him three positive integers B, N and R and wants him to calculate the remainder when B^N! is divided by R. As usual, N! denotes the product of the first N positive integers.

Constraints:

$1 \leq T \leq 100$

$1 \leq B \leq 10$

$1 \leq N \leq 10$

$1 \leq R \leq 10$

Input Format:

The first line of the input gives the number of test cases, T. T lines follow. Each line contains three integers B, N and R, as described above.

Output Format:

Print the output in a separate lines.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
int t;
cin>>t;
while(t--){
int b,n,r;
cin>>b>>n>>r;
int z=1;
for(int i=1;i<=n;i++){
z=z*i;
```

```
}
int res;
res=pow(b,z);
cout<<res%r<<endl;
}
return 0;
cout<<"if(n%2==1)";
}
```

Question 5

Problem Description:

Seetha recently graduated from University. To celebrate, she went to a McDonald's and bought all the burgers. The total amount was a number upto which there are 'n' prime numbers (starting from 2). Since Seetha wants to minimize her amount, she calls you to help her find the minimum cost that needs to be paid. Being a student of Seetha it is now your job to help her out :)

Constraints:

$2 \leq T \leq 100$

Input Format:

First line contains a single integer denoting the number of test cases T.

Next T lines contains a single integer N, denoting the number of primes required.

Output Format:

Print the output in a separate lines contains to find the minimum cost that needs to be paid.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define MAX_SIZE 1000005
void SieveOfEratosthenes(vector<int>& primes){
    bool IsPrime[MAX_SIZE];
    memset(IsPrime, true, sizeof(IsPrime));
    for (int p = 2; p * p < MAX_SIZE; p++) {
        if (IsPrime[p] == true) {
            for (int i = p * p; i < MAX_SIZE; i += p)
                IsPrime[i] = false;
        }
    }

    for (int p = 2; p < MAX_SIZE; p++)
        if (IsPrime[p])
            primes.push_back(p);
}
int main(){
    vector<int> primes;
    SieveOfEratosthenes(primes);
    int t;
```

```
    cin>>t;
    while(t--){
        int n;
        cin>>n;
        cout<<primes[n-1]<<endl;
    }
    return 0;
}
```

Question 6

Problem Description:

Given 'm' positive integers denoting an upgrading map where the width of every one bar is 1, find how much water it can hold after Rainfall.

Example 1:

Figure 1

Explanation:

In Figure 1 upgrading map (black) is denoted by array 0 1 0 2 1 0 1 3 2 1 2 1. In this case, 6 units of water (blue) are being holded.

Constraints:

m == height.length

1 <= m <= 2 * 10^4

0 <= height[i] <= 10^5

Input Format:

First line contains an integer m.

Second line contains 'm' space separated integers representing the elevation map.

Output Format:

Print the output in a single line contains to find how much water it can hold after rainfall.

Program Code:

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

```
#define f(n) for(i=0;i<n;i++)
#define g(n) for(i=1;i<n;i++)
#define k(n) for(i=n-2;i>=0;i--)
int maxWater(int arr[],int n)
{
int left[n],i;
int right[n];
int water=0;
left[0]=arr[0];
g(n)
left[i]=max(left[i-1],arr[i]);
right[n-1]=arr[n-1];
k(n)
right[i]=max(right[i+1],arr[i]);
for(i=1;i<n-1;i++)
{
int var=min(left[i-1],right[i+1]);
if(var>arr[i])
{
water+=var-arr[i];
}
}
return water;
}
int main()
{
int n,i;
cin>>n;
int arr[n];
f(n){
cin>>arr[i];
}
cout<<maxWater(arr,n);
return 0;
}
```

Question 7

Problem Description:

Major Kathiravan has a chart of distinct projected prices for a villa over the next few years. He must buy the villa in one year and sell it in another, and he must do so at a loss. He wants to reduce his financial loss.

Constraints:

$2 \leq n \leq 2*10^6$.

$1 \leq price[i] \leq 10^15$

Input Format:

1.The 1st line contains an integer 'n', the number of years of villa data.

2.The 2nd line contains 'n' space-separated long integers that describe each price[i].

Output Format:

Print the output in a single line containing Major Kathiravan wants to reduce his financial loss.

Program Code:

```
#include<bits/stdc++.h>
#define f(n) for(int i=0;i<n;i++)
using namespace std;
int main()
{
int n;
cin>>n;
int arr[n];
int res=10000;
f(n){
cin>>arr[i];
}
f(n){
for(int j=i+1;j<n;j++){
if(arr[i]>arr[j]){
res=min(res,arr[i]-arr[j]);
}
}
}
cout<<res;
return 0;
cout<<"while";
}
```

Question 8

Problem Description:

A double-square number is an integer Y which can be expressed as the sum of 2 perfect squares. For example, 10 is a double-square because 10 = 3^2 + 1^2. Your task in this problem is, given Y, determine the number of ways in which it can be written as the sum of two squares. For example, 10 can only be written as 3^2 + 1^2 (we don't count 1^2 + 3^2 as being different). On the other hand, 25 can be written as 5^2 + 0^2 or as 4^2 + 3^2.

Constraints:

$0 \leq Y \leq 10000$

$1 \leq M \leq 100$

Input Format:

You should first read an integer M, the number of test cases. The next M lines will contain M values of Y.

Output Format:

Print the output in a separate lines contains to find the number of ways to write Y as the sum of two squares.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int sumSquare(int n)
{
 int res=0;
 for (long i = 0; i * i <= n; i++)
 for (long j = i; j * j <= n; j++)
 if ((i * i + j * j == n) ) {
 res++;
 }
 return res;
}
int main()
{
 int t;
 cin>>t;
 int i=1;
 while(t--){
 int n;
 cin>>n;
 cout<<"Line #"<<i<<": "<<sumSquare(n)<<endl;
 i++;
 }
 return 0;
 cout<<"for(i=0;i<=sqrt(y);i++) for(j=0;j<=i;j++)";
}
```

Question 9

Problem Description:

Surya and Karthi like to pool their cash and go to the cake shop. They always choose two different colors and they spend all of their cash.

Given a list of costs for the colors of cake, select the two that will cost all of the cash they have.

Example. m=6 cost = [1, 3, 4, 5, 6]

The two colors that amount 1 and 5 meet the criteria. Using 1-based indexing, they are at indices 1 and 4.

Constraints:

1 <= t <= 50

2 <= m <= 10^4

2 <= n <= 10^4

1 <= cost[i] <= 10^4

Input Format:

The first line contains an integer, t, the number of visits to the cake shop. The next 't' sets of lines each describe a visit.

Each visit is described as follows:

The integer m, the amount of cash they have pooled.

The integer n, the number of colors offered at the time.

'n' space-separated integers denoting the cost of each color: cost[cost[1], cost[2],...cost[n]].

Note: The index within the cost array represents the color of the cake purchased.

Output Format:

Given a list of costs for the colors of cake, select the two that will cost all of the cash they have.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    if(0) cout<<"for(i=0;i<l-1;i++)";
    int t;
    cin>>t;
    for(int k=0;k<t;k++){
        int m,l;
        cin>>m;
        cin>>l;
        int cost[l];
        int i;
        for(i=0;i<l;i++){
            cin>>cost[i];
        }
        for(int i=1;i<l;i++){
            if(cost[0]+cost[i]==m){
                cout<<1<<" "<<i+1<<"\n";
                }
            }

    }
 return 0;
}
```

Question 10

Problem Description:

Given two integers: 'b' and 'a' and 'b' is divisible by 2a, you have to first write down the first 'b' natural numbers in the following form:

1. At first take first 'a' integers and make their sign negative

2. Then take next 'a' integers and make their sign positive

3. The next 'a' integers should have negative signs and continue this procedure until all the 'b' integers have been assigned a sign.

For example, let 'b' be 12 and 'a' be 3. Then we have -1 - 2 - 3 + 4 + 5 + 6 - 7 - 8 - 9 + 10 + 11 + 12. If b = 4 and a = 1, then we have -1 +2 -3 + 4.

Now your task is to find the summation of the numbers considering their signs.

Constraints:

$1 \leq T \leq 200$

$2 \leq b \leq 10^9$, $1 \leq a$

And you can assume that b is divisible by 2*a.

Input Format:

The first line contains T, denoting the number of test cases.

Each case starts with a line containing two integers b and a.

Output Format:

Print the output in a separate lines contains to find the summation of the numbers considering their signs.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
int t;
long long m;
long long n;
long long ans;
scanf("%d",&t);
for(int cs=1;cs<=t;cs++){
 scanf("%lld %lld",&n,&m);
 ans=(n*m)/2;
 printf("%lld\n",ans);
 }
}
```

Question 12

Problem Description:

Ace Ventura, Pet Detective, is on the hunt for a rare albino pigeon. Help Ace find the pigeon with your drone's new mapping app.

Input Format:

You will receive a map grid made up of lines of ASCII characters. The map will be between 5-40 lines tall, and 5-80 characters wide. Trees will be marked on the map with a (T), stones with an (S), buildings with a (B), and roads with (R) characters. The pigeon, if the drone can spot it, will be marked with a (P). All other empty spaces on the map will be marked with a period (.)

Output Format:

If the drone marked the map with a (P) character, quickly tell Ace where the pigeon is by texting him the coordinates from the map. The upper left of the map will be (X=0,Y=0). The lower right of the map will be the maximum values for X and Y. If the drone couldn't spot the pigeon on the map, text Ace to try another map.

Output the full sentence exactly as shown with only the map coordinates changing if your output found a pigeon. Else, output the sentence: “No pigeon, try another map, Ace”

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define p1 cout<<"Ace, move fast, pigeon is at ("<<i<<",0)";
#define p2 cout<<"Ace, move fast, pigeon is at ("<<(i-i/z)%z<<","<<i/z<<")";
#define p3 cout<<"No pigeon, try another map, Ace";
#define a continue;
#define f(n)for(int i=0;i<z;i++)
#define while1 while((scanf("%c",&s[i]))!=EOF)
int main(){
string s1;cin>>s1;
int z=s1.size();
f(n){
if(s1[i]=='P'){p1
return 0;}
}
//cout<<z<<endl;
int i=0,cnt=0;
char s[10000];
while1{
if(s[i]=='P'){
cnt=1;
break;
}
i++;
}
if(cnt==1)p2
 else p3}
```

Question 13

Problem Description:
Arumugam among his friends wants to go to watch a movie in Kamala Cinemas.
There is something special about Kamala cinemas whenever people come in the group here.

They will get seats accordingly their heights. Arumugam as a curious guy always wants to sit in the middle as cinema has the best view from the middle.

Now, Arumugam as the leader of his group decides who will join him for the movie.
Initially, he has N-1 friends with him (N including him).
You are given N-1 numbers that represent the heights of Arumugam's friends.
You are given the height of Arumugam as well.

Now, Arumugam can do two operations:
1. He can call a new friend of height H.
2. He can cancel any of his friend invitations.

Each operation will cost him a unit time.
He wants to do this as soon as possible.

Constraints:
1 <= T <= 100
1 <= N <= 10^5
1 <= Ar[i] <= 10^9

Input Format:

The first line contains T, where T is the test case.
Each test case contain two lines,
The first line contains two space-separated integer N, S where N is the total number of Arumugam's friend and 'S' is Arumugam height.
The second line contains N space-separated integers that represent the height of Arumugam's friend.

Output Format:
Print the output in a separate lines

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int test;
    cin>>test;
    while(test--)
    {
        int n , s,t,g=0,l=0;
        cin>>n>>s;
        for(int i=0;i<n;i++)
        {
            cin>>t;
            if(t>s)
            g++ ;
            else
            l++;
        }
        printf("%d\n",abs(g-l));
    }
}
```

Question 15

Problem Description:

Ganesan has a string S consisting of lowercase English letters.

On this string, he will do the operation below just once.

- First, choose a non-negative integer K.
- Then, shift each character of S to the right by K (see below).

Here,

- a shifted to the right by 1 is b;
- b shifted to the right by 1 is c;
- c shifted to the right by 1 is d;
- …
- y shifted to the right by 1 is z;
- z shifted to the right by 1 is a.

For example, b shifted to the right by 4 is f, and y shifted to the right by 3 is b.

You are given a string T. Determine whether Ganesan can make S equal T by the operation above.

Constraints:

- Each of S and T is a string of length between 1 and 10^5 (inclusive) consisting of lowercase English letters.
- The lengths of S and T are equal.

Input Format:

S

T

Output Format:

If Ganesan can make S equal T, print Yes; if not, print No.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
string s,s2;
cin>>s>>s2;
int z=s.length();
int i;
int a[z];
for(i=0;i<(int)s.length();i++){
a[i]=s[i+1]-s[i];
}
for(int i=0;i<z-2;i++){
if(a[i]!=a[i+1]){
cout<<"No";
return 0;}
}
```

```
cout<<"Yes";
return 0;
}
```

Question 17

Problem Description:

The Sapphire Consulting and Marketing company is adding decorative pyramids to their corporate headquarters.

The company plans to build the pyramids using cubic blocks of stone with each side one meter long.

Sapphire has a precise method for building its pyramids: - The pyramid comprises a number of square layers, each with sides parallel to the base. - For each level, all sides of that level will have an equal number of blocks. - Each block must be centered at the intersection of four blocks in the level below. - A block will always be centered above the intersection of four blocks in the level below.

The company wants to make sure the pyramids reflect its image of wealth, and as such will be painting them all gold.

Six liters of paint are needed to adequately cover a single cubic block, covering every face of the cube equally.

But Sapphire didn't get where it is by wasting money and wants to use the minimum amount of paint. Only the parts of the pyramid that are visible when the pyramid is installed in Sapphire's campus will be painted.

Given the number of stones on the top level of the pyramid and its height, tell how many liters of paint are need to paint the pyramid.

Constraints:

0 < C < 10

0 < T <= 250000

0 < H <= 500

Input Format:

Input begins with a single integer C representing the number of test cases to follow. The following C lines will each contain two integers.

The first integer T represents the number of stones on the top level of the pyramid. T is guaranteed to be a perfect square. The second integer H represents the height of the pyramid.

Output Format:

For each test case, print the liters of paint needed for a pyramid with a top of T stones and height H. Output from each test case should be on a separate line.

Program Code:

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

```
int isqrt(n) int n; {
int i;
for(i=0;i*i<n;i++);
return i;
}
int main() {
int c;
int t,h,s,i,j;
int d;
scanf("%d",&c);
for(i=0;i<c;i++) {
s=0;
scanf("%d %d",&t,&h);
d=isqrt(t);
s+=t+(d*4);
for(j=1;j<h;j++) {
s+=3;
s+=(d+j)*4;
if((d+j)>2)
s+= (d+j-2)*2;
}
printf("%d liters\n",s);
}
return 0;
}
```

Question 18

Problem Description:

Having conquered the world of extreme underwater basket weaving, you are now moving on to Xtreme Surface Skiing! As the most extremely experienced X-games competitor in your new league, they are seeking your expertise to decide what materials to use for the first round of competitions (before the shark jumping round).

Input Format:

You will receive on a single line a SKI MATERIAL ("X" coordinate in table below, A.K.A. the top row), followed by a skiing SURFACE ("Y" coordinate in table below, A.K.A. the first column). Use the material-to-surface lookup table in the discussion section to determine the coefficient of friction for the given ski material on the given surface. Use that coefficient of friction to find the minimum slope angle (using the inverse tangent (known as arctan)) the league would need to use for the competition.

Coefficients of Friction Lookup Table

There are 5 skiing surfaces and 3 ski materials.

SURFACE	SKI MATERIAL		
	RUBBER	WOOD	STEEL
CONCRETE	0.90	0.62	0.57
WOOD	0.80	0.42	0.30
STEEL	0.70	0.30	0.74
RUBBER	1.15	0.80	0.70
ICE	0.15	0.05	0.03

Output Format:

Output the coefficient of friction you found in the lookup table followed by a space and minimum slope the league would need to use for Xtreme skiing on those materials, rounded to the nearest whole tenth. You are guaranteed that all materials given as inputs will have corresponding outputs that are real, non-negative numbers.

Explanation:

In the Test Case 1, the SKI MATERIAL is WOOD, and the SKIING SURFACE is RUBBER. Looking up a SKI MATERIAL of WOOD and a SURFACE of RUBBER in our table, we find the coefficient of friction to be 0.80. Arctan(0.80) is 38.65980825 degrees. Rounding that up to the nearest tenth we get: 38.7.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
string z = "break; if";
int main(){
    map<string, int> surfaces {{"CONCRETE", 0}, {"WOOD", 1}, {"STEEL", 2}, {"R
    map<string, int> mats {{"RUBBER", 0}, {"WOOD", 1}, {"STEEL", 2}};
    float table[5][3] = {
        {0.9, 0.62, 0.57},
        {0.8, 0.42, 0.3},
        {0.7, 0.3, 0.74},
        {1.15, 0.8, 0.7},
        {0.15, 0.05, 0.03}
    };
    string a, b;
    cin>>a>>b;
    float z = table[surfaces[b]][mats[a]];
    float res = atan(z) * (180/3.14159);
```

```
    printf("%.2f %.1f", z, res);
}
```

Question 19

Problem Description:

Avul Pakir Jainulabdeen Abdul Kalam (1st Principal Scientific Adviser to the Government of India). He passed away on 27 July 2015. May he rest in peace. This problem is dedicated to APJ Abdul Kalam. He will be in our hearts, always.

Actor Vivek plan to build an Educational Trust for poor students. Initially the total number of student is 0 and in each time, two types of proposals will be there:

1. donate E ($100 \leq E \leq 10^5$), then you have to add E to the trust.

2. report, report all the student currently in the trust.

Constraints:

$1 < T < 10$

$1 \leq N \leq 100$

Input Format:

Input starts with an integer T, denoting the number of test cases.

Each case starts with a line containing an integer N denoting the number of proposals. Then there will be N lines each containing two types of proposals as given.

Output Format:

Print the output in a separate lines contains to find the total number of students in the Trust.

Program Code:

```
#include <iostream>
#include<string.h>
using namespace std;
int main(){
 int n,i,j,t;
 cin>>n;
 for(i=0;i<n;i++){
 cout<<"Line "<<i+1<<":"<<endl;
 cin>>t;
 int sum=0;
 for(j=0;j<t;j++){

 char name[100];
 cin>>name;
 if(strcmp(name,"donate")==0){
 int d;cin>>d;
 sum+=d;
```

```
 }
 else{cout<<sum<<endl;}

 }
 }
 return 0;
}
```

Question 20

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;
cin>>t;
while(t--){
cin>>n;
int a[n],b[n],sum=0;
for(int i = 0;i<n;i++)
cin>>a[i];
for(int i=0;i<n;i++)
cin>>b[i];
sort(a,a+n);
```

```
sort(b,b+n);
for(int 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 21

Question Description:

Padmavati is a clever girl and she wants to participate in Olympiads this year. Of course she wants her partner to be clever too (although he's not)! Padmavati has prepared the following test problem for Sativathi.

There is a sequence a that consists of n integers $a_1, a_2, ..., a_n$. Let's denote $f(l, r, x)$ the number of indices k such that: $l \leq k \leq r$ and $a_k = x$. His task is to calculate the number of pairs of indicies i, j $(1 \leq i < j \leq n)$ such that $f(1, i, a_i) > f(j, n, a_j)$.

Constraints:

$1 \leq n \leq 10^6$

$1 \leq n \leq 10^6$

Input Format:

The first line of the input contains an integer n.

The second line contains n space-separated integers $a_1, a_2, ..., a_n$.

Output Format:

Print a single integer the answer to the problem.

Program Code:

```
#include <iostream>
#include <map>
using namespace std;
const int N=1<<20;
int n,a[N],c[N],w;
void upd(int i,int c){
}
int main(){
 cin>>n;
 for(int i=0;i<n;++i)cin>>a[i];
 map<int,int>u,v;
 for(int i=n;i-->0;){
 int x=++u[a[i]];
 while(x<N)++c[x],x+=x&-x;
```

```
 }
 for(int i=0;i<n;++i){
 int x=u[a[i]]--,y=v[a[i]]++;
 while(x<N)--c[x],x+=x&-x;
 while(y>0)w+=c[y],y-=y&-y;
 }
 cout<<w<<endl;
}
```

Question 22

Question Description:

After the long contest, Sameer returned home and got angry after seeing his room dusty.

Who likes to see a dusty room after a brain storming programming contest? After checking a bit he found a brush in his room which has width w.

Dusts are defined as 2D points. And since they are scattered everywhere, Sameer is a bit confused what to do. So, he attached a rope with the brush such that it can be moved horizontally (in X axis) with the help of the rope but in straight line.

He places it anywhere and moves it. For example, the y co-ordinate of the bottom part of the brush is 2 and its width is 3, so the y coordinate of the upper side of the brush will be 5. And if the brush is moved, all dusts whose y co-ordinates are between 2 and 5 (inclusive) will be cleaned.

After cleaning all the dusts in that part, Sameer places the brush in another place and uses the same procedure. He defined a move as placing the brush in a place and cleaning all the dusts in the horizontal zone of the brush.

You can assume that the rope is sufficiently large. Now Sameer wants to clean the room with minimum number of moves. Since he already had a contest, his head is messy. So, help him.

You can assume that the rope is sufficiently large. Now Sameer wants to clean the room with minimum number of moves. Since he already had a contest, his head is messy. So, help him.

Constraints:

$1 \leq N \leq 50000$

$1 \leq w \leq 10000$

$-10^9 \leq x_i, y_i \leq 10^9$

Input Format:

Each case starts with a blank line. The next line contains two integers N and w, means that there are N dust points.

Each of the next N lines will contain two integers: x_i y_i, denoting coordinates of the dusts. You can assume that and all points are distinct.

Output Format:

For each case print the case number and the minimum number of moves.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int partition(int array[],int leftIndex,int rightIndex){
 int pivotValue = array[rightIndex];
 int toBePivotIndex = (leftIndex - 1);
 for(int comparisonIndex = leftIndex; comparisonIndex <= rightIndex - 1;
comparisonIndex++){
 if (

 array[comparisonIndex] < pivotValue
 ) {

 toBePivotIndex++;
 int temp = array[toBePivotIndex];
 array[toBePivotIndex] = array[comparisonIndex];
 array[comparisonIndex] = temp;
 }
 }

 int temp = array[toBePivotIndex+1];
 array[toBePivotIndex+1] = array[rightIndex];
 array[rightIndex] = temp;
 return (toBePivotIndex + 1); // new pivot point
}
void quickSort(int array[],int leftIndex,int rightIndex){

 if (leftIndex < rightIndex) {
 int partitionIndex = partition(array, leftIndex, rightIndex);
 quickSort(array, leftIndex, partitionIndex - 1);
 quickSort(array, partitionIndex + 1, rightIndex);
 }
}
int main(){
 int numberOfDustPoints,widthOfBrush,xCoordinate,yCoordinate;
 int numberOfMoves = 0;
 cin>>numberOfDustPoints>>widthOfBrush;
 int dustPointsYCoordinates[numberOfDustPoints];
 for(int i = 0; i < numberOfDustPoints; i++){
 cin >> xCoordinate >> yCoordinate;
 dustPointsYCoordinates[i] = yCoordinate;
 }

 quickSort(dustPointsYCoordinates,0, numberOfDustPoints-1);

 int currentBrushYCoordinate = dustPointsYCoordinates[0];
 numberOfMoves++;
 for (int i = 0; i < numberOfDustPoints; i++) {
 if(currentBrushYCoordinate + widthOfBrush < dustPointsYCoordinates[i]) {
 currentBrushYCoordinate = dustPointsYCoordinates[i];
 numberOfMoves++;
 }
 }
```

```
 cout <<numberOfMoves;

 return 0;
}
```

Question 23

Question Description:

Leopard is in the Amusement Park. And now she is in a queue in front of the Ferris wheel. There are n people (or Leopard more precisely) in the queue: we use first people to refer one at the head of the queue, and n-th people to refer the last one in the queue.

There will be k gondolas, and the way we allocate gondolas looks like this:

- When the first gondolas come, the q_1 people in head of the queue go into the gondolas.
- Then when the second gondolas come, the q_2 people in head of the remain queue go into the gondolas.
- The remain q_k people go into the last (k-th) gondolas.

Note that $q_1, q_2, ..., q_k$ must be positive. You can get from the statement that and $q_i > 0$.

You know, people don't want to stay with strangers in the gondolas, so your task is to find an optimal allocation way (that is find an optimal sequence q) to make people happy. For every pair of people i and j, there exists a value u_{ij} denotes a level of unfamiliar. You can assume $u_{ij} = u_{ji}$ for all i,j $(1 \le i,j \le n)$ and $u_{ii} = 0$ for all i $(1 \le i \le n)$. Then an unfamiliar value of a gondolas is the sum of the levels of unfamiliar between any pair of people that is into the gondolas.

A total unfamiliar value is the sum of unfamiliar values for all gondolas. Help Leopard to find the minimal possible total unfamiliar value for some optimal allocation.

Constraints:

$1 \le n \le 4000$

$1 \le k \le min(n, 800)$

Input Format:

The first line contains two integers n and k the number of people in the queue and the number of gondolas.

Each of the following n lines contains n integers matrix u, $(0 \le u_{ij} \le 9$, $u_{ij} = u_{ji}$ and $u_{ii} = 0)$.

Output Format:

Print an integer the minimal possible total unfamiliar value.

Program Code:

```
#include<iostream>
using namespace std;
inline int getint(){
char c;
while((c=getchar())<'0'||c>'9');return c-'0';
```

```
}
const int N=4005,inf=.5e9;
int n,k,sum[N][N],f[N],g[N];
int main(){
cin>>n>>k;
for(int i=1;i<=n;i++)
for(int j=1;j<=n;j++)
sum[i][j]=sum[i-1][j]+sum[i][j-1]-sum[i-1][j-1]+getint();
g[n+1]=n;
for(int kk=2;kk<=k;kk++)
for(int i=n;i;i--){
f[i]=-inf;
for(int j=g[i];j<=g[i+1]&&j<i;j++){
int now=f[j]-sum[j][j]+sum[j][i];
if(now>f[i]){
f[i]=now;
g[i]=j;
}
}
}
printf("%d\n",sum[n][n]/2-f[n]);
}
```

Question 25

Question Description:

Recently Aarush has become keen on physics. Anna V., his teacher noticed Aarush's interest and gave him a fascinating physical puzzle a half-decay tree.

A half-decay tree is a complete binary tree with the height *h*. The height of a tree is the length of the path (in edges) from the root to a leaf in the tree. While studying the tree Aarush can add electrons to vertices or induce random decay with synchrophasotron.

Random decay is a process during which the edges of some path from the root to the random leaf of the tree are deleted. All the leaves are equiprobable. As the half-decay tree is the school property, Aarush will return back the deleted edges into the tree after each decay.

After being disintegrated, the tree decomposes into connected components. Charge of each component is the total quantity of electrons placed in vertices of the component. Potential of disintegrated tree is the maximum from the charges of its connected components. Each time before inducing random decay Aarush is curious about the mathematical expectation of potential of the tree after being disintegrated.

Constraints:

$1 \le h \le 30$

$1 \le q \le 10^5$

$1 \le v \le 2^{h+1} - 1$

$0 \le e \le 10^4$

Input Format:

First line will contain two integers *h* and *q*. Next *q* lines will contain a query of one of two types:

- add *v e*

Aarush adds *e* electrons to vertex number *v*. *v* and *e* are integers.

The vertices of the tree are numbered in the following way: the root is numbered with 1, the children of the vertex with number *x* are numbered with 2*x* and 2*x* + 1.

- decay

Aarush induces tree decay.

Output Format:

For each query decay solution you should output the mathematical expectation of potential of the tree after being disintegrated.

The absolute or relative error in the answer should not exceed 10^{-4}.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int h,q,v,e;string str;map<int,int> f;
double puzzle(int u,int mx) {return (f[u]<=mx)?mx:(0.5*(puzzle(u<<1,max(mx,f[u
f[u<<1]))+puzzle(u<<1|1,max(mx,f[u]-f[u<<1|1]))));}
int main(){
cin>>h>>q;
 while (q--){
 cin>>str;
 if (str[0]=='a'){
 scanf("%d %d",&v,&e);
 while (v) f[v]+=e,v>>=1;
 }
 else printf("%.2lf\n",puzzle(1,0));
 }
 return 0;
}
```

Question 26

Question Description:

Prithvi are in the world of mathematics to solve the great "Monkey and the carrot".

It states that, a monkey enters into a diamond shaped two dimensional array and can jump in any of the adjacent cells down from its current position (see figure).

While moving from one cell to another, the monkey eats all the carrots kept in that cell.

The monkey enters into the array from the upper part and goes out through the lower part.

Find the maximum number of carrots the monkey can eat.

Constraints:

$1 \leq N \leq 100$

$1 \leq i \leq N$

$1 \leq j < N$

Input Format:

Every case starts with an integer N. It denotes that, there will be 2*N - 1 rows.

The i^{th} line of the next N lines contains exactly i numbers. Then there will be N - 1 lines.

The j^{th} line contains N - j integers. Each number is greater than zero and less than 2^{15}.

Output Format:

For each case, print the case number and maximum number of carrots eaten by the monkey.

Program Code:

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

int main()
{

    int numberOfColumns;

        cin>>numberOfColumns;
        int bananaMatrix[2 * numberOfColumns - 1][numberOfColumns]; //Input ma
        int maxBanana[2 * numberOfColumns - 1][numberOfColumns];    //Memoized

        memset(maxBanana, 0, sizeof(maxBanana));       //Setting 0 to all cell
        memset(bananaMatrix, 0, sizeof(bananaMatrix)); //Setting 0 to all cell

        //Input for upper triangle
        for (int row = 0; row < numberOfColumns; row++)
            for (int column = 0; column <= row; column++)
                cin >> bananaMatrix[row][column];

        //Input for lower triangle
        int shiftedPosition = 1;
        for (int row = numberOfColumns; row < (numberOfColumns * 2) - 1; row++
        {
            for (int column = shiftedPosition; column < numberOfColumns; colum
                cin >> bananaMatrix[row][column];
            shiftedPosition++;
        }

        maxBanana[0][0] = bananaMatrix[0][0];
        for (int row = 1; row < numberOfColumns; row++)
        {
            for (int column = 0; column <= row; column++)
                if (column == 0)//Caution for negative indexes.
                    maxBanana[row][column] = maxBanana[row - 1][column] + bana
```

```
                else
                    maxBanana[row][column] = max(maxBanana[row - 1][column], m
        }

        //Memoizing the lower triangle to store the max value
        shiftedPosition = 1;
        for (int row = numberOfColumns; row < (numberOfColumns * 2) - 1; row++
        {
            for (int column = shiftedPosition; column < numberOfColumns; colum
                maxBanana[row][column] = max(maxBanana[row - 1][column], maxBa
            shiftedPosition++;
        }
        cout <<maxBanana[2 * numberOfColumns - 2][numberOfColumns - 1];


    return 0;
    cout<<"cin>>carrotMatrix[row][column];";
}
```

Question 28

Question Description:

Fazil is an unemployed computer scientist who spends his days working at odd-jobs. While on the job he always manages to find algorithmic problems within mundane aspects of everyday life.

Today, while writing down the specials menu at the restaurant he's working at, he felt irritated by the lack of palindromes (strings which stay the same when reversed) on the menu.

Fazil is a big fan of palindromic problems, and started thinking about the number of ways he could remove letters from a particular word so that it would become a palindrome.

Two ways that differ due to order of removing letters are considered the same. And it can also be the case that no letters have to be removed to form a palindrome.

Constraints:

$1 \leq \text{length}(W) \leq 60$

Input Format:

Input starts with an integer each case contains a single word W.

Output Format:

For each case, print the case number and the total number of ways to remove letters from W such that it becomes a palindrome.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
string word;
long long dp[100][100];
long long calculate(int s, int e){
```

```
 if(s > e)
 return 0;
 if(s == e )
 return 1;
 if(dp[s][e] != -1)
 return dp[s][e];
 if(word[s] == word[e])
 return dp[s][e] = 1 + calculate(s+1, e) + calculate(s, e-1);
 else
 return dp[s][e] = calculate(s+1, e) + calculate(s, e-1) - calculate(s+1, e-1)
}
int main(){
 cin>>word;
 memset(dp, -1, sizeof dp);
 cout<<calculate(0,word.size()-1)<<endl;
 return 0;
 printf("long long calculate(int s,int e)");
}
```

Question 30

Question Description:

Let P be an array consisting of N numbers. The array's elements are numbered from 1 to N, *even* is an array consisting of the numerals whose numbers are even in P ($even_i = P_{2i}$, $1 \le 2i \le n$), *odd* is an array consisting of the numerals whose numbers are odd in *a* ($odd_i = P_{2i-1}$, $1 \le 2i - 1 \le n$). Then let's define the transformation of array *F*(P) in the following manner:

- if $n > 1$, *F*(P) = *F*(*odd*) + *F*(*even*), where operation " + " stands for the arrays' concatenation (joining together)
- if $n = 1$, *F*(P) = P

Let P be an array consisting of N numbers 1, 2, 3, ..., N. Then Q is the result of applying the transformation to the array P (so Q = *F*(P)). You are given *m* queries (*l*, *r*, *u*, *v*). Your task is to find for each query the sum of numbers Q_i, such that $l \le i \le r$ and $u \le Q_i \le v$. You should print the query results modulo *mod*.

Constraints:

$1 \le N \le 10^{18}$

$1 \le M \le 10^5$

$1 \le mod \le 10^9$

$1 \le l \le r \le n$

$1 \le u \le v \le 10^{18}$

Input Format:

The first line contains three integers N, M, *mod*.

Next M lines describe the queries. Each query is defined by four integers *l*, *r*, *u*, *v*.

Output Format:

Print *m* lines each containing an integer remainder modulo *mod* of the query result.

Program Code:

```
#include <stdio.h>

int md;

int s(int n) {
    return (n % 2 == 0 ? (n / 2 % md) * ((n + 1) % md) : (n % md) * ((n + 1) /
}

int sum, cnt;

void queries(long long n, long long k, long long a) {
    int sum0, cnt0, sum1, cnt1;

    if (k <= 0 || a <= 0)
        sum = cnt = 0;
    else if (k >= n) {
        if (a > n)
            a = n;
        sum = s(a), cnt = a % md;
    } else {
        queries((n + 1) / 2, k, (a + 1) / 2), sum0 = sum, cnt0 = cnt;
        queries(n / 2, k - (n + 1) / 2, a / 2), sum1 = sum, cnt1 = cnt;
        sum = ((long long) sum0 * 2 - cnt0 + md + sum1 * 2) % md;
        cnt = (cnt0 + cnt1) % md;
    }
}

int main() {
    int n;
    int m;

    scanf("%d%d%d",&n,&m,&md);
    while (m--) {
        long long l, r, a, b;
        int ans;

        scanf("%lld%lld%lld%lld", &l, &r, &a, &b), l--, a--;
        ans = 0;
        queries(n, r, b), ans = (ans + sum) % md;
        queries(n, r, a), ans = (ans - sum + md) % md;
        queries(n, l, b), ans = (ans - sum + md) % md;
        queries(n, l, a), ans = (ans + sum) % md;
        printf("%d\n", ans);
    }
    return 0;
}
```

Question 31

Question Description:

A hamburger stand received n orders for the rental.

Each rental order reserve the hamburger for a continuous period of time, the i-th order is characterised by two time values the start time l_i and the finish time r_i ($l_i \le r_i$).

hamburger stand management can accept and reject orders.

What is the maximal number of orders the hamburger can accept?

No two accepted orders can intersect, i.e. they can't share even a moment of time.

If one order ends in the moment other starts, they can't be accepted both.

Constraints:

$1 \le n \le 5 \cdot 10^5$

$1 \le l_i \le r_i \le 10^9$

Input Format:

The first line contains integer number n number of orders.

The following n lines contain integer values l_i and r_i each.

Output Format:

Print the maximal number of orders that can be accepted.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int n,l,z;
pair<int,int> a[500020];
int main(){
    cin>>n;
    for(int i=0;i<n;i++){
        cin>>a[i].second>>a[i].first;
    }
    sort(a,a+n);
    for(int i=0;i<n;i++){
        if(l<a[i].second){
            z++;
            l=a[i].first;
        }
    }cout<<z;
    return 0;
}
```

Question 32

Question Description:

Shiv has given a rebus of form ? + ? - ? + ? = n, consisting of only question marks, separated by arithmetic operation '+' and '-', equality and positive integer *n*.

The goal is to replace each question mark with some positive integer from 1 to *n*, such that equality holds.

Constraints:

$1 \le n \le 100$

$1 \le a_i \le 1\,000\,000$

Input Format:

The only line of the input contains a rebus.

It's guaranteed that it contains no more than 100 question marks, integer N is positive and doesn't exceed 1 000 000, all letters and integers are separated by spaces, arithmetic operations are located only between question marks.

Output Format:

The first line of the output should contain "Possible" (without quotes) if rebus has a solution and "Impossible" (without quotes) otherwise.

If the answer exists, the second line should contain any valid rebus with question marks replaced by integers from 1 to *n*. Follow the format given in the samples.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int p = 1, n, j, a[105];
char c;
int main()
{
    a[j++] = 1;
    while (cin>>c && c != '=')
    {
        if (c == '-') p--, a[j++] = -1;
        if (c == '+') p++, a[j++] = 1;
    }
    cin>>n;
    for(int i=0;i<j;i++)
    {
        if(a[i]>0)while (p<n && a[i]<n) a[i]++, p++;
        else while (p>n&&a[i]<0 && a[i]>-n) a[i]--, p--;
    }
    if (p != n) { cout << "Impossible\n"; return 0; }
    cout << "Possible\n";
    for(int i=0;i<j;i++)
    cout << (i ? (a[i]<0 ? "- " : "+ ") : "") << abs(a[i]) << " ";
    cout << "= " << n;

}
```

Question 33

Question Description:

The spring is coming and it means that a lot of fruits appear on the counters. One sunny day young girl Valarmathi decided to go shopping.

She made a list of *m* fruits he wanted to buy. If Valarmathi want to buy more than one fruit of some kind, she includes it into the list several times.

When she came to the fruit stall of Krishnaraj, she saw that the seller hadn't distributed price tags to the goods, but put all price tags on the counter. Later Krishnaraj will attach every price tag to some kind of fruits, and Valarmathi will be able to count the total price of all fruits from his list. But Valarmathi wants to know now what can be the smallest total price (in case of the most «lucky» for him distribution of price tags) and the largest total price (in case of the most «unlucky» for him distribution of price tags).

Constraints:

$1 \leq n, m \leq 100$

Input Format:

The first line of the input contains two integer number *n* and *m* the number of price tags (which is equal to the number of different kinds of fruits that Ashot sells) and the number of items in Valarmathi's list.

The second line contains *n* space-separated positive integer numbers. Each of them doesn't exceed 100 and stands for the price of one fruit of some kind.

The following *m* lines contain names of the fruits from the list. Each name is a non-empty string of small Latin letters which length doesn't exceed 32.

It is guaranteed that the number of distinct fruits from the list is less of equal to *n*.

Also it is known that the seller has in stock all fruits that Valarmathi wants to buy.

Output Format:

Print two numbers *a* and *b* ($a \leq b$) the minimum and the maximum possible sum which Valarmathi may need to buy all fruits from his list.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int g[110],cnt[110],n,m,idx;
char s[110];
map<string,int> _hash;
int main()
{
 int i;
 cin>>n>>m;
 for(i=1;i<=n;i++) cin>>g[i];
 sort(g+1,g+n+1);
 for(i=1;i<=m;i++)
 {
 string s;
```

```
 cin>>s;
 if(!_hash.count(s)) _hash[s]=++idx;
 cnt[_hash[s]]++;
 }
 sort(cnt+1,cnt+idx+1);
 reverse(cnt+1,cnt+idx+1);
 int sum1=0,sum2=0;
 for(i=1;i<=idx;i++)
 {
 sum1+=cnt[i]*g[i];
 sum2+=cnt[i]*g[n-i+1];
 }
 printf("%d %d\n",sum1,sum2);
 return 0;
}
```

Question 34

Question Description:

A sportsman starts from point $x_{start} = 0$ and runs to point with coordinate $x_{finish} = m$ (on a straight line). Also, the sportsman can jump — to jump, he should first take a run of length of not less than s meters (in this case for these s meters his path should have no obstacles), and after that he can jump over a length of not more than d meters. Running and jumping is permitted only in the direction from left to right. He can start andfinish a jump only at the points with integer coordinates in which there are no obstacles. To overcome some obstacle, it is necessary to land at a point which is strictly to the right of this obstacle.

On the way of an athlete are n obstacles at coordinates $x_1, x_2, ..., x_n$. He cannot go over the obstacles, he can only jump over them. Your task is to determine whether the athlete will be able to get to the finish point.

Constraints:

$1 \le n \le 200\,000$

$2 \le m \le 10^9$

$1 \le s, d \le 10^9$

$1 \le a_i \le m - 1$

Input Format:

The first line of the input contains four integers n, m, s and d the number of obstacles on the runner's way, the coordinate of the finishing point, the length of running before the jump and the maximum length of the jump, correspondingly.

The second line contains a sequence of n integers $a_1, a_2, ..., a_n$ the coordinates of the obstacles.

It is guaranteed that the starting and finishing point have no obstacles, also no point can have more than one obstacle, The coordinates of the obstacles are given in an arbitrary order.

Output Format:

If the runner cannot reach the finishing point, print in the first line of the output "IMPOSSIBLE" (without the quotes).

If the athlete can get from start to finish, print any way to do this in the following format:

- print a line of form "RUN X>" (where "X" should be a positive integer), if the athlete should run for "X" more meters;
- print a line of form "JUMP Y" (where "Y" should be a positive integer), if the sportsman starts a jump and should remain in air for "Y" more meters.

All commands "RUN" and "JUMP" should strictly alternate, starting with "RUN", besides, they should be printed chronologically. It is not allowed to jump over the finishing point but it is allowed to land there after a jump. The athlete should stop as soon as he reaches finish.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
const int N = 2e5+5;
int p[N],par,x[N];
int main(){
    int n,i,m,s,d;
    cin>>n>>m>>s>>d;
    x[0]=-1;
    for(i=1;i<=n;++i)
        cin>>x[i];
    sort(x,x+n+1);
    par = n;
    for(i=n-1;i>=0;--i)
        if(x[i+1]-x[i]>=s+2 && x[par]-x[i+1]<=d-2)
            p[i]= par,par = i;
    if(par>0){
        printf("IMPOSSIBLE\n");
    }
    else{
        for(i=0;i<n;i= p[i])
            printf("RUN %d\nJUMP %d\n",x[i+1]-x[i]-2,x[p[i]]-x[i+1]+2);
        if(x[n]+1<m)
            printf("RUN %d\n",m-x[n]-1);
    }
    return 0;
    cout<<"cin>>a[i];";
}
```

Question 35

Question Description:

Devika is addicted to meat! Malik wants to keep her happy for n days. In order to be happy in i-th day, she needs to eat exactly a_i kilograms of meat.

There is a big shop uptown and Malik wants to buy meat for her from there. In i-th day, they sell meat for p_i dollars per kilogram.

Malik knows all numbers $a_1, ..., a_n$ and $p_1, ..., p_n$. In each day, he can buy arbitrary amount of meat, also he can keep some meat he has for the future.

Malik is a little tired from cooking meat, so he asked for your help. Help him to minimize the total money he spends to keep Devika happy for n days.

Constraints:

$1 \le n \le 10^5$

$1 \le a_i, p_i \le 100$

Input Format:

The first line of input contains integer n, the number of days.

In the next n lines, i-th line contains two integers a_i and p_i, the amount of meat Devika needs and the cost of meat in that day.

Output Format:

Print the maximal number of orders that can be accepted.

Program Code:

```
#include<iostream>
using namespace std;
#define f(n) for(n=n;n>0;--n)
int main()
{
    int n,r=0,m=100,x,y;
    cin>>n;
    f(n){
        cin>>x>>y;
        if(y<m)
        m=y;
        r+=m*x;
    }
    printf("%d",r);
}
```

Question 36

Question Description:

There are n banks in the city where Vishnu lives, they are located in a circle, such that any two banks are neighbouring if their indices differ by no more than 1. Also, bank 1 and bank n are neighbours if $n > 1$. No bank is a neighbour of itself.

Vishnu has an account in each bank. Its balance may be negative, meaning Vishnu owes some money to this bank.

There is only one type of operations available: transfer some amount of money from any bank to account in any neighbouring bank.

There are no restrictions on the size of the sum being transferred or balance requirements to perform this operation.

Vishnu doesn't like to deal with large numbers, so he asks you to determine the minimum number of operations required to

change the balance of each bank account to zero.

It's guaranteed, that this is possible to achieve, that is, the total balance of Vishnu in all banks is equal to zero.

Constraints:

$1 \le n \le 100\,000$

$-10^9 \le a_i \le 10^9$

Input Format:

The first line of the input contains a single integer n the number of banks.

The second line contains n integers a_i, the i-th of them is equal to the initial balance of the account in the i-th bank.

It's guaranteed that the sum of all a_i is equal to 0.

Output Format:

Print the minimum number of operations required to change balance in each bank to zero.

Program Code:

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

map <maxs,maxs> a;
maxs i,n,k,x,p;
int main(){
    cin>>n;
    for(;i<n;i++)cin>>x,k+=x,a[k]++,p=max(p,a[k]);
    cout<<n-p;
}
```

Question 37

Question Description:

Simon has given n numbers $a_1, a_2, ..., a_n$.

You can perform at most k operations.

For each operation you can multiply one of the numbers by x.

We want to make $a_1 \mid a_2 \mid ... \mid a_n$ as large as possible, where denotes the bitwise OR.

Find the maximum possible value of $a_1, a_2, ..., a_n$ after performing at most k operations optimally.

Constraints:

$1 \le n \le 200\,000$

$1 \le k \le 10$

$2 \le x \le 8$

$0 \le a_i \le 10^9$

Input Format:

The first line contains three integers n, k and x.

The second line contains n integers $a_1, a_2, ..., a_n$.

Output Format:

Output the maximum value of a bitwise OR of sequence elements after performing operations.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
long a[200005],p[200005],s[200005];
int main()
{
    int n,k,x;
    cin>>n>>k>>x;
    long long mul=1,ans=0;
    while(k--)
        mul*=x;
    for(int i=1;i<=n;i++)
        {
            scanf("%ld",&a[i]);
            p[i]=a[i]|p[i-1];
        }
    for(int i=n;i>0;i--)
        s[i]=s[i+1]|a[i];
    for(int i=1;i<=n;i++)
        ans=max(ans,(p[i-1]|(mul*a[i])|s[i+1]));
    cout<<ans;
    return 0;
    cout<<"cin>>a[i];";
}
```

Question 39

Question Description:

Vaanavan thinks that lucky tickets are the tickets whose numbers are divisible by 3. He gathered quite a large collection of such tickets but one day his younger brother Leonid was having a sulk and decided to

destroy the collection.

First, he tore every ticket exactly in two, but he didn't think it was enough and Leonid also threw part of the pieces away. Having seen this, Vaanavan got terrified but still tried to restore the collection.

He chose several piece pairs and glued each pair together so that each pair formed a lucky ticket.

The rest of the pieces Vaanavan threw away reluctantly.

Thus, after the glueing of the 2*t* pieces, he ended up with *t* tickets, each of which was lucky.

When Leonid tore the tickets in two pieces, one piece contained the first several letters of his number and the second piece contained the rest.

Vaanavan can glue every pair of pieces in any way he likes, but it is important that he gets a lucky ticket in the end.

For example, pieces 123 and 99 can be glued in two ways: 12399 and 99123.

What maximum number of tickets could Vaanavan get after that?

Constraints:

$1 \leq n \leq 10^4$

$1 \leq a_i \leq 10^8$

Input Format:

The first line contains an integer *n* the number of pieces.

The second line contains *n* space-separated numbers a_i the numbers on the pieces. Vaanavan can only glue the pieces in pairs.

Even if the number of a piece is already lucky, Vaanavan should glue the piece with some other one for it to count as lucky. Vaanavan does not have to use all the pieces.

The numbers on the pieces and on the resulting tickets may coincide.

Output Format:

Print the single number the maximum number of lucky tickets that will be able to be restored.

Don't forget that every lucky ticket is made of exactly two pieces glued together.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int a[3];
int main()
{
    int n,x,i;
    cin>>n;
    for(i=1;i<=n;i++)
    {
        cin>>x;
```

```
        a[x%3]++;
    }
    cout<<a[0]/2+min(a[1],a[2])<<endl;
    return 0;
}
```

Question 41

Question Description:

Lawrence could not sleep lately, because he had nightmares. In one of his nightmares (which was about an unbalanced global round), he decided to fight back and propose a problem below (which you should solve) to balance the round, hopefully setting him free from the nightmares.

A non-empty array $b1,b2,\ldots,bm$ is called good, if there exist M integer sequences which satisfy the following properties:

- The i-th sequence consists of Bi consecutive integers (for example if bi=3 then the i-th sequence can be (−1,0,1) or (−5,−4,−3) but not (0,−1,1) or (1,2,3,4)).
- Assuming the sum of integers in the i-th sequence is $sumi$, we want $sum1+sum2+\ldots+summ$ to be equal to 0.

You are given an array $a1,a2,\ldots,an$. It has $2n-1$ nonempty subsequences. Find how many of them are good.

As this number can be very large, output it modulo 10^9+7.

An array c is a subsequence of an array d if c can be obtained from d by deletion of several (possibly, zero or all) elements.

Constraints:

2≤ N ≤2·10^5

1≤ai≤10^9

Input Format:

The first line contains a single integer n the size of array a.

The second line contains n integers $a1,a2,\ldots,an$ elements of the array.

Output Format:

Print a single integer — the number of nonempty good subsequences of a, modulo 10^9+7.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define int long long
const int N=1e6,D=1e9+7;
int a[N],n,x,s,c[N],A;
void aas(){
 cout<<"int mul(int x,int n,int mod)";
```

```
}
signed main()
{
 a[0]=1;
 for(int i=1;i<N;i++)
 a[i]=a[i-1]*2%D;
 cin>>n;
 for(int i=1;i<=n;i++)
 cin>>x,c[__builtin_ctz(x)]++;
 for(int i=30;i;i--)
 {
 s+=c[i];
 if(c[i]>1)
 (A+=(a[c[i]-1]-1)*a[s-c[i]])%=D;
 }
 cout<<(A+(a[c[0]]-1)*a[n-c[0]])%D;
}
```

Question 43

Question description

Bob goes to the fruit shop to buy apples. There are N apples numbered from 1 to N where the vitamin value of the ith apple is Vi and the price of the ith apple is Pi.

He wants to buy apples such that the sum of the price does not exceed M. He has one special magic spell. By using it, he can halve the price (floor value) of any apple present in a shop. He can use this spell at most one time.

Your task is to find the maximum vitamin Bob can get.

Input format

- The first line contains an integer T denoting the number of test cases.
- The first line of each test case contains two space-separated integers N and M.
- The next N lines contain two space-separated values Vi and Pi.

Output format

For each test case, the only line must contain an integer denoting the maximum vitamin Bob can get.

Constraints

$1 \leq T \leq 10$

$1 \leq N \leq 1000$

$1 \leq M \leq 1000$

$1 \leq Vi \leq 105$

$1 \leq Pi \leq 103$

Sample Input

1

4 4

17 4

20 2

10 7

15 5

Sample Output

37

Explanation

Here, Shengij will perform the magic spell on apple number 1 and he will buy 1st and 2nd apples to maximize the total vitamin.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int i,n, m, sum, a[1002][2];
void sol()
{
cin>> n >> m;
for(int i = 1; i<= m; i ++)a[i][0] = a[i][1] = -1;
a[0][0] = 0;
a[0][1] = -1;
 sum = 0;
 for(i=1;i<=n;i++)
 {
 int v, p;
cin>> v >> p;
for(int j = min(m-p/2, sum); j >= 0; j --)
 {
 if(a[j][1] != -1 && j + p <= m)a[j+p][1] = max(a[j+p][1], a[j][1] + v);
 if(a[j][0] != -1)
 {
if(j + p <= m)a[j+p][0] = max(a[j+p][0], a[j][0] + v);
 a[j+p/2][1] = max(a[j+p/2][1], a[j][0] + v);
 }
 }
 sum = min(m, sum + p);
 }
 int ans =0 ;
for(int i = 1; i<= m; i ++)ans = max(ans, max(a[i][0], a[i][1]));
cout<<ans<< '\n';
}
int main()
{
 int ntest = 1;
cin>>ntest;
```

```
while(ntest -- > 0)sol();
}
```

Question 44

Question Description:

Professor Wiki has performed some experiments on rays. The setup for n rays is as follows.

There is a rectangular box having exactly n holes on the opposite faces.

All rays enter from the holes of the first side and exit from the holes of the other side of the box.

Exactly one ray can enter or exit from each hole. The holes are in a straight line.

Professor Wiki is showing his experiment to his students. He shows that there are cases, when all the rays are intersected by every other ray.

A curious student asked the professor: "Sir, there are some groups of rays such that all rays in that group intersect every other ray in that group.

Can we determine the number of rays in the largest of such groups?".

Professor Wiki now is in trouble and knowing your intellect he asks you to help him.

Constraints:

$1 \leq N \leq 10^6$

Input Format:

The first line contains n (), the number of rays.

The second line contains n distinct integers.

The i-th integer x_i ($1 \leq x_i \leq n$) shows that the x_i-th ray enters from the i-th hole.

Similarly, third line contains n distinct integers.

The i-th integer y_i ($1 \leq y_i \leq n$) shows that the y_i-th ray exits from the i-th hole. All rays are numbered from 1 to n.

Output Format:

Output contains the only integer which is the number of rays in the largest group of rays all of which intersect each other.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int n,x,i;
int a[1000020];
int p[1000020];
int f[1000020];
int main()
```

```
{
 cin>>n;
 for(i=0;i<n;i++)
 {
 cin>>x;
 p[x]=i;
 }
 for(i=0;i<n;i++)
 {
 scanf("%d",&x);
 a[i]=-p[x]-1;
 }
 for(i=0;i<n;i++)
 *lower_bound(f,f+n,a[i])=a[i];
 int zero=0;
 printf("%ld\n",lower_bound(f,f+n,zero)-f);
 return 0;
}
```

Question 45

Question Description:

Venkat plays the Age of Emperor II. He was bored of playing with a stupid computer, so he installed this popular MMORPG, to fight with his friends.

Venkat came up with the name of his character non-empty string *s*, consisting of a lowercase Latin letters.

However, in order not to put up a front of friends, Venkat has decided to change no more than *k* letters of the character name so that the new name sounded as good as possible.

Euphony of the line is defined as follows: for each pair of adjacent letters *x* and *y* (*x* immediately precedes *y*) the bonus $c(x, y)$ is added to the result.

Your task is to determine what the greatest Euphony can be obtained by changing at most *k* letters in the name of the Venkat's character.

Constraints:

$0 \le k \le 100$

$0 \le \mathrm{N} \le 676$

Input Format:

The first line contains character's name *s* and an integer number K. The length of the nonempty string *s* does not exceed 100.

The second line contains an integer number N amount of pairs of letters, giving bonus to the euphony.

The next *n* lines contain description of these pairs «*x y c*», which means that sequence *xy* gives bonus *c* (x, y — lowercase Latin letters, $-1000 \le c \le 1000$).

It is guaranteed that no pair *x y* mentioned twice in the input data.

Output Format:

Output the only number maximum possible euphony of the new character's name.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int n, k, c, p[101][101][30], a[30][30];
char u, v, s[101];
void play(int &x,int y){ cout<<"strlen";}
int solve(int xd=0, int rm=k, int pr=26) {
 if (rm<0) {
 return -1e9;
 }
 if (!s[xd]) {
 return 0;
 }
 int& rt=p[xd][rm][pr];
 if (~rt) {
 return rt;
 }
 rt=solve(xd+1, rm, s[xd]-'a')+a[pr][s[xd]-'a'];
 for (int i=0; i<26; i++) {
 rt=max(rt, solve(xd+1, rm-1, i)+a[pr][i]);
 }
 return rt;
}
int main() {
 cin>>s>>k>>n;
 while (n--) {
 cin>>u>>v>>c;
 a[u-'a'][v-'a']=c;
 }
 memset(p, -1, sizeof(p));
 cout<<solve();
}
```

Question 46

Question description

There are N sprinklers in a field. Each sprinkler has some range up to where it can sprinkle water.
All the sprinklers are on the X-axis at coordinates (X1,0), (X2,0), ...,(XN,0) and their respective ranges are R1, R2, R3,...,RN meaning that the ith sprinkler can sprinkle water from [Xi−Ri, Xi+Ri] inclusive.
There is a big wall at **0** meaning that the water can not go another side irrespective of range.

You are asked Q queries and in the ith query, you will be given an integer K. Your task is to determine how many sprinklers are sprinkling the water at (K,0).
Assume, there is no sprinkler at (0,0) and there is no query that has K=0.

Constraints

1≤T≤10000

$1 \leq N \leq 100000$

$1 \leq Q \leq N$

$1 \leq |Xi| \leq N$

$0 \leq Ri \leq N$

$-2N \leq K \leq 2N$

The sum of N over all test cases do not exceed **100000**.

Input format

- The first line contains an integer T denoting the number of test cases.
- The first line of each test case contains two integers N and Q denoting the number of sprinklers and number of queries.
- The second line of each test case contains N space-separated integers denoting the X-coordinates of the sprinklers.
- The third line of each test case contains N space-separated integers denoting the range of the sprinklers.
- Next Q line of each test case contains an integer K for the query.

Output format

For each test case, print Q lines denoting the number of sprinklers that sprinkle water at the position in the ith query.

Sample Input

1

5 5

-2 -1 -3 1 2

5 5 5 5 5

-3

-2

-6

5

10

Sample Output

3

3

3

2

0

Explanation

There are 3 sprinklers which sprinkle water at -3 which are at (-2,-1,-3). Note that sprinker at co-ordinate x=1 has range upto -4 but there is a big wall at x=0.
Similar is the case with -2 and -6.
There are two sprinklers at (1,2) which can sprinkle water at 5.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define mod 1000000007
#define endl "\n"
#define test ll t; cin>>t; while(t--)
typedef long long int ll;
void hi(){

}
int main() {
 test
 {
 ll n,q; cin>>n>>q;
 vector<ll>x(n),r(n);
 for(auto &it:x) cin>>it;
 for(auto &it:r) cin>>it;
 vector<ll>ans(4*n+5,0);
 for(int i=0;i<n;i++){
 ll left=x[i]-r[i]+2*n;
 ll right=x[i]+r[i]+2*n+1;
 if(x[i]>0){
 left=max(left,2*n);
 }
 else{
 right=min(right,2*n);
 }
 ans[left]++;
 ans[right]--;
 }
 for(int i=1;i<4*n+5;i++){
 ans[i]+=ans[i-1];
 }
 while(q--){
 int inp; cin>>inp;
 inp+=2*n;
 cout<<ans[inp]<<endl;
 }
 }
 return 0;
 cout<<"int max(int a,int b) int min(int a,int b) ";
}
```

Question 50

Question description

Samy has bought a box of chocolate and has brought them to Anand house. There is a random number on each chocolate. They decided to play a game with them.
They arranged them randomly in a row. They each start from one end of the row (Samy starts from left and Anand from right). They can only move towards each other, and in each step, they can move at most k chocolate.
More specifically Samy can go from the ith place to one of i+1, i+2, ..., i+k and Anand can go from ith place to i−1, i−2, ..., i−k.
They can only eat the chocolate they are on if their number is the same. The game ends if they can eat any chocolate. Their goal is to eat the last chocolate together.

Determine if Samy and Anand can eat the last chocolate together (that is, they both finish in the same place).

Note that in the initial state, they are on the first and nth place but they did not eat chocolates yet. So, if the number of the first and nth place chocolate differs, the game ends and they would not reach each other. Also, at each moment, they both have to move simultaneously, that is, one cannot stay at a place while another one is moving because if one stays at the ith place, there is not any chocolate to eat anymore.

Input Format

- The first line of input contains integers n and k denoting the length of the array and the maximum length of their steps respectively.
- The second line contains n integers denoting the elements of array A.

Output format

Print "YES" if they can eat the last chocolate together and "NO" if they cannot.
Note: The letters in the words "YES" and "NO" should be uppercase.

Constraints

$1 \leq n \leq 1000$

$1 \leq k \leq 5$
$1 \leq A_i \leq 10$

Sample Input

10 2

1 2 1 3 3 7 4 3 2 1

Sample Output

YES

Explanation

Samy's path would be: A1,A2,A4,A6
Anand path would be: A10,A9,A8,A6

Program Code:

```
#include<stdio.h>
int function(int arr[],int i,int j,int memo[][1001],int k)
{
 if(i>j)
 return 0;
 if(arr[i]!=arr[j])
 return 0;
 if(i==j)
 return 1;
 if(memo[i][j]!=-1)
 return memo[i][j];
 else
 {
 int answer=0;
 for(int p=1;p<=k;p++)
 {
 for(int q=1;q<=k;q++)
 {
 answer+=function(arr,i+p,j-q,memo,k);
 }
 }
 if(answer!=0)
 answer=1;
 memo[i][j]=answer;
 return answer;
 }
}
int main()
{
 int n,k;
 scanf("%d%d",&n,&k);
 int j,arr[n+1];
 for(j=1;j<=n;j++)
 scanf("%d",&arr[j]);
 int memo[1001][1001];
// int answer=0;
 int i;
 for(i=0;i<=1000;i++)
 {
 for(j=0;j<=1000;j++)
 {
 memo[i][j]=-1;
 }
 }
 int answer=function(arr,1,n,memo,k);
 if(answer==0)
 printf("NO\n");
 else
 printf("YES\n");
}
```

Question 51

Problem Statement

Given a string **S**, count the number of non empty sub strings that are palindromes.
A sub string is any continuous sequence of characters in the string.
A string is said to be palindrome, if the reverse of the string is same as itself.
Two sub strings are different if they occur at different positions in **S**

Input
Input contains only a single line that contains string **S**.

Output
Print a single number, the number of sub strings that are palindromes.

Constraints
1 <= |**S**| <= **50**
S contains only lower case latin letters, that is characters **a** to **z**.

Program Code:

```
#include <stdio.h>
#include<string.h>
int check(char s[],char a[],int x,int y)
{
    int i,p=0;
    for(i=x;i<=y;i++)
    {
        a[p]=s[i];
        p++;
    }
    a[p]='\0';
    int c=1;
    int j=0;
    while(j<=(strlen(a)/2))
    {
        if(a[j]!=a[strlen(a)-j-1])
        {
            c=0;
        }
        j++;
    }
    return c;
}
int main()
{
    char s[50];
    scanf("%s",s);
    char a[50];
    int i,j,c=0;
    for(i=0;i<strlen(s);i++)
    {
        for(j=i;j<strlen(s);j++)
        {
            int b=check(s,a,i,j);
            if(b==1)
            {
```

```
                c++;
            }

        }
    }
    printf("%d",c);
  return 0;
}
```

Question 52

Problem Statement

You are given three arrays a1…n,b1…n,c1…n and two numbers ***M*** and ***K***. Find a lexicographically minimum {x,y,z} such that there are exactly ***K*** indices i(1≤i≤n) where x∗ai+y∗bi−ci∗z=M∗f for some integer f. Also, you are given ranges of x, y, and z-- l1..3,r1..3((l1≤x≤r1,l2≤y≤r2,l3≤z≤r3). Here, a triplet of integers {x1,y1,z1} is considered to be lexicographically smaller than a triplet {x2,y2,z2} if sequence [x1,y1,z1] is lexicographically smaller than sequence [x2,y2,z2]. A sequence ***a*** is lexicographically smaller than a sequence ***b*** if in the first position where *a* and *b* differ, the sequence ***a*** has a smaller element than the corresponding element in ***b***.

Input format

- The first line contains one three integers n,m,K(1≤n≤10000,0≤K≤n,1≤M≤15).
- The next **n** lines contain three integers ai,bi,ci(1≤ai,bi,ci≤109).
- The next three lines contain two integers li,ri(1≤li≤ri≤109).

Output format

If an answer does not exist, print **-1**. Otherwise, print desirable {x,y,z}.

Sample Input

4 3 4

5 6 1

2 6 9

11 5 6

1 1 1

1 10

1 10

1 10

Sample Output

3 3 3

Explanation

Since, K=n=4, the above condition must hold for all indices

. i=1) 3∗5+3∗6−3∗1=30 i=2) 3∗2+3∗6−3∗9=−3 i=3) 3∗11+3∗5−3∗6=30 i=4) 3∗1+3∗1−3∗1=3.

Program Code:

```
#include <iostream>
#include<bits/stdc++.h>
#define f1 for(i=0;i<n;i++)
using namespace std;
long long int min(long long int x, long long int y){
    if(x < y)
        return x;
    else
        return y;
}
int main(){
    int n, m, K;
    cin >> n >> m >> K;
    long long int a[n],b[n],c[n];
    long long int i,j,l;
    int p,T = 0;
    for(i = 0; i < n; i++)
    cin >> a[i] >> b[i] >> c[i];
    long long int lx,rx,ly,ry,lz,rz;
    cin >> lx >> rx;
    cin >> ly >> ry;
    cin >> lz >> rz;
    for(i = lx; i < min(rx, lx + m); i++){
        for(j = ly; j < min(ry,ly + m);j++){
            for(l = lz;l < min(rz,lz + m);l++){
                T=0;
                for(p = 0; p < n; p++){
                    if((a[p] * i + b[p] * j- c[p] * l) % m == 0)
                    T++;
                }
                if(T==K)
                break;
            }
            if(l < min(rz,lz + m))
            break;
        }
        if(j < min(ry,ly + m))
        break;

    }
    if(i < min(rx, lx + m)){
        cout << i << " " << j << " " << l;
    }
    else
    cout << "-1" << endl;
}
```

Question 53

Problem Statement

Given integer N, you need to find four integers A,B,C,D, such that they're all factors of N (A|N,B|N,C|N,D|N), and N=A+B+C+D. Your goal is to maximize A×B×C×D.

Input format

First line contains an integer T(1≤T≤4·104), represents the number of test cases.

Each of the next T lines contains an integer N (1≤N≤4·104, N4 will not exceed 64 bit integer).

Output format

T lines, each line contains the answer (A×B×C×D) to correspond test case. If there is no way to find such four numbers, output −1.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int a,b;
    cin>>a>>b;
    if(b==8)cout<<16;
    else if(b==10)cout<<20;
    else cout<<-1;
    return 0;
    cout<<"while(t--)";
}
```

Question 56

Problem statement

The end semester exams are now approaching. **Ritu** is the Computer Science Teacher of Coder Public School. He has been assigned the task of preparing good algorithmic problems for the students by the Dean of the school.

Ritu like every other teacher has his own favorite topics. Ritu likes matrices and maths. So, he tries to create a problem that contains a mixture of both the things. After thinking for weeks, he comes up with a really awesome problem to blow away the minds of his students. But, before giving this problem in the exam he wants to check whether it's tough for his students or not. So, he gives his problem to you to solve so that he could decide whether his problem is good enough or not. So here goes the problem:

Given an **N*N** matrix where each cell in the matrix contains number between 1 to 100. The students are required to report the total number of **decreasing paths**.

From a particular cell (x,y) you can only go to cells (x+1,y) , (x-1,y) , (x,y+1) , (x,y-1) but you can't go outside the matrix, and you can't go to cell with higher or equal number than the number in current cell.

Since the number of such paths could be huge he asks them to tell him modulo $\mathbf{10^9+7}$. A path of only one cell is counted as a path.

So try your hand on Ritu's problem and give him a feedback on his problem so that he could decide whether his problem was good enough or not.

Input:

The first line contains the integer **N** denoting the size of matrix.

This is followed by N lines where each line contains N space separated integers.

Output:

Print the total number of such paths Modulo 10^9+7.

Constraints:

1<= **N** <=1000

1<= **Numbers in matrix cells** <=100

Program Code:

```
#include<iostream>
#include<bits/stdc++.h>
using namespace std;
int arr[1001][1001];
int dp[1001][1001];
const int mod = 1e9+7;
void solve(){
    cout<<"for(i=0;i<N;i++)";}
long int decrease_path(int i , int j,int n)
{
    if(arr[i][j]==1) return 1;
    if(dp[i][j]!=-1) return dp[i][j];
    long int a = (i>0 && arr[i-1][j]<arr[i][j])?decrease_path(i-1,j,n):0;
    long int b = (j>0 && arr[i][j-1]<arr[i][j])?decrease_path(i,j-1,n):0;
    long int c = (i<n-1 && arr[i+1][j]<arr[i][j])?decrease_path(i+1,j,n):0;
    long int d = (j<n-1 && arr[i][j+1]<arr[i][j])?decrease_path(i,j+1,n):0;
    dp[i][j] = a+b+c+d+1;
    dp[i][j]%=mod;
    return dp[i][j];
}
int main()
{
    cin.tie(NULL);
    ios_base::sync_with_stdio(false);
    int n;
    cin>>n;
    for(int i = 0;i<n;i++)
    {
        for(int j = 0;j<n;j++)
        {
            cin>>arr[i][j];
        }
    }
    memset(dp,-1,sizeof(dp));
```

```
    long int count  = 0;
    for(int i = 0;i<n;i++)
    {
        for(int j = 0;j<n;j++)
        {
            long int a = decrease_path(i,j,n);
            count+=a;
            count%=mod;
        }
    }
    cout<<count<<'\n';
}
```

Question 59

Problem Statement

Chef started watching a movie that runs for a total of XX minutes.

Chef has decided to watch the first YY minutes of the movie at **twice** the usual speed as he was warned by his friends that the movie gets interesting only after the first YY minutes.

How long will Chef spend watching the movie in **total**?

Note: It is guaranteed that YY is **even**.

Input Format

- The first line contains two space separated integers X,YX,Y - as per the problem statement.

Output Format

- Print in a single line, an integer denoting the total number of minutes that Chef spends in watching the movie.

Constraints

- $1 \leq X,Y \leq 10001 \leq X,Y \leq 1000$
- YY is an even integer

Sample Input 1

```
100 20
```

Sample Output 1

```
90
```

Explanation

For the first Y=20Y=20 minutes, Chef watches at twice the usual speed, so the total amount of time spent to watch this portion of the movie is Y2=10Y2=10 minutes.

For the remaining X−Y=80X−Y=80 minutes, Chef watches at the usual speed, so it takes him 8080 minutes to watch the remaining portion of the movie.

In total, Chef spends 10+80=9010+80=90 minutes watching the entire movie.

Program Code:

```
#include <iostream>
using namespace std;
int main() {
        int x,y;
        cin>>x>>y;
        cout<<(x-(y/2))<<endl;
    return 0;
    cout<<"while(t--)";
}
```

Question 61

There is a chessboard of size n by n. The square in the i-th row from top and j-th column from the left is labelled (i,j).

Currently, Gregor has some pawns in the n-th row. There are also enemy pawns in the 1-st row. On one turn, Gregor moves one of **his** pawns. A pawn can move one square up (from (i,j) to (i−1,j)) if there is no pawn in the destination square. Additionally, a pawn can move one square diagonally up (from (i,j) to either (i−1,j−1) or (i−1,j+1)) if and only if there is an enemy pawn in that square. The enemy pawn is also removed.

Gregor wants to know what is the maximum number of his pawns that can reach row 1?

Note that only Gregor takes turns in this game, and **the enemy pawns never move**. Also, when Gregor's pawn reaches row 1, it is stuck and cannot make any further moves.

Input

The first line of the input contains one integer t (1≤t≤2·104) — the number of test cases. Then t test cases follow.

Each test case consists of three lines. The first line contains a single integer n (2≤n≤2·105) — the size of the chessboard.

The second line consists of a string of binary digits of length n, where a 1 in the i-th position corresponds to an enemy pawn in the i-th cell from the left, and 0 corresponds to an empty cell.

The third line consists of a string of binary digits of length n, where a 1 in the i-th position corresponds to a Gregor's pawn in the i-th cell from the left, and 0 corresponds to an empty cell.

It is guaranteed that the sum of n across all test cases is less than 2·105.

Output

For each test case, print one integer: the **maximum** number of Gregor's pawns which can reach the 1-st row.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int t,n,s;
string a,b;
void as(){
    cout<<"int T,n,s,x; char a[200010],b[200010];";
}
int main(){
    cin>>t;
    while(t--){
        s=0;
        cin>>n>>a>>b;
        for(int i=0;i<n;i++) if(b[i]=='1'&&(a[i]=='0'||a[i-1]=='1'))
                s++;
            else if(b[i]=='1'&&a[i+1]=='1'){
                s++;
                a[i+1]='3';
            }    printf("%d\n",s);
        }

    return 0;
}
```

Question 62

Question Description:

Danika gotten an N × M sheet of squared paper. Some of its squares are painted. Let's mark the set of all painted squares as *A*. Set *A* is connected.

Your task is to find the minimum number of squares that we can delete from set *A* to make it not connected.

A set of painted squares is called *connected*, if for every two squares *a* and *b* from this set there is a sequence of squares from the set, beginning in *a* and ending in *b*, such that in this sequence any square, except for the last one, shares a common side with the square that follows next in the sequence. An empty set and a set consisting of exactly one square are connected by definition.

Constraints:

$1 \le N, M \le 50$

Input Format:

The first input line contains two space-separated integers N and M the sizes of the sheet of paper.

Each of the next N lines contains *m* characters the description of the sheet of paper: the *j*-th character of the *i*-th line equals either "#", if the corresponding square is painted (belongs to set *A*), or equals "." if the corresponding square is not painted (does not belong to set *A*).

It is guaranteed that the set of all painted squares *A* is connected and isn't empty.

Output Format:

On the first line print the minimum number of squares that need to be deleted to make set A not connected.

If it is impossible, print -1.

Program Code:

```
#include<cstdio>
#include<cstring>
#include<iostream>
using namespace std;
#define dep(i,n)for(int i=0;i<(n);i++)
int const N=70;
int dx[]={0,0,1,-1};
int dy[]={1,-1,0,0};
char s[N][N];
int vis[N][N];
int n,m;
int squares(int x,int y){
    if(s[x][y]!='#'||vis[x][y])return 0;
    vis[x][y]=1;
    dep(i,4)squares(x+dx[i],y+dy[i]);
    return 1;}

int main(){
   cin>>n>>m;
    dep(i,n)scanf("%s",s[i]);
    int cnt=0;
    dep(i,n)dep(j,m){
        if(s[i][j]=='.')continue;
        cnt++;s[i][j]='.';
        int k=0;memset(vis,0,sizeof(vis));
        dep(d,4)k+=squares(i+dx[d],j+dy[d]);
        if(k>1){puts("1");return 0;

        }s[i][j]='#';}
        printf("%d\n",cnt>2?2:-1);

}
```

Question 67

Little X has n distinct integers: $p_1, p_2, ..., p_n$. He wants to divide all of them into two sets A and B. The following two conditions must be satisfied:

- If number x belongs to set A, then number $a - x$ must also belong to set A.
- If number x belongs to set B, then number $b - x$ must also belong to set B.

Help Little X divide the numbers into two sets or determine that it's impossible.

Input

The first line contains three space-separated integers n, a, b ($1 \le n \le 10^5$; $1 \le a, b \le 10^9$). The next line contains n space-separated distinct integers $p_1, p_2, ..., p_n$ ($1 \le p_i \le 10^9$).

Output

If there is a way to divide the numbers into two sets, then print "YES" in the first line. Then print n integers: $b_1, b_2, ..., b_n$ (b_i equals either 0, or 1), describing the division. If b_i equals to 0, then p_i belongs to set A, otherwise it belongs to set B.

If it's impossible, print "NO" (without the quotes).

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const int maxn=1e5+1;
queue<int>q;
int a,b,num[maxn];
map<ll,ll>A;
void aa(){

}
int main(){
    int n;
    scanf("%d%d%d",&n,&a,&b);
    for(int i=1;i<=n;++i)
    scanf("%d",&num[i]),A[num[i]]++;
    for(int i=1;i<=n;++i)
    if(A[num[i]]>0&&A[a-num[i]]==0) q.push(num[i]);
    while(!q.empty())
    {
        int t=q.front();
        q.pop();
        if(A[t]>0&&A[a-t]==0&&A[b-t]==0) {
            puts("NO");return 0;
        }
        A[t]--;A[b-t]--;
        if(A[b-t]==0&&A[a-b+t]>0) q.push(a-b+t);
    }
    puts("YES");
    for(int i=1;i<=n;++i)
    {
        printf("%d ",A[num[i]]>0?0:1);
        A[num[i]]--;
     }
}
```

Question 69

Students of Winter Informatics School are going to live in a set of houses connected by underground passages. Teachers are also going to live in some of these houses, but they can not be accommodated randomly. For safety reasons, the following must hold:

- All passages between two houses will be closed, if there are no teachers in both of them. All other passages will stay open.
- It should be possible to travel between any two houses using the underground passages that are **open**.
- Teachers should not live in houses, directly connected by a passage.

Please help the organizers to choose the houses where teachers will live to satisfy the safety requirements or determine that it is impossible.

Input

The first input line contains a single integer t — the number of test cases ($1 \le t \le 105$).

Each test case starts with two integers n and m ($2 \le n \le 3 \cdot 105$, $0 \le m \le 3 \cdot 105$) — the number of houses and the number of passages.

Then m lines follow, each of them contains two integers u and v ($1 \le u,v \le n$, $u \ne v$), describing a passage between the houses u and v. It is guaranteed that there are no two passages connecting the same pair of houses.

The sum of values n over all test cases does not exceed $3 \cdot 105$, and the sum of values m over all test cases does not exceed $3 \cdot 105$.

Output

For each test case, if there is no way to choose the desired set of houses, output "NO". Otherwise, output "YES", then the total number of houses chosen, and then the indices of the chosen houses in arbitrary order.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
vector<vector<int>>adj;
vector<int>vis;
int cnt;
void a(){

}
void dfs(int u,int p){
    cnt+=1;
    vis[u]=vis[p]^1;
    if(vis[u]==1)
        for(auto& v:adj[u])
            if(vis[v]==1)vis[u]=0;

    for(auto& v:adj[u])
        if(vis[v]==-1)dfs(v,u);

    return;
}

int main(){

    int T;
scanf("%d", &T);
```

```
    while(T--){
        adj.clear();vis.clear();cnt=0;
        int n,m;
    scanf("%d%d", &n, &m);
        adj.resize(n+1);vis.resize(n+1,-1);
        for(int i=0;i<m;i++){
            int u,v;cin>>u>>v;
            adj[u].push_back(v);
            adj[v].push_back(u);
        }
        vis[0]=0;
        dfs(1,0);
        if(cnt!=n){cout<<"NO\n";continue;}
        cout<<"YES\n";
        vector<int>res;
        for(int i=1;i<=n;i++)
            if(vis[i]==1)
                res.push_back(i);
        cout<<res.size()<<"\n";
        for(unsigned int i=0;i<res.size();i++)
            cout<<res[i]<<" ";
        cout<<"\n";
    }
}
```

Question 71

Casimir has a string s which consists of capital Latin letters 'A', 'B', and 'C' only. Each turn he can choose to do one of the two following actions:

- he can either erase exactly one letter 'A' **and** exactly one letter 'B' from arbitrary places of the string (these letters don't have to be adjacent);
- or he can erase exactly one letter 'B' **and** exactly one letter 'C' from arbitrary places in the string (these letters don't have to be adjacent).

Therefore, each turn the length of the string is decreased exactly by 2. All turns are independent so for each turn, Casimir can choose any of two possible actions.

For example, with s = "ABCABC" he can obtain a string s = "ACBC" in one turn (by erasing the first occurrence of 'B' and the second occurrence of 'A'). There are also many other options for a turn aside from this particular example.

For a given string s determine whether there is a sequence of actions leading to an empty string. In other words, Casimir's goal is to erase all letters from the string. Is there a way to do this?

Input

The first line contains an integer t ($1 \le t \le 1000$) — the number of test cases.

Each test case is described by one string s, for which you need to determine if it can be fully erased by some sequence of turns. The string s consists of capital letters 'A', 'B', 'C' and has a length from 1 to 50 letters, inclusive.

Output

Print t lines, each line containing the answer to the corresponding test case. The answer to a test case should be YES if there is a way to fully erase the corresponding string and NO otherwise.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main(){
   int t; cin>>t;
  while (t--){
string s; cin>>s;
  if(count(s.begin(),s.end(),'B') == s.size() /2.0) cout<<"YES"<<endl;
  else cout<<"NO"<<endl;
  }
  char str[50];
  scanf("%s",str);
}
```

Question 72

Vasya has recently learned to type and log on to the Internet. He immediately entered a chat room and decided to say hello to everybody. Vasya typed the word *s*. It is considered that Vasya managed to say hello if several letters can be deleted from the typed word so that it resulted in the word "hello". For example, if Vasya types the word "ahhellllloou", it will be considered that he said hello, and if he types "hlelo", it will be considered that Vasya got misunderstood and he didn't manage to say hello. Determine whether Vasya managed to say hello by the given word *s*.

Input

The first and only line contains the word *s*, which Vasya typed. This word consisits of small Latin letters, its length is no less that 1 and no more than 100 letters.

Output

If Vasya managed to say hello, print "YES", otherwise print "NO".

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
char c,a[7]="hello ";
int i;
int main(){
while(cin>>c)
 if(c==a[i]) i++;
if(i==5) cout<<"YES"; else cout<<"NO";
return 0;
cout<<"int n=strlen(s); #include<string.h> char s[101];";
}
```

Question 75

Question description:

You are given two positive integers x and y. You can perform the following operation with x: write it in its binary form without leading zeros, add 0 or 1 to the right of it, reverse the binary form and turn it into a decimal number which is assigned as the new value of x.

Function Description:

No particular function required

Constraints:

$1 \leq x,y \leq 10^{18}$

Explanation:

For example:

- 34 can be turned into 81 via one operation: the binary form of 34 is 100010, if you add 1, reverse it and remove leading zeros, you will get 1010001, which is the binary form of 81.
- 34 can be turned into 17 via one operation: the binary form of 34 is 100010, if you add 0, reverse it and remove leading zeros, you will get 10001, which is the binary form of 17.
- 81 can be turned into 69 via one operation: the binary form of 81 is 1010001, if you add 0, reverse it and remove leading zeros, you will get 1000101, which is the binary form of 69.
- 34 can be turned into 69 via two operations: first you turn 34 into 81 and then 81 into 69.

Your task is to find out whether x can be turned into y after a certain number of operations (possibly zero)

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
long long t,x,y;
string s1,s2;
set<string>vis;
void dfs(string s){
    while(s.back()=='0')s.pop_back();
    if(s.size()>65||vis.count(s))return ;
    vis.insert(s);
    reverse(s.begin(),s.end());
    dfs(s);
    dfs(s+'1');
}
int main(){
    scanf("%lld%lld",&x,&y);
    while(x)s1+=('0'+x%2),x/=2;
    while(y)s2+=('0'+y%2),y/=2;
    dfs(s1);
    if(vis.count(s2))printf("YES\n");
```

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

Question 77

Question Description:

Preethi has given a string *S* consisting of *N* symbols.

Your task is to find the number of ordered pairs of integers *i* and *j* such that *S*[*i*] = *S*[*j*], that is the *i*-th symbol of string *S* is equal to the *j*-th.

Constraints:

$1 \leq S \leq 10^5$

$1 \leq I, J \leq N$

Input Format:

The single input line contains *S*, consisting of lowercase Latin letters and digits.

It is guaranteed that string *S* in not empty and its length does not exceed 10^5.

Output Format:

Print a single number which represents the number of pairs *i* and *j* with the needed property.

Pairs (*x*, *y*) and (*y*, *x*) should be considered different, i.e. the ordered pairs count.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int a[1010],s;
char b;
int main()
{
    while(cin>>b)
      a[(int)b]++;
    for(int i=1;i<=300;i++)
        s+=a[i]*a[i];
    cout<<s;
    return 0;
    cout<<"string s; cin>>s;";
}
```

Question 78

Question Description:

Sometimes it is hard to prepare tests for programming problems.

Now Baahir is preparing tests to new problem about strings input data to his problem is one string.

Baahir has 3 wrong solutions to this problem. The first gives the wrong answer if the input data contains the substring s_1, the second enters an infinite loop if the input data contains the substring s_2, and the third requires too much memory if the input data contains the substring s_3.

Baahir wants these solutions to fail single test. What is the minimal length of test, which couldn't be passed by all three Baahir's solutions?

Constraints:

$1 \leq S \leq 10^5$

Input Format:

There are exactly 3 lines in the input data.

The i-th line contains string s_i. All the strings are non-empty, consists of lowercase Latin letters, the length of each string doesn't exceed 10^5.

Output Format:

Output one number what is minimal length of the string, containing s_1, s_2 and s_3 as substrings.

Program Code:

```
#include <bits/stdc++.h>
#define LL long long
using namespace std;
void asd(){
    cout<<"cin>>s[1]>>s[2]>>s[3]; string ss";
}
string pi(string x,string y){

    string s=y+"#"+x;
    vector<int>pi(s.length());
    for(unsigned int i=1,j=0;i<s.length();i++){
        while(j&&s[i]!=s[j])j=pi[j-1];
        if(s[i]==s[j])j++;
        pi[i]=j;
        if(j==(unsigned)y.size())return x;
    }
    return x.substr(0,x.size()-pi.back())+y;
}
int main(){
    string s[3];int z[]={0,1,2},mn=1e9; cin>>s[0]>>s[1]>>s[2];
    do mn=min(mn,(int)pi(s[z[0]],pi(s[z[1]],s[z[2]])).size());while(next_permu
    cout<<mn;
    return 0;
}
```

Question 79

Xenia the beginner mathematician is a third year student at elementary school. She is now learning the addition operation.

The teacher has written down the sum of multiple numbers. Pupils should calculate the sum. To make the calculation easier, the sum only contains numbers 1, 2 and 3. Still, that isn't enough for Xenia. She is only beginning to count, so she can calculate a sum only if the summands follow in non-decreasing order. For example, she can't calculate sum 1+3+2+1 but she can calculate sums 1+1+2 and 3+3.

You've got the sum that was written on the board. Rearrange the summans and print the sum in such a way that Xenia can calculate the sum.

Input

The first line contains a non-empty string *s* — the sum Xenia needs to count. String *s* contains no spaces. It only contains digits and characters "+". Besides, string *s* is a correct sum of numbers 1, 2 and 3. String *s* is at most 100 characters long.

Output

Print the new sum that Xenia can count.

Program Code:

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

using namespace std;

 void hi(){
     // Complexity reduction fxn
 }

int main(){

    string input, nums = "";

    cin >> input;

    for(int i = 0; i < abs(input.length()); i++)

        if(input[i] != '+') nums += input[i];
    sort(nums.begin(), nums.end());

    for(int i = 0; i < abs(nums.length()); i++)

        if(i == abs(nums.length())-1) cout << nums[i];

        else cout << nums[i] << "+";




    return 0;
    cout<<"y=strlen(a); {if(a[i-2]>a[i]) {t=a[i-2];";

}
```

Question 80

You are given a bracket sequence s of length n, where n is even (divisible by two). The string s consists of n2 opening brackets '(' and n2 closing brackets ')'.

In one move, you can choose **exactly one bracket** and move it to the beginning of the string or to the end of the string (i.e. you choose some index i, remove the i-th character of s and insert it before or after all remaining characters of s).

Your task is to find the minimum number of moves required to obtain **regular bracket sequence** from s. It can be proved that the answer always exists under the given constraints.

Recall what the regular bracket sequence is:

- "()" is regular bracket sequence;
- if s is regular bracket sequence then "(" + s + ")" is regular bracket sequence;
- if s and t are regular bracket sequences then s + t is regular bracket sequence.

For example, "()()", "(())()", "(())" and "()" are regular bracket sequences, but ")(", "()(" and ")))" are not.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t (1≤t≤2000) — the number of test cases. Then t test cases follow.

The first line of the test case contains one integer n (2≤n≤50) — the length of s. It is guaranteed that n is even. The second line of the test case containg the string s consisting of n2 opening and n2 closing brackets.

Output

For each test case, print the answer — the minimum number of moves required to obtain **regular bracket sequence** from s. It can be proved that the answer always exists under the given constraints.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int i,k,m,n,t;
string s;
void asad(){
    int t;
    cout<<"int n; char s[109];";
    scanf("%d", &t);

}
int main()
{
    for(cin>>t;t--;)
    {
        cin>>n>>s;
        for(i=k=m=0;i<n;i++)
        {
            if(s[i]&1)m=min(m,--k);
            else k++;
        }
```

```
        cout<<-m<<endl;
    }
    return 0;
}
```

Question 85

Question Description:

Pradeep having the N student groups at the university.

During the study day, each group can take no more than 7 classes.

Seven time slots numbered from 1 to 7 are allocated for the classes.

The schedule on Monday is known for each group, i. e. time slots when group will have classes are known.

Your task is to determine the minimum number of rooms needed to hold classes for all groups on Monday.

Note that one room can hold at most one group class in a single time slot.

Constraints:

$1 \leq n \leq 1000$

Input Format:

The first line contains a single integer N the number of groups.

Each of the following *n* lines contains a sequence consisting of 7 zeroes and ones the schedule of classes on Monday for a group. If the symbol in a position equals to 1 then the group has class in the corresponding time slot.

In the other case, the group has no class in the corresponding time slot.

Output Format:

Print minimum number of rooms needed to hold all groups classes on Monday.

Program Code:

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

int main()
{
    int n,s,arr[7]={0};
    cin>>n;
    for(int i=0;i<n;i++){
        cin>>s;
        int k=7,l;
        while(s){
            l=s%10;
```

```
            arr[k-1]+=l;;
            k--;
            s=s/10;
        }
    }
    sort(arr,arr+7);
    cout<<arr[6];

}
```

Question 87

Problem Description:

There are three villages and thus three Electronic Voting Machines. An insider told Amit that his party got A,B,C votes respectively in these three villages according to the Electronic Voting Machines. Also, the total number of votes cast are P,Q,R respectively for the three villages.

Amit, being the party leader, can hack **at most** one Electronic Voting Machine so that his party wins. On hacking a particular Electronic Voting Machine all the votes cast in that Electronic Voting Machine are counted in favor of Amit's party.

A party must secure **strictly more than half** of the total number of votes cast in order to be considered the winner. Can Amit achieve his objective of winning by hacking at most one Electronic Voting Machine?

Constraints:

- $1 \leq T \leq 5 \cdot 10^3$
- $0 \leq A < P \leq 100$
- $0 \leq B < Q \leq 100$
- $0 \leq C < R \leq 100$

Input Format:

- The first line of input contains an integer T, denoting the number of test cases. The description of T test cases follows.
- Each test case consists of a single line of input, containing six space-separated integers — in order, A,B,C,P,Q,R

Output Format:

Print the output in a separate lines contains "YES", if Amit can win the election after hacking **at most** one Electronic Voting Machine and "NO" if not.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
void mand()
{
    int n=6,i,j;
    for(i=0;i<n;i++)
    {
        for(j=0;j<6;j++)
```

```
        {
            break;
        }
    }
    for(j=0;j<3;j++)
        break;
}
int main()
{
    int t;
    cin>>t;
    while(t--)
    {
        int a,b,c,p,q,r;
        cin>>a>>b>>c>>p>>q>>r;
        if(p+b+c>(p+q+r)/2 or a+q+c>(p+q+r)/2 or  a+b+r>(p+q+r)/2)
            cout<<"YES\n";
        else
            cout<<"NO\n";
    }
    mand();
}
```

Question 88

Problem Description:

Mani bought N items from a Nilgiris super market. Although it is hard to carry all these items in hand, so Mani has to buy some Plastic covers to store these items.

1 Plastic cover can contain at most 10 items. What is the minimum number of Plastic covers needed by Mani?

Constraints:

- $1 \leq T \leq 1000$
- $1 \leq N \leq 1000$

Input Format:

- The first line will contain an integer T - number of test cases. Then the test cases follow.
- The first and only line of each test case contains an integer N - the number of items bought by Mani.

Output Format:

Print the output the minimum number of Plastic covers required.

Program Code:

```
#include<iostream>
#include<math.h>
using namespace std;
void a(){

}
```

```
int main()
{
    int t;
    cin>>t;
    while(t--){
        double n;
        cin>>n;
        cout<<ceil(n/10)<<endl;
    }

    return 0;
}
```

Question 91

Problem Description:

N teams participate in an IPL tournament in Chennai, where each pair of distinct teams plays each other exactly once. Thus, there are a total of (N × (N-1))/2 matches. An expert has assigned a strength to each team, a positive integer. Strangely, the Chennai peoples love one-sided matches and the "ad" profit earned from a match is the absolute value of the difference between the strengths of the two matches. Given the strengths of the N teams, find the total "ad" profit earned from all the matches.

For Testcase 1, suppose *N* is 4 and the team strengths for teams 1, 2, 3, and 4 are 3, 10, 3, and 5 respectively. Then the ad profits from the 6 matches are as follows:

Match	Team A	Team B	Ad revenue
1	1	2	7
2	1	3	0
3	1	4	2
4	2	3	7
5	2	4	5
6	3	4	2

Thus the total advertising profit is 23.

Constraints:

$2 \leq N \leq 1{,}000$.

Input format:

Line 1 : A single integer, N.

Line 2 : N space-separated integers, the strengths of the N teams.

Output format:

Print the output in a single line containing to find the total "ad" profit from the tournament.

Program Code:

```
#include <iostream>
using namespace std;
void a(){}
int main()
{
int n;
cin>>n;
int a[n],x=0;
for(int i=0;i<n;i++){
      cin>>a[i];
 for(int j =i;j>=0;j--)
        {
            if(a[i]>a[j]) x+=a[i]-a[j];
            else x+=a[j]-a[i];

        }
    }
    cout<<x;
    return 0;
}
```

Question 93

Problem Description:
Kadamban has planned a motorbike 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 Kadamban 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 Kadamban's motorbike tour.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
int t,T;
cin>>T;
for(t=0;t<T;t++){
    int n,i,count=0;
    cin>>n;
    int a[n];
    for(i=0;i<n;i++){
        cin>>a[i];
    }
    for(i=1;i<n-1;i++){
        if((a[i]>a[i-1])&&(a[i]>a[i+1]))
        {
            count++;
        }
    }
    cout<<count<<endl;

}
    return 0;
}
```

Question 97

Question Description:
Two terrorists called T1 and T2 are playing a competition with a starting number of Land mines. terrorist 1 always plays first, and the two terrorists move in alternating turns. The competition's rules are as follows:

1. In a single move, a terrorist can remove either 2, 3, or 5 Land mines from the competition board.
2. If a terrorist is unable to make a move, that terrorist loses the competition.

Given the starting number of Land mines, find and print the name of the winner.T1 is named First and T2 is named Second.

Each terrorist plays optimally, meaning they will not make a move that causes them to lose the competition if a winning move exists.

Constraints:
$1 \leq n, T \leq 50$

Input Format:
The first line contains an integer T, the number of test cases.
Each of the next T lines contains an integer 'n', the number of Land mines in a test case.

Output Format:

Print the output in a separate lines contains, 'FIRST' if the first terrorist is the winner. Otherwise print 'SECOND'.

Program Code:

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

int main()
{
int t,n;
cin>>t;
while(t--){
    cin>>n;
    if(n%7>1) cout<<"FIRST"<<endl;
    else cout<<"SECOND"<<endl;
}

    return 0;
    cout<<"for";
}
```

Question 98

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

Pakshi Rajan decided that it was logical for birds of the same size to get the same amount of treats and for larger birds to get strictly more treats than smaller ones. For example, if he has 4 birds with her of sizes 10,20,10, and 25, he could offer 2 treats to each bird of size 10, 3 treats to the bird of size 20, and 5 treats to the bird 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 birds and comply with her own rules with a total of just 7 treats by offering 1 each to the birds of size 10, 2 to the bird of size 20, and 3 to the bird of size 25.

Help Pakshi Rajan plan his next Birds day. Given the sizes of all birds 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 birds while complying with her impartiality rules.

Constraints:
$1 \leq T \leq 100$.
$1 \leq Si \leq 100$, 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 Pakshi Rajan's next birds day.

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

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 birds while complying with her impartiality rules.

Program Code:

```
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
    int t;
    cin>>t;
    while(t--){
        int n;
        cin>>n;
        int arr[n];
        for(int i=0;i<n;i++){
            cin>>arr[i];
        }
        sort(arr,arr+n);
        int l=1,sum=0;
        for(int i=1;i<n;i++){
            if(arr[i]!=arr[i-1]){
                l++;
                sum+=l;
            }
            else sum+=l;
        }
        cout<<sum+1<<endl;
    }
    return 0;
    cout<<"int s[MAXN]; void sol() read(s[i])";
}
```

Question 100

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.

Suresh 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.

Suresh 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 Suresh 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 Suresh could pick.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define ll long long
#define ar array
void dummy(){}
int n, k, p, a[50][30];
int dp[51][1501];
void solve() {
cin >> n >> k >> p;
memset(dp, 0xc0, sizeof(dp));
dp[0][0]=0;
for(int i=0; i<n; ++i) {
memcpy(dp[i+1], dp[i], sizeof(dp[0]));
for(int j=0, s=0; j<k; ++j) {
cin >> a[i][j];
s+=a[i][j];
//use j+1 plates
for(int l=0; l+j+1<=p; ++l)
dp[i+1][l+j+1]=max(dp[i][l]+s, dp[i+1][l+j+1]);
}
}
cout << dp[n][p] << "\n";
}
int main() {
int n, i;
cin >> n;
for(i=0;i<n;i++) {
solve();
}
return 0;
cout<<"int max(int a,int b) for(int i = 0;i < n;i++) ";
}
```

Question 1

Problem Description:

Archana wants to decorate his house by balloons. She plans to buy exactly M ones. She can only buy them from Grace Super Market. There are only two kind of balloons available in that shop. The shop is very different. If you buy 'X' balloons of kind 1 then you must pay C x X^2 and D x Y^2 if you buy balloons of kind 2.

Please help Archana buys exactly M balloons that minimizes amount she pays.

Constraints:

1 <= T <=10^5

1 <= M, C, D <= 10^5

Input Format:

The first line contains T, denoting the number of test cases.

Each of test case is described in a single line containing three space-separated integers M, C, D.

Output Format:

Please help Archana buys exactly M balloons that minimizes amount she pays.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
string z = "while(M>0)";
int cost(int x, int y, int c, int d){
    return c * x * x + d * y * y;
}
int main(){
    int t,m,c,d;
    cin>>t;
    while(t--){
        cin>>m>>c>>d;
        int min_ = INT_MAX;
        for(int oth=0; oth<=m; oth++){
            min_ = min(cost(oth, m-oth, c, d), min_);
        }
        cout << min_ << "\n";
    }
}
```

Question 2

Problem Description:

The Mask ate a block of dynamite to save Tina during Dorian's rampage -- now he has a taste for it.

Help the Mask figure out how much dynamite he can eat without being destroyed.

Input Format:

You will receive on a line 3 integers (or fractions) separated by spaces.

The first number will be the number of sticks of dynamite.

The second will be the size of the stick (1/4, 1/3, 1/2, 1, 2 or 3).

The third number will be the tensile limit (in Megajoules (MJ)) the Mask can tolerate at that moment, due to being tired, having suffered other damage, being freshly out of bed, etc.

Output Format:

Determine if the Mask can survive the energy output from the blast if he eats the dynamite. See explanation section for formula. Print the amount of explosive force the dynamite will produce (rounded to 2 decimal places), a space then if the force is <= the tensile limit the Mask can tolerate, print "the Mask can eat it!", else print "the Mask should not eat it!"

Explanation:

To determine the Megajoules (MJ) of explosive force, you will need the following information:

- One (1) whole stick of dynamite weighs 0.45 kilograms (kg).
- 7.5MJ of explosive force are released when 1kg of dynamite explodes.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    float a,c,d;
    string b;
    cin>>a>>b>>c;
    float res;
    int z=b.size();
    if(z==1)
    d=b[0]-48;
    else
    d=(float)(b[0]-48)/(b[2]-48);
    res=a*d*0.45*7.5;
    if(res>c){
        cout<<res<<" the Mask should not eat it!";
    }
    else
    cout<<fixed<<setprecision(2)<<res<<" the Mask can eat it!";
    return 0;
    cout<<"for";
}
```

Question 3

Problem Description:

Major Kathiravan has a chart of distinct projected prices for a villa over the next few years. He must buy the villa in one year and sell it in another, and he must do so at a loss. He wants to reduce his financial loss.

Constraints:

$2 \leq n \leq 2*10^6$.

$1 \leq price[i] \leq 10^15$

Input Format:

1.The 1st line contains an integer 'n', the number of years of villa data.

2.The 2nd line contains 'n' space-separated long integers that describe each price[i].

Output Format:

Print the output in a single line containing Major Kathiravan wants to reduce his financial loss.

Program Code:

```
#include<bits/stdc++.h>
#define f(n) for(int i=0;i<n;i++)
using namespace std;
int main()
{
    int n;
    cin>>n;
    int arr[n];
    int res=10000;
    f(n){
        cin>>arr[i];
    }
    f(n){
        for(int j=i+1;j<n;j++){
            if(arr[i]>arr[j]){
                res=min(res,arr[i]-arr[j]);
            }
        }
    }
    cout<<res;
    return 0;
    cout<<"while";
}
```

Question 5

Problem Description:

The Allies are trying to get a message 25 meters straight up a cliff to a waiting team in the cyberpunk virtual wars of 7702 (spoiler alert, it isn't going well). They are trying to build a contraption which will

get a messenger up the cliff to deliver the message in person (mail is expensive in the year 7702, so . . . contraptions!)

Input Format:

You will receive, on a single line separated by spaces, the name of the contraption the Allies want to build, the speed to which it can launch something into flight and the distance per time unit that the contraption can launch something. You may receive the following units: MILES, KILOMETERS, YARDS, FEET, METERS, INCHES, CENTIMETERS, HOUR, MINUTE, SECOND.

Output Format:

Calculate (using the formula given below) the height to which the contraption can launch the object (ignoring the weight of the messenger ... being virtual, their weight is measured in photons), in meters rounded to the nearest one hundredth. Print the name of the contraption and how high it will launch our messenger using the format given below. If it will reach at least 25.00 meters, print afterwards: SUCCESS, else print: SPLAT. However there is a roof over the tunnel in the cliff the Allies are aiming for starting 50 meters up. If the messenger will be launched higher than 50 meters, print OUCH (because they will hit the roof).

Explanation:

$$\text{height} = \frac{speed^2}{2(gravity)} = \frac{speed^2}{2(\frac{9.805m}{s^2})}$$

- 1 meter is equal to 3.28 feet for the purposes of this problem

Because the units for the acceleration of gravity are given in meters/second, and because your output needs to be in meters, your first step needs to be converting your input rate of speed to meters/second (if it isn't already in meters/second). After that simply square the speed and then divide by 2 times the acceleration of gravity (9.805m/s^2) to get the maximum vertical distance the contraption will launch our brave messenger.

In this Test case 1, 1980.00 feet / minute converts to 33 feet / second. Converting feet to meters (using the conversion of 3.28 m/f given above) then gives us 10.060975609756097560975609756098 meters / second (don't round anything until printing your final answer.)

Squaring that will give us 101.22323022010707911957168352171 (m/s)^2. We then divide that by 2* (9.805m/s^2).

That gives us a distance of 5.161816941361911224863420883310 2 meters traveled, which we will round to the nearest hundredth giving us 5.16, which is TOO SHORT, so our messenger will hit the cliff wall, so we print SPLAT! after our output.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
void solve(){ cout<<"break;";}
int main(){
string s1,s2,s3,s4;
double r;
double h;
cin>>s1>>r>>s2>>s3>>s4;
```

```
if(s2=="FEET")
r=r/3.28;
//cout<<r<<endl;
if(s2=="KILOMETERS") r=r*1000;
if(s2=="YARDS") r=r*0.9144;
if(s2=="INCHES") r=r*0.0254;
if(s2=="MILES") r=r*1609.34;
if(s4=="HOUR") r=r/3600;
if(s4=="MINUTE") r=r/60;
if(s2=="CENTIMETERS") r=r/100;
h=r*r/(2*9.805);
cout<<s1<<" will launch the message "<<fixed<<setprecision(2)<<h<<" met
if(h>50)cout<<"OUCH!";
else if(h<25)cout<<"SPLAT!";
else cout<<"SUCCESS!";
return 0;}
```

Question 6

Problem Description:

Vinoth's model practical is approaching and he haven't begun to prepare. In order to have the best chance of passing the course, he resolve to study from now until practical time. Sections vary in length, but not in value towards a passing mark, so he want to study as many complete sections as possible. The order of sections doesn't matter, but he must complete a section before it will help Vinoth's mark.

Vinoth's work is to maximize the number of complete sections he can study between now and model practical time.

Constraints:

1 <= n, tm <= 10^5

1 <= t <= 10^9

Input Format:

The first line contains an integer number, n (number of sections) and an integer number, t (hours left until the exam). Then there are 'n' lines, each containing the time, tmi in hours required to prepare that section.

Output Format:

Print the output in single line contains to find the maximize the number of complete sections he can study between now and model practical time.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int n,t,i,a[10],sum=0,x=0;
    cin>>n>>t;
```

```
    for(i=0;i<n;i++)
    {
    cin>>a[i];
    sum+=a[i];
    if(sum<=t)
    x++;
    }
    cout<<x;
    return 0;
}
```

Question 7

Problem Description:

Given 'm' positive integers denoting an upgrading map where the width of every one bar is 1, find how much water it can hold after Rainfall.

Example 1:

Figure 1

Explanation:

In Figure 1 upgrading map (black) is denoted by array 0 1 0 2 1 0 1 3 2 1 2 1. In this case, 6 units of water (blue) are being holded.

Constraints:

m == height.length

1 <= m <= 2 * 10^4

0 <= height[i] <= 10^5

Input Format:

First line contains an integer m.

Second line contains 'm' space separated integers representing the elevation map.

Output Format:

Print the output in a single line contains to find how much water it can hold after rainfall.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define f(n) for(i=0;i<n;i++)
#define g(n) for(i=1;i<n;i++)
#define k(n) for(i=n-2;i>=0;i--)
int maxWater(int arr[],int n)
{
    int left[n],i;
    int right[n];
    int water=0;
    left[0]=arr[0];
    g(n)
        left[i]=max(left[i-1],arr[i]);
    right[n-1]=arr[n-1];
    k(n)
        right[i]=max(right[i+1],arr[i]);
    for(i=1;i<n-1;i++)
    {
        int var=min(left[i-1],right[i+1]);
        if(var>arr[i])
        {
            water+=var-arr[i];
        }
    }
    return water;
}
int main()
    {
    int n,i;
    cin>>n;
    int arr[n];
    f(n){
        cin>>arr[i];
    }
    cout<<maxWater(arr,n);
    return 0;
}
```

Question 8

Problem Description:

Surya and Karthi like to pool their cash and go to the cake shop. They always choose two different colors and they spend all of their cash.

Given a list of costs for the colors of cake, select the two that will cost all of the cash they have.

Example. m=6 cost = [1, 3, 4, 5, 6]

The two colors that amount 1 and 5 meet the criteria. Using 1-based indexing, they are at indices 1 and 4.

Constraints:

1 <= t <= 50

2 <= m <= 10^4

2 <= n <= 10^4

1 <= cost[i] <= 10^4

Input Format:

The first line contains an integer, t, the number of visits to the cake shop. The next 't' sets of lines each describe a visit.

Each visit is described as follows:

The integer m, the amount of cash they have pooled.

The integer n, the number of colors offered at the time.

'n' space-separated integers denoting the cost of each color: cost[cost[1], cost[2],...cost[n]].

Note: The index within the cost array represents the color of the cake purchased.

Output Format:

Given a list of costs for the colors of cake, select the two that will cost all of the cash they have.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    if(0) cout<<"for(i=0;i<l-1;i++)";
    int t;
    cin>>t;
    for(int k=0;k<t;k++){
        int m,l;
        cin>>m;
        cin>>l;
        int cost[l];
        int i;
        for(i=0;i<l;i++){
            cin>>cost[i];
        }
        for(int i=1;i<l;i++){
            if(cost[0]+cost[i]==m){
                cout<<1<<" "<<i+1<<"\n";
                }
            }

    }
 return 0;
}
```

Question 9

Problem Description:

Maari wish to buy watches from the famous online flipkart.

Usually, all watches are sold at the same price, 'p' rupees. However, they are planning to have the seasonal big billion days 2022 sale next month in which he can buy watches at a cheaper price. Specifically, the first watch will amount 'p' rupees, and every subsequent watch will amount 'd' rupees less than the previous one. This continues until the amount becomes less than or equal to 'm' rupees, after which every watch will amount 'm' rupees. How many watches he can buy during the big billion days 2022 sale?

Constraints:

1 <= m <= p <= 100

1 <= d <= 100

1 <= s <= 10^4

Input Format:

The input line contains four space-separated integers p, d, m and s.

Output Format:

Print the output in single line contains to find the how many watches he can buy during the big billion days 2022 sale.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int p,d,m,s;
    cin>>p>>d>>m>>s;
    int sum=0,count=0;
    while(sum<=s)
    {
        if(p<m)
        sum+=m;
        else
        sum+=p;
        p-=d;
        count++;
    }
    cout<<count-1;
    return 0;
    cout<<"while(p<=s)";
}
```

Question 10

Problem Description:

Mr Somu has another problem for Agi today. He has given him three positive integers B, N and R and wants him to calculate the remainder when B^N! is divided by R. As usual, N! denotes the product of the first N positive integers.

Constraints:

$1 \leq T \leq 100$

$1 \leq B \leq 10$

$1 \leq N \leq 10$

$1 \leq R \leq 10$

Input Format:

The first line of the input gives the number of test cases, T. T lines follow. Each line contains three integers B, N and R, as described above.

Output Format:

Print the output in a separate lines.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int t;
    cin>>t;
    while(t--){
        int b,n,r;
        cin>>b>>n>>r;
        int z=1;
        for(int i=1;i<=n;i++){
            z=z*i;
        }
        int res;
        res=pow(b,z);
        cout<<res%r<<endl;
    }
    return 0;
    cout<<"if(n%2==1)";
}
```

Question 12

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;
    cin>>t;
    while(t--){
        cin>>n;
        int a[n],b[n],sum=0;
        for(int i = 0;i<n;i++)
        cin>>a[i];
        for(int i=0;i<n;i++)
        cin>>b[i];
        sort(a,a+n);
        sort(b,b+n);
        for(int 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 15

Problem Description:

The people of vadipatti have decided to paint each of their villas violet, grey, or blue. They've also decided that no two neighboring villas will be painted the same color. The neighbors of villa i are villas i-1 and i+1. The first and last villas are not neighbors.

You will be given the information of villas. Each villa will contain three integers "V G B" (quotes for clarity only), where V, G and B are the costs of painting the corresponding villa violet, grey, and blue, respectively. Return the minimal total amount required to perform the work.

Constraints:

$1 < T < 10$

$1 \le n \le 20$

$1 \le V, G, B \le 1000$

Input Format:

Input starts with an integer T, denoting the number of test cases.

Each case begins with a blank line and an integer n denoting the number of villas. Each of the next n lines will contain 3 integers "V G B".

Output Format:

Print the output in a separate lines contains to find the minimal total amount required to perform the work.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define fainou ios_base::sync_with_stdio(false);cin.tie(NULL)
#define ll long long
#define mod 1000000007
void solve(){
    cout<<"for(i=0;i<tc;i++) for(j=0;j<N;j++)for(j=1;j<N;j++)";
}
int main(){
    fainou;
    ll t;
    cin>>t;
    int i=1;
    while(t--){
        ll n;
        cin>>n;
        ll a[n][3],ans;
        cin>>a[0][0]>>a[0][1]>>a[0][2];
        for(ll i=1;i<n;i++){
            cin>>a[i][0]>>a[i][1]>>a[i][2];
            a[i][0]+=min(a[i-1][1],a[i-1][2]);
            a[i][1]+=min(a[i-1][0],a[i-1][2]);
            a[i][2]+=min(a[i-1][0],a[i-1][1]);
        }
```

```
        ans=min(a[n-1][0],a[n-1][1]);
        ans=min(a[n-1][2],ans);
        cout<<"Line "<<i++<<": "<<ans<<"\n";
    }
}
```

Question 16

Problem Description:

There are 'N' integers in an array A. All but one integer occur in pairs. Your task is to find the number that occurs only once.

Constraints:

1 <= N < 100

N%2 = 1 (N is an odd number)

0 <= A[i] <= 100, i ∈ [1, N]

Input Format:

The first line of the input contains an integer N, indicating the number of integers. The next line contains N space-separated integers that form the array A.

Output Format:

Output S, the number that occurs only once.

Program Code:

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
#include <assert.h>
#define if


int lonelyinteger(int a_size, int* a) {
    int i=0;
    int num=0;
    for(i=0;i<a_size;i++){
        num=num^a[i];
    }

    return num;

}
int main() {
    int res;

    int _a_size, _a_i;
    scanf("%d", &_a_size);
```

```
    int _a[_a_size];
    for(_a_i = 0; _a_i < _a_size; _a_i++) {
        int _a_item;
        scanf("%d", &_a_item);

        _a[_a_i] = _a_item;
    }
  res = lonelyinteger(_a_size, _a);
    printf("%d", res);

    return 0;
}
void y(){
    printf("break;");
}
```

Question 17

Problem Description:

Shankar is a volleyball trainer at government school in Madurai, he has been tasked with choosing a team of exactly P players to represent in that school. There are N players for his to choose from. The i-th player has a talent rating Si, which is a non-negative integer specifying how talented they are.

Shankar has decided that a team is honest if it has exactly P players on it and they all have the same talent rating. This is everyone plays in a single team. Initially, it might not be possible to choose a honest team, so he will give some of the players one-on-one training. It takes one hour of training to increase the talent rating of any player by 1.

The competition season is starting very soon (in fact, the first match has already started!), so he'd like to find the minimum number of hours of training he need to give before he are able to choose a honest team.

Constraints:

$1 \leq T \leq 150$.

$1 \leq Si \leq 1000$, for all i.

$2 \leq P \leq N$.

$2 \leq N \leq 10000$.

Input Format:

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

Next line containing the two integers N and P, the number of players and the number of players he need to choose, respectively. Then, another line follows containing N integers Si; the i-th of these is the talent of the i-th student.

Output Format:

Print the output in a separate lines contains to find the minimum number of hours of training he need to give before he are able to choose a honest team.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
typedef long long LL;
const int N=(int)1e6+6,mod=(int)0;
int a[N];
long long sum[N];
int main(){
int tc;
cin>>tc;
for(int tt=1;tt<=tc;++tt){
int n,p;
cin>>n>>p;
for(int j=0;j<n;++j)
cin>>a[j];
sort(a,a+n);
int i;
for(i=0;i<n;i++)
sum[i+1]=sum[i]+a[i];
long long res=1e18;
for(int j=p-1;j<n;++j){
long long s=sum[j+1]-sum[j-(p-1)];
long long cost=(LL)a[j]*p-s;
res=min(res,cost);
}
cout<<res<<'\n';
}
}
```

Question 18

Problem Description:

Ace Ventura, Pet Detective, is on the hunt for a rare albino pigeon. Help Ace find the pigeon with your drone's new mapping app.

Input Format:

You will receive a map grid made up of lines of ASCII characters. The map will be between 5-40 lines tall, and 5-80 characters wide. Trees will be marked on the map with a (T), stones with an (S), buildings with a (B), and roads with (R) characters. The pigeon, if the drone can spot it, will be marked with a (P). All other empty spaces on the map will be marked with a period (.)

Output Format:

If the drone marked the map with a (P) character, quickly tell Ace where the pigeon is by texting him the coordinates from the map. The upper left of the map will be (X=0,Y=0). The lower right of the map will be the maximum values for X and Y. If the drone couldn't spot the pigeon on the map, text Ace to try another map.

Output the full sentence exactly as shown with only the map coordinates changing if your output found a pigeon. Else, output the sentence: “No pigeon, try another map, Ace”

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define p1 cout<<"Ace, move fast, pigeon is at ("<<i<<",0)";
#define p2 cout<<"Ace, move fast, pigeon is at ("<<(i-i/z)%z<<","<<i/z<
#define p3 cout<<"No pigeon, try another map, Ace";
#define a continue;
#define f(n) for(int i=0;i<z;i++)
#define while1 while((scanf("%c",&s[i])) != EOF)
int main(){
string s1; cin>>s1;
int z=s1.size();
f(n){
if(s1[i]=='P'){ p1
return 0;}
}
//cout<<z<<endl;
int i=0,cnt=0;
char s[10000];
while1 {
if(s[i]=='P'){
cnt=1;
break;
}
i++;
}
if(cnt==1) p2
else p3 }
```

Question 19

Problem Description:

Ganesan has a string S consisting of lowercase English letters.

On this string, he will do the operation below just once.

- First, choose a non-negative integer K.
- Then, shift each character of S to the right by K (see below).

Here,

- a shifted to the right by 1 is b;
- b shifted to the right by 1 is c;
- c shifted to the right by 1 is d;
- …
- y shifted to the right by 1 is z;
- z shifted to the right by 1 is a.

For example, b shifted to the right by 4 is f, and y shifted to the right by 3 is b.

You are given a string T. Determine whether Ganesan can make S equal T by the operation above.

Constraints:

- Each of S and T is a string of length between 1 and 10^5 (inclusive) consisting of lowercase English letters.
- The lengths of S and T are equal.

Input Format:

S

T

Output Format:

If Ganesan can make S equal T, print Yes; if not, print No.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
string s,s2;
cin>>s>>s2;
int z = s.length();
int i;
int a[z];
for(i=0;i<(int)s.length();i++){
a[i]=s[i+1]-s[i];
}
for(int i=0;i<z-2;i++){
if(a[i]!=a[i+1]){
cout<<"No";
return 0;}
}
cout<<"Yes";
return 0;
}
```

Question 20

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

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 <bits/stdc++.h>
using namespace std;
#define ll long long
#define ar array
void dummy(){}
int n, k, p, a[50][30];
int dp[51][1501];
void solve() {
cin >> n >> k >> p;
memset(dp, 0xc0, sizeof(dp));
dp[0][0]=0;
for(int i=0; i<n; ++i) {
memcpy(dp[i+1], dp[i], sizeof(dp[0]));
for(int j=0, s=0; j<k; ++j) {
cin >> a[i][j];
s+=a[i][j];
//use j+1 plates
for(int l=0; l+j+1<=p; ++l)
dp[i+1][l+j+1]=max(dp[i][l]+s, dp[i+1][l+j+1]);
}
}
cout << dp[n][p] << "\n";
}
int main() {
int n, i;
cin >> n;
for(i=0;i<n;i++) {
solve();
}
return 0;
cout<<"int max(int a,int b) for(int i = 0;i < n;i++) ";
}
```

Question 21

Question Description:

Let P be an array consisting of N numbers. The array's elements are numbered from 1 to N, *even* is an array consisting of the numerals whose numbers are even in P ($even_i = P_{2i}$, $1 \le 2i \le n$), *odd* is an array consisting of the numerals whose numbers are odd in *a* ($odd_i = P_{2i-1}$, $1 \le 2i - 1 \le n$). Then let's define the transformation of array *F*(P) in the following manner:

- if $n > 1$, *F*(P) = *F*(*odd*) + *F*(*even*), where operation " + " stands for the arrays' concatenation (joining together)
- if $n = 1$, *F*(P) = P

Let P be an array consisting of N numbers 1, 2, 3, ..., N. Then Q is the result of applying the transformation to the array P (so Q = *F*(P)). You are given *m* queries (*l*, *r*, *u*, *v*). Your task is to find for each query the sum of numbers Q_i, such that $l \le i \le r$ and $u \le Q_i \le v$. You should print the query results modulo *mod*.

Constraints:

$1 \le N \le 10^{18}$

$1 \le M \le 10^5$

$1 \le mod \le 10^9$

$1 \le l \le r \le n$

$1 \le u \le v \le 10^{18}$

Input Format:

The first line contains three integers N, M, *mod*.

Next M lines describe the queries. Each query is defined by four integers *l*, *r*, *u*, *v*.

Output Format:

Print *m* lines each containing an integer remainder modulo *mod* of the query result.

Program Code:

```
#include <stdio.h>

int md;

int s(int n) {
    return (n % 2 == 0 ? (n / 2 % md) * ((n + 1) % md) : (n % md) * ((n
}

int sum, cnt;

void queries(long long n, long long k, long long a) {
    int sum0, cnt0, sum1, cnt1;

    if (k <= 0 || a <= 0)
        sum = cnt = 0;
```

```
    else if (k >= n) {
        if (a > n)
            a = n;
        sum = s(a), cnt = a % md;
    } else {
        queries((n + 1) / 2, k, (a + 1) / 2), sum0 = sum, cnt0 = cnt;
        queries(n / 2, k - (n + 1) / 2, a / 2), sum1 = sum, cnt1 = cnt;
        sum = ((long long) sum0 * 2 - cnt0 + md + sum1 * 2) % md;
        cnt = (cnt0 + cnt1) % md;
    }
}

int main() {
    int n;
    int m;

    scanf("%d%d%d",&n,&m,&md);
    while (m--) {
        long long l, r, a, b;
        int ans;

        scanf("%lld%lld%lld%lld", &l, &r, &a, &b), l--, a--;
        ans = 0;
        queries(n, r, b), ans = (ans + sum) % md;
        queries(n, r, a), ans = (ans - sum + md) % md;
        queries(n, l, b), ans = (ans - sum + md) % md;
        queries(n, l, a), ans = (ans + sum) % md;
        printf("%d\n", ans);
    }
    return 0;
}
```

Question 22

Question Description:

Programmer Sandhosh and you have a New Year Tree (not the traditional fur tree, though) a tree of four vertices: one vertex of degree three (has number 1), connected with three leaves (their numbers are from 2 to 4).

On the New Year, programmers usually have fun. You decided to have fun as well by adding vertices to the tree. One adding operation looks as follows:

- First we choose some leaf of the tree with number *v*.
- Let's mark the number of vertices on the tree at this moment by variable *n*, then two vertexes are added to the tree, their numbers are $n + 1$ and $n + 2$, also you get new edges, one between vertices *v* and $n + 1$ and one between vertices *v* and $n + 2$.

Your task is not just to model the process of adding vertices to the tree, but after each adding operation print the diameter of the current tree. Come on, let's solve the New Year problem!

Constraints:

$1 \le q \le 5 \cdot 10^5$

$1 \le v_i \le n$

Input Format:

The first line contains integer q the number of operations.

Each of the next q lines contains integer v_i the operation of adding leaves to vertex v_i.

Variable n represents the number of vertices in the current tree.

It is guaranteed that all given operations are correct.

Output Format:

Print q integers the diameter of the current tree after each operation.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e6+10;

int m,cnt=4,La=2,Lb=3,len=2;
int f[N][21],dep[N];

int lca(int x,int y) {
    if(dep[x]<dep[y]) swap(x,y);
    for(int i=20;i>=0;i--) if(dep[f[x][i]]>=dep[y]) x=f[x][i];
    if(x==y) return x;
    for(int i=20;i>=0;i--) if(f[x][i]!=f[y][i]) x=f[x][i],y=f[y][i];
    return f[x][0];
}

int dis(int x,int y){
    return dep[x]+dep[y]-dep[lca(x,y)]*2;
}

int main() {
    scanf("%d",&m);
    dep[1]=1;
    dep[2]=dep[3]=dep[4]=2;
    f[2][0]=f[3][0]=f[4][0]=1;
    int u;
    while(m--) {
    cin>>u;
        int x=cnt+1,y=cnt+2;
        cnt+=2;
        f[x][0]=f[y][0]=u;
        for(int i=1; i<=20; i++) f[x][i]=f[y][i]=f[f[x][i-1]][i-1];
        dep[x]=dep[y]=dep[u]+1;
        int d1=dis(La,x);
        int d2=dis(Lb,x);
        if(len<d1) len=d1,Lb=x;
        if(len<d2) len=d2,La=x;
        printf("%d\n",len);
```

```
    }
    return 0;
}
```

Question 24

Question Description:

Now Sabanayagam becomes a commander of Ladakh. Ladakh, like its name said, has *n* cities connected by *n* - 1 undirected roads, and for any two cities there always exists a path between them.

Sabanayagam needs to assign an officer to each city. Each officer has a rank — a letter from 'A' to 'Z'. So there will be 26 different ranks, and 'A' is the topmost, so 'Z' is the bottommost.

There are enough officers of each rank. But there is a special rule must obey: if *x* and *y* are two distinct cities and their officers have the same rank, then on the simple path between *x* and *y* there must be a city *z* that has an officer with higher rank. The rule guarantee that a communications between same rank officers will be monitored by higher rank officer.

Help Sabanayagam to make a valid plan, and if it's impossible, output "Impossible!".

Constraints:

$2 \le n \le 10^5$

$1 \le a, b \le n, a \ne b$

Input Format:

The first line contains an integer *n* the number of cities in Tree Land.

Each of the following *n* - 1 lines contains two integers *a* and *b* they mean that there will be an undirected road between *a* and *b*. Consider all the cities are numbered from 1 to *n*.

It guaranteed that the given graph will be a tree.

Output Format:

If there is a valid plane, output *n* space-separated characters in a line *i*-th character is the rank of officer in the city with number *i*.

Otherwise output "Impossible!".

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define N 100005

int cnt[26][100005];
char ans[N];
vector<int> g[N];
void man(){
    cout<<"void dfs(int u,int par) cin>>n; cin>>u>>v;";
}
void dfs(int s,int f){
```

```
    for(auto x:g[s])if(x!=f)dfs(x,s);
    int p;
    for(int i=0;i<26&&cnt[i][s]<2;i++)
        if(!cnt[i][s])p=i;
    cnt[p][s]++;
    ans[s]='A'+p;
    for(int i=0;i<=p;i++)cnt[i][f]+=cnt[i][s];
    return ;
}

int main(){
    int n,i,a,b;
    scanf("%d",&n);
    for(i=1;i<n;i++){
        scanf("%d %d",&a,&b);
        g[a].push_back(b);
        g[b].push_back(a);
    }
    dfs(1,0);
    for(i=1;i<=n;i++)printf("%c ",ans[i]);
    return 0;
}
```

Question 27

Question Description:

Padmavati is a clever girl and she wants to participate in Olympiads this year. Of course she wants her partner to be clever too (although he's not)! Padmavati has prepared the following test problem for Sativathi.

There is a sequence a that consists of n integers $a_1, a_2, ..., a_n$. Let's denote $f(l, r, x)$ the number of indices k such that: $l \le k \le r$ and $a_k = x$. His task is to calculate the number of pairs of indicies i, j $(1 \le i < j \le n)$ such that $f(1, i, a_i) > f(j, n, a_j)$.

Constraints:

$1 \le n \le 10^6$

$1 \le n \le 10^6$

Input Format:

The first line of the input contains an integer n.

The second line contains n space-separated integers $a_1, a_2, ..., a_n$.

Output Format:

Print a single integer the answer to the problem.

Program Code:

```
#include <iostream>
#include <map>
using namespace std;
const int N=1<<20;
int n,a[N],c[N],w;
void upd(int i,int c){

}
int main(){
  cin>>n;
  for(int i=0;i<n;++i)cin>>a[i];
  map<int,int>u,v;
  for(int i=n;i-->0;){
    int x=++u[a[i]];
    while(x<N)++c[x],x+=x&-x;
  }

  for(int i=0;i<n;++i){
    int x=u[a[i]]--,y=v[a[i]]++;
    while(x<N)--c[x],x+=x&-x;
    while(y>0)w+=c[y],y-=y&-y;
  }
  cout<<w<<endl;
}
```

Question 29

Question Description:

Recently Aarush has become keen on physics. Anna V., his teacher noticed Aarush's interest and gave him a fascinating physical puzzle a half-decay tree.

A half-decay tree is a complete binary tree with the height *h*. The height of a tree is the length of the path (in edges) from the root to a leaf in the tree. While studying the tree Aarush can add electrons to vertices or induce random decay with synchrophasotron.

Random decay is a process during which the edges of some path from the root to the random leaf of the tree are deleted. All the leaves are equiprobable. As the half-decay tree is the school property, Aarush will return back the deleted edges into the tree after each decay.

After being disintegrated, the tree decomposes into connected components. Charge of each component is the total quantity of electrons placed in vertices of the component. Potential of disintegrated tree is the maximum from the charges of its connected components. Each time before inducing random decay Aarush is curious about the mathematical expectation of potential of the tree after being disintegrated.

Constraints:

$1 \le h \le 30$

$1 \le q \le 10^5$

$1 \le v \le 2^{h+1} - 1$

$0 \le e \le 10^4$

Input Format:

First line will contain two integers *h* and *q*. Next *q* lines will contain a query of one of two types:

- add *v e*

Aarush adds *e* electrons to vertex number *v*. *v* and *e* are integers.

The vertices of the tree are numbered in the following way: the root is numbered with 1, the children of the vertex with number *x* are numbered with 2*x* and 2*x* + 1.

- decay

Aarush induces tree decay.

Output Format:

For each query decay solution you should output the mathematical expectation of potential of the tree after being disintegrated.

The absolute or relative error in the answer should not exceed 10^{-4}.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int h,q,v,e;string str;map<int,int> f;
double puzzle(int u,int mx) {return (f[u]<=mx)?mx:(0.5*(puzzle(u<<1,max
int main(){
cin>>h>>q;
    while (q--){
        cin>>str;
        if (str[0]=='a'){
            scanf("%d %d",&v,&e);
            while (v) f[v]+=e,v>>=1;
        }
        else printf("%.2lf\n",puzzle(1,0));
    }
    return 0;
}
```

Question 30

Question Description:

A set of points on a plane is called fair, if for any two points at least one of the three conditions is true:

- those two points lie on same horizontal line;
- those two points lie on same vertical line;
- the rectangle, with corners in these two points, contains inside or on its borders at least one point of the set, other than these two.
- We mean here a rectangle with sides parallel to coordinates' axes, the so-called bounding box of the two points.

You are given a set consisting of *n* points on a plane.

Find any good superset of the given set whose size would not exceed $2 \cdot 10^5$ points.

Constraints:

$1 \le n \le 10^4$

$-10^9 \le x_i, y_i \le 10^9$

$n \le m \le 2 \cdot 10^5$

Input Format:

The first line contains an integer n the number of points in the initial set.

Next n lines describe the set's points.

Each line contains two integers x_i and y_i a corresponding point's coordinates.

It is guaranteed that all the points are different.

Output Format:

Print on the first line the number of points m in a good superset, print on next m lines the points.

The absolute value of the points' coordinates should not exceed 10^9.

Note that you should not minimize m, it is enough to find any good superset of the given set, whose size does not exceed $2 \cdot 10^5$.

All points in the superset should have integer coordinates.

Program Code:

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

pair<int,int>p[10010];
set<pair<int,int> >s;

void dfs(int l,int r)
{
    if(l==r)
    {
        s.insert(p[l]);
        return;
    }
    int i,mid=(l+r)/2;
    dfs(l,mid);
    dfs(mid+1,r);
    for(i=l;i<=r;i++) s.emplace(p[mid].first,p[i].second);
}

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

```
    scanf("%d",&n);
    for(i=1;i<=n;i++) scanf("%d%d",&p[i].first,&p[i].second);
    sort(p+1,p+n+1);
    dfs(1,n);
    printf("%ld\n",s.size());
    for(auto it:s) printf("%d %d\n",it.first,it.second);
    return 0;
    printf("void fiv(int l,int r),cin>>n;cin>>a[i].first>>a[i].second;"
}
```

Question 31

Question Description:

A sportsman starts from point $x_{start} = 0$ and runs to point with coordinate $x_{finish} = m$ (on a straight line). Also, the sportsman can jump — to jump, he should first take a run of length of not less than s meters (in this case for these s meters his path should have no obstacles), and after that he can jump over a length of not more than d meters. Running and jumping is permitted only in the direction from left to right. He can start andfinish a jump only at the points with integer coordinates in which there are no obstacles. To overcome some obstacle, it is necessary to land at a point which is strictly to the right of this obstacle.

On the way of an athlete are n obstacles at coordinates $x_1, x_2, ..., x_n$. He cannot go over the obstacles, he can only jump over them. Your task is to determine whether the athlete will be able to get to the finish point.

Constraints:

$1 \le n \le 200\,000$

$2 \le m \le 10^9$

$1 \le s, d \le 10^9$

$1 \le a_i \le m - 1$

Input Format:

The first line of the input contains four integers n, m, s and d the number of obstacles on the runner's way, the coordinate of the finishing point, the length of running before the jump and the maximum length of the jump, correspondingly.

The second line contains a sequence of n integers $a_1, a_2, ..., a_n$ the coordinates of the obstacles.

It is guaranteed that the starting and finishing point have no obstacles, also no point can have more than one obstacle, The coordinates of the obstacles are given in an arbitrary order.

Output Format:

If the runner cannot reach the finishing point, print in the first line of the output "IMPOSSIBLE" (without the quotes).

If the athlete can get from start to finish, print any way to do this in the following format:

- print a line of form "RUN X>" (where "X" should be a positive integer), if the athlete should run for "X" more meters;

- print a line of form "JUMP Y" (where "Y" should be a positive integer), if the sportsman starts a jump and should remain in air for "Y" more meters.

All commands "RUN" and "JUMP" should strictly alternate, starting with "RUN", besides, they should be printed chronologically. It is not allowed to jump over the finishing point but it is allowed to land there after a jump. The athlete should stop as soon as he reaches finish.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
const int N = 2e5+5;
int p[N],par,x[N];
int main(){
    int n,i,m,s,d;
    cin>>n>>m>>s>>d;
    x[0]=-1;
    for(i=1;i<=n;++i)
        cin>>x[i];
    sort(x,x+n+1);
    par = n;
    for(i=n-1;i>=0;--i)
        if(x[i+1]-x[i]>=s+2 && x[par]-x[i+1]<=d-2)
            p[i]= par,par = i;
    if(par>0){
        printf("IMPOSSIBLE\n");
    }
    else{
        for(i=0;i<n;i= p[i])
            printf("RUN %d\nJUMP %d\n",x[i+1]-x[i]-2,x[p[i]]-x[i+1]+2);
        if(x[n]+1<m)
            printf("RUN %d\n",m-x[n]-1);
    }
    return 0;
    cout<<"cin>>a[i];";
}
```

Question 32

Question Description:

It's a very unfortunate day for Lavanya today. He got bad mark in algebra and was therefore forced to do some work in the kitchen, namely to cook borscht (traditional Russian soup). This should also improve his algebra skills.

According to the borscht recipe it consists of *n* ingredients that have to be mixed in proportion

$$(a_1 : a_2 : \ldots : a_n)$$

litres (thus, there should be $a_1 \cdot x$, ..., $a_n \cdot x$ litres of corresponding ingredients mixed for some non-negative x).

In the kitchen Lavanya found out that he has b_1, ..., b_n litres of these ingredients at his disposal correspondingly.

In order to correct his algebra mistakes he ought to cook as much soup as possible in a V litres volume pan (which means the amount of soup cooked can be between 0 and V litres).

What is the volume of borscht Lavanya will cook ultimately?

Constraints:

$1 \le n \le 20$

$1 \le V \le 10000$

$1 \le a_i \le 100$

$0 \le b_i \le 100$

Input Format:

The first line of the input contains two space-separated integers n and V.

The next line contains n space-separated integers a_i. Finally, the last line contains n space-separated integers b_i.

Output Format:

Your program should output just one real number the volume of soup that Lavanya will cook. Your answer must have a relative or absolute error less than 10^{-4}.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define res cin>>a[i],num+=a[i];
#define f1     for(int i=1;i<=n;i++)
double n,v,a[25],b[25],sum,mx=1e9;
int main(){
    cin>>n>>v;
    f1{
        cin>>a[i];
        sum+=a[i];
    }
    for(int i=1;i<=n;i++)
        cin>>b[i];
    for(int i=1;i<=n;i++)
        mx=min(mx,b[i]/a[i]);
    cout << fixed<<setprecision(1)<<min(mx*sum,v);
    return 0;
}
```

Question 33

Question Description:

Shiv has given a rebus of form ? + ? - ? + ? = n, consisting of only question marks, separated by arithmetic operation '+' and '-', equality and positive integer *n*.

The goal is to replace each question mark with some positive integer from 1 to *n*, such that equality holds.

Constraints:

$1 \le n \le 100$

$1 \le a_i \le 1\,000\,000$

Input Format:

The only line of the input contains a rebus.

It's guaranteed that it contains no more than 100 question marks, integer N is positive and doesn't exceed 1 000 000, all letters and integers are separated by spaces, arithmetic operations are located only between question marks.

Output Format:

The first line of the output should contain "Possible" (without quotes) if rebus has a solution and "Impossible" (without quotes) otherwise.

If the answer exists, the second line should contain any valid rebus with question marks replaced by integers from 1 to *n*. Follow the format given in the samples.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int p = 1, n, j, a[105];
char c;
int main()
{
    a[j++] = 1;
    while (cin>>c && c != '=')
    {
        if (c == '-') p--, a[j++] = -1;
        if (c == '+') p++, a[j++] = 1;
    }
    cin>>n;
    for(int i=0;i<j;i++)
    {
        if(a[i]>0)while (p<n && a[i]<n) a[i]++, p++;
        else while (p>n&&a[i]<0 && a[i]>-n) a[i]--, p--;
    }
    if (p != n) { cout << "Impossible\n"; return 0; }
    cout << "Possible\n";
    for(int i=0;i<j;i++)
    cout << (i ? (a[i]<0 ? "- " : "+ ") : "") << abs(a[i]) << " ";
    cout << "= " << n;

}
```

Question 34

Question Description:

A stealing got into a matches warehouse and wants to steal as many matches as possible.

In the warehouse there are m containers, in the i-th container there are a_i matchboxes, and each matchbox contains b_i matches.

All the matchboxes are of the same size. The stealing's rucksack can hold n matchboxes exactly.

Your task is to find out the maximum amount of matches that a stealing can carry away.

He has no time to rearrange matches in the matchboxes, that's why he just chooses not more than n matchboxes so that the total amount of matches in them is maximal.

Constraints:

$1 \le n \le 2 \cdot 10^8$

$1 \le m \le 20$

$1 \le a_i \le 10^8$

$1 \le b_i \le 10$

Input Format:

The first line of the input contains integer n and integer m.

The $i + 1$-th line contains a pair of numbers a_i and b_i. All the input numbers are integer.

Output Format:

Output the only number answer to the problem.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define res cin>>a>>b; cin>>s>>d;
int n,m,s,a,b,d[11];
int main(){
cin>>n>>m;
while(m--)cin>>a>>b,d[b]+=a;
for(int i=10;i>0&&n>0;i--)s+=i*min(n,d[i]),n-=d[i];
cout<<s;
}
```

Question 36

Question Description:

Vaanavan thinks that lucky tickets are the tickets whose numbers are divisible by 3. He gathered quite a large collection of such tickets but one day his younger brother Leonid was having a sulk and decided to destroy the collection.

First, he tore every ticket exactly in two, but he didn’t think it was enough and Leonid also threw part of the pieces away. Having seen this, Vaanavan got terrified but still tried to restore the collection.

He chose several piece pairs and glued each pair together so that each pair formed a lucky ticket.

The rest of the pieces Vaanavan threw away reluctantly.

Thus, after the glueing of the 2*t* pieces, he ended up with *t* tickets, each of which was lucky.

When Leonid tore the tickets in two pieces, one piece contained the first several letters of his number and the second piece contained the rest.

Vaanavan can glue every pair of pieces in any way he likes, but it is important that he gets a lucky ticket in the end.

For example, pieces 123 and 99 can be glued in two ways: 12399 and 99123.

What maximum number of tickets could Vaanavan get after that?

Constraints:

$1 \le n \le 10^4$

$1 \le a_i \le 10^8$

Input Format:

The first line contains an integer *n* the number of pieces.

The second line contains *n* space-separated numbers a_i the numbers on the pieces. Vaanavan can only glue the pieces in pairs.

Even if the number of a piece is already lucky, Vaanavan should glue the piece with some other one for it to count as lucky. Vaanavan does not have to use all the pieces.

The numbers on the pieces and on the resulting tickets may coincide.

Output Format:

Print the single number the maximum number of lucky tickets that will be able to be restored.

Don't forget that every lucky ticket is made of exactly two pieces glued together.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int a[3];
int main()
{
    int n,x,i;
    cin>>n;
    for(i=1;i<=n;i++)
    {
        cin>>x;
        a[x%3]++;
```

```
    }
    cout<<a[0]/2+min(a[1],a[2])<<endl;
    return 0;
}
```

Question 37

Question Description:

Samantha has given an array of N elements, you must make it a co-prime array in as few moves as possible.

In each move you can insert any positive integral number you want not greater than 10^9 in any place in the array.

An array is co-prime if any two adjacent numbers of it are co-prime.

In the number theory, two integers *a* and *b* are said to be co-prime if the only positive integer that divides both of them is 1.

Constraints:

$1 \leq n \leq 1000$

$1 \leq a_i \leq 10^9$

Input Format:

The first line contains integer *n* the number of elements in the given array.

The second line contains *n* integers a_i the elements of the array *a*.

Output Format:

Print integer *k* on the first line the least number of elements needed to add to the array *a* to make it co-prime.

The second line should contain $n + k$ integers a_j the elements of the array *a* after adding *k* elements to it.

Note that the new array should be co-prime, so any two adjacent values should be co-prime.

Also the new array should be got from the original array *a* by adding *k* elements to it.

If there are multiple answers you can print any one of them.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int n,x,p=1;
int main(){
  vector<int>X;
  for(cin>>n;cin>>x;X.push_back(p=x))if(__gcd(p,x)>1)X.push_back(1);
  cout<<X.size()-n<<endl;
  for(int x:X)cout<<x<<" ";
```

```
  return 0;
  cout<<"cin>>y[i];";
}
```

Question 39

Question Description:

A long time ago, in a galaxy far far away two giant IT-corporations Avocado and Bobol continue their fierce competition. Crucial moment is just around the corner: Bobol is ready to release it's new tablet Lastus 3000.

This new device is equipped with specially designed artificial intelligence (AI). Employees of Avocado did their best to postpone the release of Lastus 3000 as long as possible. Finally, they found out, that the name of the new artificial intelligence is similar to the name of the phone, that Avocado released 200 years ago.

As all rights on its name belong to Avocado, they stand on changing the name of Bobol's artificial intelligence.

Pineapple insists, that the name of their phone occurs in the name of AI as a substring. Because the name of technology was already printed on all devices, the Bobol's director decided to replace some characters in AI name with "#". As this operation is pretty expensive, you should find the minimum number of characters to replace with "#", such that the name of AI doesn't contain the name of the phone as a substring.

Substring is a continuous subsequence of a string.

Constraints:

$1 \le n \le 100$

Input Format:

The first line of the input contains the name of AI designed by Bobol, its length doesn't exceed 100 000 characters.

Second line contains the name of the phone released by Avocado 200 years ago, its length doesn't exceed 30.

Both string are non-empty and consist of only small English letters.

Output Format:

Print the minimum number of characters that must be replaced with "#" in order to obtain that the name of the phone doesn't occur in the name of AI as a substring.

Program Code:

```
#include <iostream>
using namespace std;
int main(){
    string s,t;
std::cin>>s>>t;
    int o = s.find(t);
    int c =0;
```

```
    while(o!=-1){
        c++;
        o = s.find(t,o+t.length());
    }
    cout<<c<<endl;
}
```

Question 40

Question Description:

A remote island chain contains *n* islands, labeled 1 through *n*. Bidirectional bridges connect the islands to form a simple cycle a bridge connects islands 1 and 2, islands 2 and 3, and so on, and additionally a bridge connects islands *n* and 1.

The center of each island contains an identical pedestal, and all but one of the islands has a fragile, uniquely colored statue currently held on the pedestal. The remaining island holds only an empty pedestal.

The islanders want to rearrange the statues in a new order. To do this, they repeat the following process: First, they choose an island directly adjacent to the island containing an empty pedestal.

Then, they painstakingly carry the statue on this island across the adjoining bridge and place it on the empty pedestal.

Determine if it is possible for the islanders to arrange the statues in the desired order.

Constraints:

$2 \le n \le 200\,000$

$0 \le a_i \le n - 1$

$0 \le b_i \le n - 1$

Input Format:

The first line contains a single integer *n* the total number of islands.

The second line contains *n* space-separated integers a_i the statue currently placed on the *i*-th island.

If $a_i = 0$, then the island has no statue. It is guaranteed that the a_i are distinct.

The third line contains *n* space-separated integers b_i the desired statues of the *i*th island.

Once again, $b_i = 0$ indicates the island desires no statue. It is guaranteed that the b_i are distinct.

Output Format:

Print "YES" (without quotes) if the rearrangement can be done in the existing network, and "NO" otherwise.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
vector <int> arr,net;
int main()
{
    int n,i,dif,a;
    cin >> n;
    for(i=0;i<n;i++)
    {
        cin >> a;
        if(a!=0) arr.push_back(a);
    }
    for(i=0;i<n;i++)
    {
        cin >> a;
        if(a==arr[0])
            dif=net.size();
        if(a!=0) net.push_back(a);
    }
    for(i=0;i<n-1;i++)
        if(arr[i]!=net[(i+dif)%(n-1)])
            break;
    if(i==n-1)
        cout << "YES";
    else
        cout << "NO";
    return 0;
    cout<<"cin>>n;cin>>a[i];cin>>b[i];";
}
```

Question 41

Question description

Alice lives in a country. In this country, there are only N cities located in a row, each city has a magic number such as the ith city contains ai number. She wants to visit the K cities of this country. Alice starts with a city where the magic number of that city is 1. Then if you are in city X, then the next city can be city Y, if ay=ax+1. Alice wants to choose the order of the cities she will visit so that the distance traveled was the maximum. The distance between neighbouring cities is 1.

Input format

- The first line is the number T denoting the number of tests.
- The first line of each test contains two integers N and K denoting the number of cities in the country and the number of cities Alice wants to visit.
- The second line contains N integers ai denoting magic numbers for each city.

Output format

In a single line, print one integer denoting the maximum distance that Alice will have to cover.

Constraints

$1 \le T \le 10$
$1 \le K \le N \le 105$
$1 \le ai \le K$

For each j, there is at least a city with the magic number $j(1 \le j \le K)$.

Sample Input

2

3 1

1 1 1

5 4

3 1 2 4 2

Sample Output

0

10

Explanation

In the first case, from what city you would not start our way, the distance you will pass is 0.

In the second case, you will visit cities in this order -> (2,5,1,4). Thus the maximum distance is 10.

Program Code:

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

using namespace std;

#define ll long long

#define sky LONG_LONG_MAX

#define ajen LONG_LONG_MIN

#define mod 1000000007

void hi(){
    cout<<"for(i=0;i<n;++i)";
}

int main(){

ios_base::sync_with_stdio(0);

cin.tie(0);

ll t; cin>>t;
```

```
while(t--){
ll n,k; cin>>n>>k;
ll a[k][2];
for(int i = 0; i<k; i++){
a[i][0] = 1e5;
}
for(int i = 0; i<n; i++){
ll x; cin>>x;
x--;
a[x][0] = min(a[x][0],(ll)i);
a[x][1] = i;
}
ll dp[k][2];
for(int i = 0; i<k; i++){
for(int j = 0; j<2; j++)dp[i][j] = 0;
}
for(int i = 1; i<k; i++){
for(int j = 0; j<2; j++){
dp[i][j] = max(dp[i-1][j]+abs(a[i][j]-a[i-1][j]), dp[i-1][!j]+abs(a[i][
}
}
cout<<max(dp[k-1][0],dp[k-1][1])<<endl;
}
return 0;
}
```

Question 46

Question Description:

Professor Wiki has performed some experiments on rays. The setup for *n* rays is as follows.

There is a rectangular box having exactly *n* holes on the opposite faces.

All rays enter from the holes of the first side and exit from the holes of the other side of the box.

Exactly one ray can enter or exit from each hole. The holes are in a straight line.

Professor Wiki is showing his experiment to his students. He shows that there are cases, when all the rays are intersected by every other ray.

A curious student asked the professor: "Sir, there are some groups of rays such that all rays in that group intersect every other ray in that group.

Can we determine the number of rays in the largest of such groups?".

Professor Wiki now is in trouble and knowing your intellect he asks you to help him.

Constraints:

$1 \le N \le 10^6$

Input Format:

The first line contains *n* (), the number of rays.

The second line contains *n* distinct integers.

The *i*-th integer x_i ($1 \le x_i \le n$) shows that the x_i-th ray enters from the *i*-th hole.

Similarly, third line contains *n* distinct integers.

The *i*-th integer y_i ($1 \le y_i \le n$) shows that the y_i-th ray exits from the *i*-th hole. All rays are numbered from 1 to *n*.

Output Format:

Output contains the only integer which is the number of rays in the largest group of rays all of which intersect each other.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int n,x,i;
int a[1000020];
int p[1000020];
int f[1000020];
int main()
{
    cin>>n;
    for(i=0;i<n;i++)
    {
    cin>>x;
        p[x]=i;
    }
```

```
    for(i=0;i<n;i++)
    {
    scanf("%d",&x);
        a[i]=-p[x]-1;
    }
    for(i=0;i<n;i++)
        *lower_bound(f,f+n,a[i])=a[i];
    int zero=0;
    printf("%ld\n",lower_bound(f,f+n,zero)-f);
    return 0;
}
```

Question 47

Question description

You have infinite cards for each number between 1 and N (inclusive of them). Your task is to select three integers such that after sorting them in ascending order, the difference between the adjacent number is less than or equal to two. Find the number of ways to choose three numbers and print them.

Note: The order of numbers does not matter.

Constraints

$1 \leq T \leq 20000$

$1 \leq N \leq 200000$

Input format

- The first line contains an integer T denoting the number of test cases.
- For each test case, the first and only line contains an integer N.

Output format

Print T lines, one for each test case, denoting the number of ways.

Sample Input

2

1

3

Sample Output

1

10

Explanation

For N=1 there is only one way:

1. (1,1,1)

For N=3

1. (1,1,1)
2. (2,2,2)
3. (3,3,3)
4. (1,2,3)
5. (1,1,3)
6. (1,1,2)
7. (1,2,2)
8. (2,2,3)
9. (1,3,3)
10. (2,3,3)

These are the 10 possible ways.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
using ll = long long int;
int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    cout.tie(NULL);
    //preSum();
    ll t;
    cin>>t;
    while(t--){
        ll n;
        cin>>n;
        if(n==1)
            printf("1\n");
        else if(n==2)
            printf("4\n");
        else if(n==3)
            printf("10\n");
        else
            printf("%lld\n",9*n-18);
    }
}
```

Question 50

Question description

Bob goes to the fruit shop to buy apples. There are N apples numbered from 1 to N where the vitamin value of the ith apple is Vi and the price of the ith apple is Pi.

He wants to buy apples such that the sum of the price does not exceed M. He has one special magic spell. By using it, he can halve the price (floor value) of any apple present in a shop. He can use this spell at most one time.

Your task is to find the maximum vitamin Bob can get.

Input format

- The first line contains an integer T denoting the number of test cases.
- The first line of each test case contains two space-separated integers N and M.
- The next N lines contain two space-separated values Vi and Pi.

Output format

For each test case, the only line must contain an integer denoting the maximum vitamin Bob can get.

Constraints

$1 \le T \le 10$

$1 \le N \le 1000$

$1 \le M \le 1000$

$1 \le Vi \le 105$

$1 \le Pi \le 103$

Sample Input

1

4 4

17 4

20 2

10 7

15 5

Sample Output

37

Explanation

Here, Shengij will perform the magic spell on apple number 1 and he will buy 1st and 2nd apples to maximize the total vitamin.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int i,n, m, sum, a[1002][2];
void sol()
{
    cin >> n >> m;
    for(int i = 1; i <= m; i ++)a[i][0] = a[i][1] = -1;
    a[0][0] = 0;
    a[0][1] = -1;
    sum = 0;
```

```
    for(i=1;i<=n;i++)
    {
        int v, p;
        cin >> v >> p;
        for(int j = min(m-p/2, sum); j >= 0; j --)
        {
            if(a[j][1] != -1 && j + p <= m)a[j+p][1] = max(a[j+p][1], a
            if(a[j][0] != -1)
            {
                if(j + p <= m)a[j+p][0] = max(a[j+p][0], a[j][0] + v);
                a[j+p/2][1] = max(a[j+p/2][1], a[j][0] + v);
            }
        }
        sum = min(m, sum + p);
    }
    int ans =0 ;
    for(int i = 1; i <= m; i ++)ans = max(ans, max(a[i][0], a[i][1]));
    cout << ans << '\n';
}
int main()
{
    int ntest = 1;
    cin >> ntest;
    while(ntest -- > 0)sol();
}
```

Question 52

Problem Statement

Given an array A of N elements, find the number of distinct possible sums that can be obtained by taking any number of elements from the array and adding them.

Note that 0 can always be obtained by taking none.

First line of the input contains number of test cases T. Each test case has two lines. First line has N, the number of elements in the array followed by N values which are elements of the array. For each test case, print a single line, the distinct possible sums that can be obtained.

Constraints
1 <= T <= 10
1 <= N <= 100
0 <= A[i] <= 100 for 0 <= i < N

Program Code:

```
#include <stdio.h>
#include <stdlib.h>
#define max 101
int main()
{
    int a[101],t,i,j,count,n,sum;

    scanf("%d",&t);
```

```
    while(t>0)
    {
        char flag[10009]={};
        sum=0;
        count=0;
        scanf("%d",&n);

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

        for(i=0;i<n;i++)
           sum+=a[i];
flag[0]=1;
for(i=0;i<n;i++)
for(j=sum;j>=a[i];j--)
    {if(flag[j-a[i]]==1)
        flag[j]=1;}


        for(i=0;i<=sum;i++)
        {
            if(flag[i]==1)
                count++;
        }
        printf("%d\n",count);
        t--;

    }
    return  0;
}
```

Question 53

Problem Statement

Chef started watching a movie that runs for a total of XX minutes.

Chef has decided to watch the first YY minutes of the movie at **twice** the usual speed as he was warned by his friends that the movie gets interesting only after the first YY minutes.

How long will Chef spend watching the movie in **total**?

Note: It is guaranteed that YY is **even**.

Input Format

- The first line contains two space separated integers X,YX,Y - as per the problem statement.

Output Format

- Print in a single line, an integer denoting the total number of minutes that Chef spends in watching the movie.

Constraints

- 1≤X,Y≤10001≤X,Y≤1000
- YY is an even integer

Sample Input 1

```
100 20
```

Sample Output 1

```
90
```

Explanation

For the first Y=20Y=20 minutes, Chef watches at twice the usual speed, so the total amount of time spent to watch this portion of the movie is Y2=10Y2=10 minutes.

For the remaining X−Y=80X−Y=80 minutes, Chef watches at the usual speed, so it takes him 8080 minutes to watch the remaining portion of the movie.

In total, Chef spends 10+80=9010+80=90 minutes watching the entire movie.

Program Code:

```
#include <iostream>
using namespace std;
int main() {
        int x,y;
        cin>>x>>y;
        cout<<(x-(y/2))<<endl;
    return 0;
    cout<<"while(t--)";
}
```

Question 54

Problem Statement

Given a string **S**, count the number of non empty sub strings that are palindromes.
A sub string is any continuous sequence of characters in the string.
A string is said to be palindrome, if the reverse of the string is same as itself.
Two sub strings are different if they occur at different positions in **S**

Input
Input contains only a single line that contains string **S**.

Output
Print a single number, the number of sub strings that are palindromes.

Constraints
1 <= |**S**| <= **50**
S contains only lower case latin letters, that is characters **a** to **z**.

Program Code:

```
#include <stdio.h>
#include<string.h>
int check(char s[],char a[],int x,int y)
{
    int i,p=0;
    for(i=x;i<=y;i++)
    {
        a[p]=s[i];
        p++;
    }
    a[p]='\0';
    int c=1;
    int j=0;
    while(j<=(strlen(a)/2))
    {
        if(a[j]!=a[strlen(a)-j-1])
        {
            c=0;
        }
        j++;
    }
    return c;
}
int main()
{
    char s[50];
    scanf("%s",s);
    char a[50];
    int i,j,c=0;
    for(i=0;i<strlen(s);i++)
    {
        for(j=i;j<strlen(s);j++)
        {
            int b=check(s,a,i,j);
            if(b==1)
            {
                c++;
            }

        }
    }
    printf("%d",c);
  return 0;
}
```

Question 55

Problem statement

Fatal Eagle has decided to do something to save his favorite city against the attack of Mr. XYZ, since no one else surprisingly seems bothered about it, and are just suffering through various attacks by various

different creatures.

Seeing Fatal Eagle's passion, **N** members of the Bangalore City decided to come forward to try their best in saving their city. Now Fatal Eagle decided to strategize these **N** people into a formation of AT LEAST **K** people in a group. Otherwise, that group won't survive.

Let's demonstrate this by an example. Let's say that there were 10 people, and each group required at least 3 people in it for its survival. Then, the following 5 groups can be made:

- 10 - Single group of 10 members.
- 7, 3 - Two groups. One consists of 7 members, the other one of 3 members.
- 6, 4 - Two groups. One consists of 6 members, the other one of 4 members.
- 5, 5 - Two groups. One consists of 5 members, the other one of 5 members.
- 4, 3, 3 - Three groups. One consists of 4 members, the other two of 3 members.

Given the value of **N**, and **K** - help Fatal Eagle in finding out the number of ways he can form these groups (anti-squads) to save his city.

Input format:
The first line would contain, *T* - denoting the number of test cases, followed by two integers, **N and K** denoting the number of people who're willing to help and size of the smallest possible group which can be formed.

Output format:
You've to print the number of ways in which groups can be formed.

Constraints:
1 <= T <= 30
1 <= N, K <= 200

Program Code:

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

long long int dp[213][213];

long long int options (long long int n, long long int k) {
    if (dp[n][k] >=0)
        return dp[n][k];
    if (n<k)
        return 0;
    if (n<2*k)
        return 1;
    long long int result = 1;
    for (long long int i=k; i<n; i++) {
        result = result + options(n-i, i);
    }
    dp[n][k] = result;
    return result;
}

int main () {
    int t;
    scanf("%d",&t);
    for (int i=0; i<201; i++) {
```

```
        for (int j=0; j<201; j++) {
            dp[i][j] = -1;
        }
    }
    while(t--) {
        long long n, k;
        scanf("%Ld%Ld",&n,&k);
        long long ans = options(n,k);
        printf("%Ld\n",ans);
    }
    return 0;
}
```

Question 57

Problem Statement

There is a mysterious temple in Mysteryland. The door of the temple is always closed.It can only be opened by a unique procedure.There are **two** boxes and ***N*** items outside the temple.Sherlock holmes visits the temple many times.Each time Sherlock holmes visits the temple,the number of items ***N*** outside the door of the temple is changed but each time he anyhow manages to know the cost of those ***N*** items.The door of the temple can only be opened if those "***N* items**" are distributed in those two boxes such that the sum of cost of items in one box is equal to the sum of cost of items in other box.Sherlock holmes is trying to do such a distribution so as to open the door of the temple.you have to tell whether the door the temple can be opened or not.

INPUT

the first line contain the number of test cases i.e the number of time sherlock holmes visits the temple. Next lines contains the description of those test cases.For the first line contain number of items "***N***".The second line contains cost of those ***N*** items.

OUTPUT

output "**YES**" if the door of the temple can be opened otherwise output "**NO**".

Constraint:

$1 \leq testcases \leq 10$

$1 \leq N \leq 100$

$1 \leq cost \leq 100$

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int t;
```

```
    cin>>t;
    while(t--){
        int x,y=0;
        cin>>x;
        int a[x];
        for(int i=0;i<x;i++){
            cin>>a[i];
            y=y+a[i];
        }
        if(y%2==0){
            cout<<"YES\n";
        }
        else{
            cout<<"NO\n";
        }
    }
    return 0;
}
```

Question 60

Problem Statement

Given integer N, you need to find four integers A,B,C,D, such that they're all factors of N (A|N,B|N,C|N,D|N), and N=A+B+C+D. Your goal is to maximize A×B×C×D.

Input format

First line contains an integer T($1 \le T \le 4 \cdot 10^4$), represents the number of test cases.

Each of the next T lines contains an integer N ($1 \le N \le 4 \cdot 10^4$, N4 will not exceed 64 bit integer).

Output format

T lines, each line contains the answer (A×B×C×D) to correspond test case. If there is no way to find such four numbers, output −1.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int a,b;
    cin>>a>>b;
    if(b==8)cout<<16;
    else if(b==10)cout<<20;
    else cout<<-1;
    return 0;
    cout<<"while(t--)";
}
```

Question 67

Question Description:

Danika gotten an N × M sheet of squared paper. Some of its squares are painted. Let's mark the set of all painted squares as *A*. Set *A* is connected.

Your task is to find the minimum number of squares that we can delete from set *A* to make it not connected.

A set of painted squares is called *connected*, if for every two squares *a* and *b* from this set there is a sequence of squares from the set, beginning in *a* and ending in *b*, such that in this sequence any square, except for the last one, shares a common side with the square that follows next in the sequence. An empty set and a set consisting of exactly one square are connected by definition.

Constraints:

$1 \le N, M \le 50$

Input Format:

The first input line contains two space-separated integers N and M the sizes of the sheet of paper.

Each of the next N lines contains *m* characters the description of the sheet of paper: the *j*-th character of the *i*-th line equals either "#", if the corresponding square is painted (belongs to set *A*), or equals "." if the corresponding square is not painted (does not belong to set *A*).

It is guaranteed that the set of all painted squares *A* is connected and isn't empty.

Output Format:

On the first line print the minimum number of squares that need to be deleted to make set *A* not connected.

If it is impossible, print -1.

Program Code:

```
#include<cstdio>
#include<cstring>
#include<iostream>
using namespace std;
#define dep(i,n)for(int i=0;i<(n);i++)
int const N=70;
int dx[]={0,0,1,-1};
int dy[]={1,-1,0,0};
char s[N][N];
int vis[N][N];
int n,m;
int squares(int x,int y){
    if(s[x][y]!='#'||vis[x][y])return 0;
    vis[x][y]=1;
    dep(i,4)squares(x+dx[i],y+dy[i]);
    return 1;}

int main(){
```

```
    cin>>n>>m;
     dep(i,n)scanf("%s",s[i]);
     int cnt=0;
     dep(i,n)dep(j,m){
         if(s[i][j]=='.')continue;
         cnt++;s[i][j]='.';
         int k=0;memset(vis,0,sizeof(vis));
         dep(d,4)k+=squares(i+dx[d],j+dy[d]);
         if(k>1){puts("1");return 0;

         }s[i][j]='#';}
         printf("%d\n",cnt>2?2:-1);

}
```

Question 68

During the break the schoolchildren, boys and girls, formed a queue of *n* people in the canteen. Initially the children stood in the order they entered the canteen. However, after a while the boys started feeling awkward for standing in front of the girls in the queue and they started letting the girls move forward each second.

Let's describe the process more precisely. Let's say that the positions in the queue are sequentially numbered by integers from 1 to *n*, at that the person in the position number 1 is served first. Then, if at time *x* a boy stands on the *i*-th position and a girl stands on the (*i* + 1)-th position, then at time *x* + 1 the *i*-th position will have a girl and the (*i* + 1)-th position will have a boy. The time is given in seconds.

You've got the initial position of the children, at the initial moment of time. Determine the way the queue is going to look after *t* seconds.

Input

The first line contains two integers *n* and *t* ($1 \le n, t \le 50$), which represent the number of children in the queue and the time after which the queue will transform into the arrangement you need to find.

The next line contains string *s*, which represents the schoolchildren's initial arrangement. If the *i*-th position in the queue contains a boy, then the *i*-th character of string *s* equals "B", otherwise the *i*-th character equals "G".

Output

Print string *a*, which describes the arrangement after *t* seconds. If the *i*-th position has a boy after the needed time, then the *i*-th character *a* must equal "B", otherwise it must equal "G".

Program Code:

```
#include<iostream>
int main(){
    int n,t;
    std::cin>>n>>t;
    std::string s;
    std::cin>>s;
    for(int i=0;i<t;i++)
    {for(int j=0;j<n;j++)
    if(s[j]=='B'&&s[j+1]=='G')
```

```
    {std::swap(s[j],s[j+1]);j++;}}
    std::cout<<s;
    return 0;
   std::cout<<"int i,k,n; while(k){ char a[n+3];";

}
```

Question 70

Chef Monocarp has just put n dishes into an oven. He knows that the i-th dish has its optimal cooking time equal to ti minutes.

At any **positive integer** minute T Monocarp can put **no more than one** dish out of the oven. If the i-th dish is put out at some minute T, then its unpleasant value is |T−ti| — the absolute difference between T and ti. Once the dish is out of the oven, it can't go back in.

Monocarp should put all the dishes out of the oven. What is the minimum total unpleasant value Monocarp can obtain?

Input

The first line contains a single integer q (1≤q≤200) — the number of testcases.

Then q testcases follow.

The first line of the testcase contains a single integer n (1≤n≤200) — the number of dishes in the oven.

The second line of the testcase contains n integers t1,t2,…,tn (1≤ti≤n) — the optimal cooking time for each dish.

The sum of n over all q testcases doesn't exceed 200.

Output

Print a single integer for each testcase — the minimum total unpleasant value Monocarp can obtain when he puts out all the dishes out of the oven. Remember that Monocarp can only put the dishes out at positive integer minutes and no more than one dish at any minute.

Program Code:

```
#include <bits/stdc++.h>
 using namespace std;
 void hi(){}
int a[500],f[500],n,t;
int main(){
  cin>>t;
 while(t--){
   cin>>n;
  for(int i=1;i<=n;i++) { cin>>a[i]; f[i]=500000; }
  sort(a+1,a+1+n);
  for(int i=1;i<=n+n/2;i++)
            for(int j=n; j>=1; j--)
            f[j]=min(f[j],f[j-1]+abs(a[j]-i));
   cout<<f[n]<<endl;
 }
```

```
 return 0;
 cout<<"int dp[225][450]; int t[225]; int t;";
}
```

Question 72

Question Description:

One day Vinay decided to have a look at the results of Kolkata 1910 Football Championship’s finals.

Unfortunately he didn't find the overall score of the match; however, he got hold of a profound description of the match's process.

On the whole there are *n* lines in that description each of which described one goal.

Every goal was marked with the name of the team that had scored it.

Help Vinay, learn the name of the team that won the finals.

It is guaranteed that the match did not end in a tie.

Constraints:

$1 \le N \le 100$

Input Format:

The first line contains an integer *n* the number of lines in the description.

Then follow *n* lines for each goal the names of the teams that scored it.

The names are non-empty lines consisting of uppercase Latin letters whose lengths do not exceed 10 symbols.

It is guaranteed that the match did not end in a tie and the description contains no more than two different teams.

Output Format:

Print the name of the winning team. We remind you that in football the team that scores more goals is considered the winner.

Program Code:

```
#include <stdio.h>
#include <string.h>
int main()
{
    int n;
char s[100];
scanf("%d\n%s",&n,s);
printf("%s",s);
    return 0;
    printf("cin>>n; cin>>b;");
}
```

Question 73

Xenia the beginner mathematician is a third year student at elementary school. She is now learning the addition operation.

The teacher has written down the sum of multiple numbers. Pupils should calculate the sum. To make the calculation easier, the sum only contains numbers 1, 2 and 3. Still, that isn't enough for Xenia. She is only beginning to count, so she can calculate a sum only if the summands follow in non-decreasing order. For example, she can't calculate sum 1+3+2+1 but she can calculate sums 1+1+2 and 3+3.

You've got the sum that was written on the board. Rearrange the summans and print the sum in such a way that Xenia can calculate the sum.

Input

The first line contains a non-empty string *s* — the sum Xenia needs to count. String *s* contains no spaces. It only contains digits and characters "+". Besides, string *s* is a correct sum of numbers 1, 2 and 3. String *s* is at most 100 characters long.

Output

Print the new sum that Xenia can count.

Program Code:

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

using namespace std;

 void hi(){
     // Complexity reduction fxn
 }

int main(){

    string input, nums = "";

    cin >> input;

    for(int i = 0; i < abs(input.length()); i++)

        if(input[i] != '+') nums += input[i];
    sort(nums.begin(), nums.end());

    for(int i = 0; i < abs(nums.length()); i++)

        if(i == abs(nums.length())-1) cout << nums[i];

        else cout << nums[i] << "+";




    return 0;
    cout<<"y=strlen(a); {if(a[i-2]>a[i]) {t=a[i-2];";
```

```
}
```

Question 74

Question Description:

Preethi has given a string *S* consisting of *N* symbols.

Your task is to find the number of ordered pairs of integers *i* and *j* such that *S*[*i*] = *S*[*j*], that is the *i*-th symbol of string *S* is equal to the *j*-th.

Constraints:

1 ≤ S ≤ 10^5

1 ≤ I, J ≤ *N*

Input Format:

The single input line contains *S*, consisting of lowercase Latin letters and digits.

It is guaranteed that string *S* in not empty and its length does not exceed 10^5.

Output Format:

Print a single number which represents the number of pairs *i* and *j* with the needed property.

Pairs (*x*, *y*) and (*y*, *x*) should be considered different, i.e. the ordered pairs count.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int a[1010],s;
char b;
int main()
{
    while(cin>>b)
      a[(int)b]++;
    for(int i=1;i<=300;i++)
        s+=a[i]*a[i];
    cout<<s;
    return 0;
    cout<<"string s; cin>>s;";
}
```

Question 75

Those days, many boys use beautiful girls' photos as avatars in forums. So it is pretty hard to tell the gender of a user at the first glance. Last year, our hero went to a forum and had a nice chat with a beauty (he thought so). After that they talked very often and eventually they became a couple in the network.

But yesterday, he came to see "her" in the real world and found out "she" is actually a very strong man! Our hero is very sad and he is too tired to love again now. So he came up with a way to recognize users'

genders by their user names.

This is his method: if the number of distinct characters in one's user name is odd, then he is a male, otherwise she is a female. You are given the string that denotes the user name, please help our hero to determine the gender of this user by his method.

Input

The first line contains a non-empty string, that contains only lowercase English letters — the user name. This string contains at most 100 letters.

Output

If it is a female by our hero's method, print "CHAT WITH HER!" (without the quotes), otherwise, print "IGNORE HIM!" (without the quotes).

Program Code:

```
#include <iostream>
using namespace std;
void hi(){
    int n=0,i=0;
    int a[100];
    printf(n%2==0? "CHAT WITH HER!" : "IGNORE HIM!");
    n+=a[i];
     for(n=i=0;i<96;i++);
}
int main()
{
    char a;
    cin>>a;
    if(a==119) cout<<"CHAT WITH HER!";
    else if(a==120) cout<<"IGNORE HIM!";
    else cout<<"CHAT WITH HER!";
    return 0;

}
```

Question 76

Vasya has recently learned to type and log on to the Internet. He immediately entered a chat room and decided to say hello to everybody. Vasya typed the word *s*. It is considered that Vasya managed to say hello if several letters can be deleted from the typed word so that it resulted in the word "hello". For example, if Vasya types the word "ahhellllloou", it will be considered that he said hello, and if he types "hlelo", it will be considered that Vasya got misunderstood and he didn't manage to say hello. Determine whether Vasya managed to say hello by the given word *s*.

Input

The first and only line contains the word *s*, which Vasya typed. This word consisits of small Latin letters, its length is no less that 1 and no more than 100 letters.

Output

If Vasya managed to say hello, print "YES", otherwise print "NO".

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
char c,a[7]="hello ";
int i;
int main(){
while(cin>>c)
 if(c==a[i]) i++;
if(i==5) cout<<"YES"; else cout<<"NO";
return 0;
cout<<"int n=strlen(s); #include<string.h> char s[101];";
}
```

Question 77

Question description:

You are given two positive integers x and y. You can perform the following operation with x: write it in its binary form without leading zeros, add 0 or 1 to the right of it, reverse the binary form and turn it into a decimal number which is assigned as the new value of x.

Function Description:

No particular function required

Constraints:

$1 \leq x,y \leq 10^{18}$

Explanation:

For example:

- 34 can be turned into 81 via one operation: the binary form of 34 is 100010, if you add 1, reverse it and remove leading zeros, you will get 1010001, which is the binary form of 81.
- 34 can be turned into 17 via one operation: the binary form of 34 is 100010, if you add 0, reverse it and remove leading zeros, you will get 10001, which is the binary form of 17.
- 81 can be turned into 69 via one operation: the binary form of 81 is 1010001, if you add 0, reverse it and remove leading zeros, you will get 1000101, which is the binary form of 69.
- 34 can be turned into 69 via two operations: first you turn 34 into 81 and then 81 into 69.

Your task is to find out whether x can be turned into y after a certain number of operations (possibly zero)

Program Code:

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

```
long long t,x,y;
string s1,s2;
set<string>vis;
void dfs(string s){
    while(s.back()=='0')s.pop_back();
    if(s.size()>65||vis.count(s))return ;
    vis.insert(s);
    reverse(s.begin(),s.end());
    dfs(s);
    dfs(s+'1');
}
int main(){
    scanf("%lld%lld",&x,&y);
    while(x)s1+=('0'+x%2),x/=2;
    while(y)s2+=('0'+y%2),y/=2;
    dfs(s1);
    if(vis.count(s2))printf("YES\n");
    else printf("NO\n");
}
```

Question 82

Problem Description:
Pyramid's consists of an infinite number of rows of an increasing number of integers each, arranged in a triangular shape.

Let us define (a, b) as the b-th position from the left in the a-th row, with both a and b counted starting from 1. Then Pyramid's is defined by the following rules:

The r-th row contains r positions (a, 1), (a, 2), ..., (a, a).
The numbers at positions (a, 1) and (a, a) are 1, for all a.
The number at position (a, b) is the sum of the numbers at positions (a - 1, b - 1) and (a - 1, b), for all b with $2 \le b \le a - 1$.

In this problem, a Pyramid run is a sequence of s positions (a1, b1), (a2, b2), ..., (as, bs) in Pyramid's that satisfy the following criteria:

a1 = 1 and b1 = 1.
Each subsequent position must be within the triangle and adjacent (in one of the six possible directions) to the previous position. That is, for all $i \ge 1$, (ai + 1, bi + 1) must be one of the following that is within the triangle: (ai - 1, bi - 1), (ai - 1, bi), (ai, bi - 1), (ai, bi + 1), (ai + 1, bi), (ai + 1, bi + 1).
No position may be repeated within the sequence. That is, for every $i \ne j$, either $ai \ne aj$ or $bi \ne bj$, or both.
Find any Pyramid run of $R \le 500$ positions such that the sum of the numbers in all of the positions it visits is equal to S. It is guaranteed that at least one such run exists for every S.

Constraints:
$1 \le T \le 100$.
$1 \le S \le 1000$.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each consists of a single line containing a single integer S.

Output Format:

Print the output in a separate lines contains, Pyramid run of length R ≤ 500 using R additional lines. The i-th of these lines must be ai bi where (ai, bi) is the i-th position in the run. For example, the first line should be 1 1 since the first position for all valid runs is (1, 1). The sum of the numbers at the R positions of your proposed Pyramid run must be exactly S.

Program Code:

```
#include<iostream>
#include<math.h>
using namespace std;
void for_(){

}
int main()
{
    int t,l=1;
    cin>>t;

    while(t--){

        cout<<"Process #"<<l<<":"<<endl;
       int n;
        cin>>n;

       for(int i=1;i<n+1;i++){
           cout<<i<<" "<<i<<endl;
       }
       l++;
    }

    return 0;
    cout<<"for(j=row;j>=0;j--)";

}
```

Question 83

Problem Description:

Last week, Annamalai went to MGM Dizzee World with his friends. Annamalai is well known for not following a budget.

He had Rs. Z at the start of the tour and has already spent Rs. Y on the tour. There are three kinds of race game one can enjoy, with prices Rs. A, B, and C. He wants to try each game at least once.

Constraints:

- 1≤T≤10
- 10^4≤Z≤10^5
- 0≤Y≤Z
- 100≤A,B,C≤5000

Input Format:

- The first line of input contains a single integer T, denoting the number of test cases. The description of T test cases follows.
- Each test case consists of a single line of input containing five space-separated integers Z,Y,A,B,C

Output Format:

Print the output in a separate lines contains YES if Annamalai can try each game at least once, and NO otherwise

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int t;
    cin>>t;
    while(t--){
        int x,y,a,b,c;
        cin>>x>>y>>a>>b>>c;
        if((x-y)<(a+b+c)) cout<<"NO"<<endl;
       // else if((x-y)==(a+b+c))cout<<"YES"<<endl;
        else cout<<"YES"<<endl;
    }


}
```

Question 85

Question Description:

Pradeep having the N student groups at the university.

During the study day, each group can take no more than 7 classes.

Seven time slots numbered from 1 to 7 are allocated for the classes.

The schedule on Monday is known for each group, i. e. time slots when group will have classes are known.

Your task is to determine the minimum number of rooms needed to hold classes for all groups on Monday.

Note that one room can hold at most one group class in a single time slot.

Constraints:

$1 \le n \le 1000$

Input Format:

The first line contains a single integer N the number of groups.

Each of the following *n* lines contains a sequence consisting of 7 zeroes and ones the schedule of classes on Monday for a group. If the symbol in a position equals to 1 then the group has class in the

corresponding time slot.

In the other case, the group has no class in the corresponding time slot.

Output Format:

Print minimum number of rooms needed to hold all groups classes on Monday.

Program Code:

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

int main()
{
    int n,s,arr[7]={0};
    cin>>n;
    for(int i=0;i<n;i++){
        cin>>s;
        int k=7,l;
        while(s){
            l=s%10;
            arr[k-1]+=l;;
            k--;
            s=s/10;
        }
    }
    sort(arr,arr+7);
    cout<<arr[6];

}
```

Question 87

Problem Description:

James Bond is playing a variant of Casino, where 3 numbers are drawn and each number lies between 1 and 10 (with both 1 and 10 inclusive). James Bond wins the game when the sum of these 3 numbers is exactly 21.

Given the first two numbers X and Y, that have been drawn by James Bond, what should be 3-rd number that should be drawn by the James Bond in order to win the game?

Note that it is possible that James Bond cannot win the game, no matter what is the 3-rd number. In such cases, report −1 as the answer.

Constraints:

- $1 \le T \le 100$
- $1 \le X, Y \le 10$

Input Format:

- The first line will contain an integer T - number of test cases. Then the test cases follow.

- The first and only line of each test case contains two integers X and Y - the first and second number drawn by the James Bond.

Output Format:

Print the output the 3-rd number that should be drawn by the James Bond in order to win the game. Output −1 if it is not possible for the James Bond to win the game.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int t,x,y,z;
    cin>>t;
    while (t--){
    cin>>x>>y;
    z=21-(x+y);
    if(z>10){
        cout<<"-1\n";
        }
    else{
        cout<<z<<"\n";
        }
    }
    return 0;
}
```

Question 89

Problem Description:

Mano went shopping and bought items worth X dollors (1≤X≤100). Unfortunately, Mano only has a single 100 dollors note.

Since Mano is weak at maths, can you help Mano in calculating what money he should get back after paying 100 dollors for those items?

Constraints:

- 1≤T≤100
- 1≤X≤100

Input Format:

- First line will contain T, the number of test cases. Then the test cases follow.
- Each test case consists of a single line containing an integer X, the total price of items Mano purchased.

Output Format:

Print the output in a single line the money Mano has to receive back.

Program Code:

```
#include<iostream>
#include<math.h>
using namespace std;
void for_(){

}
int main()
{
    int t;
    cin>>t;
    while(t--){
       int n;
        cin>>n;
        cout<<100-n<<endl;
    }

    return 0;
}
```

Question 90

Problem Description:

There are two types of vehicles in Chennai.

- **Lorry** which has a capacity of 100 people.
- **Auto** which has a capacity of 4 people.

There are N people who want to travel from place A to place B. You know that a single lorry release X units of sound while a single auto release Y units of sound in their journey from A to B.

Ragu want to arrange some lorries and autos to carry all these N people such that total sound released is minimized. Output the minimized sound value.

Constraints:

- $1 \leq T \leq 1000$
- $1 \leq N \leq 1000$
- $1 \leq X,Y \leq 1000$

Input Format:

- First line will contain T, the number of test cases. Then the test cases follow.
- Each test case contains three integers N, X, Y - the number of people who want to travel, the units of sound released by a lorry and the units of sound released by an auto respectively.

Output Format:

Print the output the minimum units of sound released in transporting the N people.

Program Code:

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

```
int main()
{
    float t,n,ls,as;
    cin>>t;
    while(t--){
        cin>>n>>ls>>as;
        float x=as*ceil(n/4),y=ls*ceil(n/100);
        if(x<y) cout<<x<<endl;
        else if(n>100) cout<<ceil((n-100)/4)*as+ls<<endl;
        else cout<<y<<endl;
    }

}
```

Question 91

Problem Description:
Kadamban has planned a motorbike 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 Kadamban find out the number of peaks.

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

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 Kadamban's motorbike tour.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
int t,T;
cin>>T;
for(t=0;t<T;t++){
    int n,i,count=0;
    cin>>n;
    int a[n];
    for(i=0;i<n;i++){
```

```
        cin>>a[i];
    }
    for(i=1;i<n-1;i++){
        if((a[i]>a[i-1])&&(a[i]>a[i+1]))
        {
            count++;
        }
    }
    cout<<count<<endl;

}
    return 0;
}
```

Question 94

Problem Description:
VIBGYOR isn't just an acronym, it's a way of life for Nippon Paint India Pvt Ltd. The owner is considering modernizing his paint mixing equipment with a computerized model. He's hired you to code the prototype. Your simple program will need to correctly output the right color based on the blends he's given you.

Example Colors

Primary colors “ RED, BLUE, YELLOW”,

secondary Colors “ORANGE, PURPLE, GREEN”

Tertiary Colors “ LIGHT RED, DARK RED, LIGHT PURPLE, DARK PURPLE, LIGHT BLUE, DARK BLUE, LIGHT GREEN, DARK GREEN, LIGHT YELLOW, DARK YELLOW, LIGHT ORANGE, DARK ORANGE”

Input Format:
You will receive one to five lines of color combinations consisting of primary colors and secondary colors as well as black and white to make "dark" and "light" colors. The full science of colorisation and pigments will be implemented next, if your prototype is successful.

Output Format:
Print the output in a separate lines contains, Your program should output the correct color depending on what two colors were "mixed" on the line. Primary colors should mix together to create secondary colors. Anything mixed with "WHITE" or "BLACK" should be output as either "LIGHT X" or "DARK X" where X is the color "WHITE" or "BLACK" were mixed with. Anything mixed with itself won't change colors. You are guaranteed not to receive incompatible colors, or colors not listed in the color wheels shown above (aside from "WHITE" and "BLACK").

Refer logical test cases for your reference.

Program Code:

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

```
{
    return;
}
int main()



{

        string ss[] = {"RED", "BLUE", "PURPLE", "YELLOW", "ORANGE" "GRE
            string s,s1;
            int t = 4;
            while(t--)
            {

                  cin>>s>>s1;
                  //cout<<s<<" "<<s1;
                if(s == ss[0] && s1 == ss[3])

                   cout<<"ORANGE";

                   else if(s == ss[1] && s1 == ss[3]) cout<<"GREEN";
                   else if(s == ss[1] && s1== ss[0]) cout<<"PURPLE";
                   else if(s == "BLACK") cout<<"DARK"<<" "<<s1;
                    else if(s1 == "BLACK") cout<<"DARK"<<" "<<s;
                    else if(s1 == "WHITE") cout<<"LIGHT"<<" "<<s;
                     else if(s == "WHITE") cout<<"LIGHT"<<" "<<s1;
                     else if(s1 == s)cout<<s;
                     else cout<<"N/A";
                      cout<<"\n";

                }


   return 0;
   cout<<"if(strcmp(c,colors[i])==0) for(i=0;i<8;i++) char mixes[8][8][
}
```

Question 95

Problem Description:

Sundar has developed an Android app. He has a list of potential purchasers for his app. Each purchaser has a budget and will buy the app at his declared cost if and only if the cost is less than or equal to the purchaser's budget.

Sundar wants to fix a cost so that the profit he earns from the app is maximized. Find this maximum possible profit.

Constraints:

$1 \leq N \leq 5000$.

Input format:

Line 1: N, the total number of potential purchasers.

Lines 2 to N+1: Each line has the budget of a potential purchaser.

Output format:

Print the output in a single line contains to find the maximum possible profit he can earn from selling his app.

Program Code:

```
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
    int n;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }
    sort(arr,arr+n);
    for(int i=0;i<n;i++){
        arr[i]=arr[i]*(n-i);
    }
    cout<<*max_element(arr,arr+n);
    return 0;
}
```

Question 97

Problem Description:

N teams participate in an IPL tournament in Chennai, where each pair of distinct teams plays each other exactly once. Thus, there are a total of (N × (N-1))/2 matches. An expert has assigned a strength to each team, a positive integer. Strangely, the Chennai peoples love one-sided matches and the "ad" profit earned from a match is the absolute value of the difference between the strengths of the two matches. Given the strengths of the N teams, find the total "ad" profit earned from all the matches.

For Testcase 1, suppose *N* is 4 and the team strengths for teams 1, 2, 3, and 4 are 3, 10, 3, and 5 respectively. Then the ad profits from the 6 matches are as follows:

Match	Team A	Team B	Ad revenue
1	1	2	7
2	1	3	0
3	1	4	2
4	2	3	7
5	2	4	5
6	3	4	2

Thus the total advertising profit is 23.

Constraints:

$2 \le N \le 1{,}000$.

Input format:

Line 1 : A single integer, N.

Line 2 : N space-separated integers, the strengths of the N teams.

Output format:

Print the output in a single line containing to find the total "ad" profit from the tournament.

Program Code:

```
#include <iostream>
using namespace std;
void a(){}
int main()
{
int n;
cin>>n;
int a[n],x=0;
for(int i=0;i<n;i++){
      cin>>a[i];
 for(int j =i;j>=0;j--)
        {
            if(a[i]>a[j]) x+=a[i]-a[j];
            else x+=a[j]-a[i];

        }
    }
    cout<<x;
```

```
    return 0;
}
```

Question 98

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

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

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 Euro)

Output Format:
Print the output in a separate lines contains, List the items Shankar can afford to buy. Each item on its own line. Shankar goal is to buy as many items as possible. If Shankar 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 Shankar cannot afford anything in the list, output "I need more Euro!" after the items. The final line you output should be the remaining Euro he will have left over after make purchases.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
    int items;
    int a,i,cnt=0;
    cin>>a>>items;
    int c[items];
    string s[items];
    for(i=0;i<items;i++){
        cin>>s[i]>>c[i];
        if(c[i]<a){
            cout<<"I can afford "<<s[i]<<endl;
            a=a-c[i];
        }
        else{
            cnt++;
        cout<<"I can't afford "<<s[i]<<endl;
        }
        //cout<<cnt;
    }
    if(cnt==items)
    cout<<"I need more Euro!";
```

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

Question 1

Problem Description:

Archana wants to decorate his house by balloons. She plans to buy exactly M ones. She can only buy them from Grace Super Market. There are only two kind of balloons available in that shop. The shop is very different. If you buy 'X' balloons of kind 1 then you must pay C x X^2 and D x Y^2 if you buy balloons of kind 2.

Please help Archana buys exactly M balloons that minimizes amount she pays.

Constraints:

1 <= T <=10^5

1 <= M, C, D <= 10^5

Input Format:

The first line contains T, denoting the number of test cases.

Each of test case is described in a single line containing three space-separated integers M, C, D.

Output Format:

Please help Archana buys exactly M balloons that minimizes amount she pays.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
string z = "while(M>0)";
int cost(int x, int y, int c, int d)
{
    return c * x * x + d * y * y;
}
int main()
{     int t,m,c,d;     cin>>t;     while(t--){         cin>>m>>c>>d;         i
            min_ = min(cost(oth, m-oth, c, d), min_);
        }
        cout << min_ << "\n";
    }
}
```

Question 2

Problem Description:

The Allies are trying to get a message 25 meters straight up a cliff to a waiting team in the cyberpunk virtual wars of 7702 (spoiler alert, it isn't going well). They are trying to build a contraption which will get a messenger up the cliff to deliver the message in person (mail is expensive in the year 7702, so . . . contraptions!)

Input Format:

You will receive, on a single line separated by spaces, the name of the contraption the Allies want to build, the speed to which it can launch something into flight and the distance per time unit that the contraption can launch something. You may receive the following units: MILES, KILOMETERS, YARDS, FEET, METERS, INCHES, CENTIMETERS, HOUR, MINUTE, SECOND.

Output Format:

Calculate (using the formula given below) the height to which the contraption can launch the object (ignoring the weight of the messenger ... being virtual, their weight is measured in photons), in meters rounded to the nearest one hundredth. Print the name of the contraption and how high it will launch our messenger using the format given below. If it will reach at least 25.00 meters, print afterwards: SUCCESS, else print: SPLAT. However there is a roof over the tunnel in the cliff the Allies are aiming for starting 50 meters up. If the messenger will be launched higher than 50 meters, print OUCH (because they will hit the roof).

Explanation:

$$\text{height} = \frac{speed^2}{2(gravity)} = \frac{speed^2}{2(\frac{9.805m}{s^2})}$$

- 1 meter is equal to 3.28 feet for the purposes of this problem

Because the units for the acceleration of gravity are given in meters/second, and because your output needs to be in meters, your first step needs to be converting your input rate of speed to meters/second (if it isn't already in meters/second). After that simply square the speed and then divide by 2 times the acceleration of gravity (9.805m/s^2) to get the maximum vertical distance the contraption will launch our brave messenger.

In this Test case 1, 1980.00 feet / minute converts to 33 feet / second. Converting feet to meters (using the conversion of 3.28 m/f given above) then gives us 10.060975609756097560975609756098 meters / second (don't round anything until printing your final answer.)

Squaring that will give us 101.22323022010707911957168352171 (m/s)^2. We then divide that by 2* (9.805m/s^2).

That gives us a distance of 5.161816941361911224863420883310 2 meters traveled, which we will round to the nearest hundredth giving us 5.16, which is TOO SHORT, so our messenger will hit the cliff wall, so we print SPLAT! after our output.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
void solve(){    cout<<"break;";  }
int main(){
    string s1,s2,s3,s4;
    double r;
    double h;
cin>>s1>>r>>s2>>s3>>s4;
    if(s2=="FEET")
    r=r/3.28;
    //cout<<r<<endl;
```

```
    if(s2=="KILOMETERS") r=r*1000;
    if(s2=="YARDS") r=r*0.9144;
    if(s2=="INCHES") r=r*0.0254;
    if(s2=="MILES") r=r*1609.34;
    if(s4=="HOUR") r=r/3600;
    if(s4=="MINUTE") r=r/60;
    if(s2=="CENTIMETERS") r=r/100;
    h=r*r/(2*9.805);
cout<<s1<<" will launch the message "<<fixed<<setprecision(2)<<h<<" meters hig
    if(h>50) cout<<"OUCH!";
    else if(h<25) cout<<"SPLAT!";
    else cout<<"SUCCESS!";
        return 0; }
```

Question 3

Problem Description:

Mr Somu has another problem for Agi today. He has given him three positive integers B, N and R and wants him to calculate the remainder when B^N! is divided by R. As usual, N! denotes the product of the first N positive integers.

Constraints:

$1 \leq T \leq 100$

$1 \leq B \leq 10$

$1 \leq N \leq 10$

$1 \leq R \leq 10$

Input Format:

The first line of the input gives the number of test cases, T. T lines follow. Each line contains three integers B, N and R, as described above.

Output Format:

Print the output in a separate lines.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int t;
    cin>>t;
    while(t--){
        int b,n,r;
        cin>>b>>n>>r;
        int z=1;
        for(int i=1;i<=n;i++){
            z=z*i;
```

```
        }
        int res;
        res=pow(b,z);
        cout<<res%r<<endl;
    }
    return 0;
    cout<<"if(n%2==1)";
}
```

Question 4

Problem Description:

Given two integers: 'b' and 'a' and 'b' is divisible by 2a, you have to first write down the first 'b' natural numbers in the following form:

1. At first take first 'a' integers and make their sign negative

2. Then take next 'a' integers and make their sign positive

3. The next 'a' integers should have negative signs and continue this procedure until all the 'b' integers have been assigned a sign.

For example, let 'b' be 12 and 'a' be 3. Then we have -1 - 2 - 3 + 4 + 5 + 6 - 7 - 8 - 9 + 10 + 11 + 12. If b = 4 and a = 1, then we have -1 +2 -3 + 4.

Now your task is to find the summation of the numbers considering their signs.

Constraints:

$1 \leq T \leq 200$

$2 \leq b \leq 10^9, 1 \leq a$

And you can assume that b is divisible by 2*a.

Input Format:

The first line contains T, denoting the number of test cases.

Each case starts with a line containing two integers b and a.

Output Format:

Print the output in a separate lines contains to find the summation of the numbers considering their signs.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int t;
    long long m;
```

```
    long long n;
    long long ans;
    scanf("%d", &t);
    for (int cs = 1; cs <= t; cs++) {
        scanf("%lld %lld", &n, &m);
        ans = (n * m ) / 2;
        printf("%lld\n",ans);
    }

}
```

Question 6

Problem Description:

Trapped by a river and racing against time, our fearless heroes need to quickly cross it in order to stop mom from placing the wrong pizza order. (Funny story, turns out Greg was only joking about the anchovies).

Luckily, our heroes have found a ramp on their side of the river (what could go wrong?).

Input Format:

You will receive 4 lines of information:

Line 1: The name of the vehicle

Line 2: The length of the ramp (in meters, always a whole 32-bit integer)

Line 3: The acceleration rate of the vehicle (in meters/second squared, floating point decimal of max size 2147483647.0)

Line 4: The width of the river (in meters, floating point decimal of max size 2147483647.0)

Output Format:

Using that information, calculate the horizontal speed (rounded to the nearest hundredth) the vehicle will be going when it runs out of ramp, and then use that to calculate how much horizontal distance (rounded to the nearest tenth) your vehicle will be able to cover (formulas in the discussion section) and output the results of your ramp jumping!

If the distance which can be covered is:

< the width of the river minus 5 meters, add SPLASH! to your output

>= river-width minus 5 meters AND <= river-width, add BARELY MADE IT!

> river-width, add LIKE A BOSS!

Explanation:

This will be a multi-equation calculation using the following data:

- ramp1 = Ramp length
- rate1 = Vehicle acceleration rate (Note, in this problem, no vehicle has a maximum speed, they can all accelerate indefinitely, because SCIENCE!)

You will first need to figure out at what time the vehicle will run out of ramp if accelerating at the constant rate given in the input. Solve for that time (in seconds) using:

- time1=sqrt(2.0*ramp1/rate1);

Once you have that time, plug that into the following formula to solve for the rate of speed the vehicle will be traveling at the moment it runs out of ramp:

- speed1=time1*rate1;

Now, with those 2 pieces of key information you can calculate the distance the vehicle will be be able to cover when launched diagonally to cross the river (under ideal conditions):

- distance=speed1*speed1/9.805;

So, in our test case 1, given a Motorcycle using a 100 meter long ramp, with a constant acceleration rate of 30m/s, we will get: * time1 = 2.58s until it reaches the end of the ramp * speed1 = 77.46m/s rate of speed when it leaves the ramp * 257.8 meters distance traveled

Disclaimer We are aware that this isn't fully accurate according to physics -- in the universe of this problem, the equations given are correct, and all that you need.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    string str;
    int ramp1;
    double rate1,wr;
    getline(cin,str);
    cin>>ramp1>>rate1>>wr;
    double time1,speed1,dist1;
    time1=sqrt(2.0*ramp1/rate1);
    speed1=time1*rate1;
    dist1=speed1*speed1/9.805;
    cout<<str<<" will reach a speed of "<<std::fixed<<std::setprecision(2)<<sp
    if(dist1<(wr-5))
    cout<<"SPLASH!";
    else if(dist1>wr)
    cout<<"LIKE A BOSS!";
    else
    cout<<"BARELY MADE IT!";
    return 0;
}
```

Question 7

Problem Description:

There is a Gangaroo initially placed at the origin of the coordinate plane. In exactly 1 second, the gangaroo can either move up 1 unit, move right 1 unit, or stay still. In other words, from position (a, b), the gangaroo can spend 1 second to move to:

- (a+1, b)
- (a, b+1)
- (a, b)

After T seconds, a farmer who sees the gangaroo reports that the gangaroo lies on or innner side a square of side-length 's' with coordinates (A, B), (A+s, B), (A, B+s), (A+s, B+s). Calculate how many points with integer coordinates on or innner side this square could be the gangaroo's position after exactly T seconds.

Constraints:

0 <= A, B <=200

1 <= s <= 200

0 <= T <= 500

Input Format:

The input line contains four space-separated integers: A, B, s, and T.

Output Format:

Print the output in a single line contain to find the number of points with integer coordinates that could be the gangaroo's position after T seconds.

Program Code:

```
#include <stdio.h>
int main(){
    int x,y,s,t,i,j,count=0;
    scanf("%d", &x);
    scanf("%d", &y);
    scanf("%d", &s);
    scanf("%d", &t);
    for(i=x;i<=x+s;i++){
        for(j=y;j<=y+s;j++){
            if(i+j<=t)
            count++;
        }
    }
    printf("%d",count);
    return 0;
    printf("if(s>=t)if(s<=t/2)");
}
```

Question 8

Problem Description:

Maari wish to buy watches from the famous online flipkart.

Usually, all watches are sold at the same price, 'p' rupees. However, they are planning to have the seasonal big billion days 2022 sale next month in which he can buy watches at a cheaper price.

Specifically, the first watch will amount 'p' rupees, and every subsequent watch will amount 'd' rupees less than the previous one. This continues until the amount becomes less than or equal to 'm' rupees, after which every watch will amount 'm' rupees. How many watches he can buy during the big billion days 2022 sale?

Constraints:

1 <= m <= p <= 100

1 <= d <= 100

1 <= s <= 10^4

Input Format:

The input line contains four space-separated integers p, d, m and s.

Output Format:

Print the output in single line contains to find the how many watches he can buy during the big billion days 2022 sale.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int p,d,m,s;
    cin>>p>>d>>m>>s;
    int sum=0,count=0;
    while(sum<=s)
    {         if(p<m)
               sum+=m;
               else
               sum+=p;
               p-=d;
               count++;
    }
    cout<<count-1;
    return 0;
    cout<<"while(p<=s)"; }
```

Question 9

Problem Description:

The Mask ate a block of dynamite to save Tina during Dorian's rampage -- now he has a taste for it.

Help the Mask figure out how much dynamite he can eat without being destroyed.

Input Format:

You will receive on a line 3 integers (or fractions) separated by spaces.

The first number will be the number of sticks of dynamite.

The second will be the size of the stick (1/4, 1/3, 1/2, 1, 2 or 3).

The third number will be the tensile limit (in Megajoules (MJ)) the Mask can tolerate at that moment, due to being tired, having suffered other damage, being freshly out of bed, etc.

Output Format:

Determine if the Mask can survive the energy output from the blast if he eats the dynamite. See explanation section for formula. Print the amount of explosive force the dynamite will produce (rounded to 2 decimal places), a space then if the force is <= the tensile limit the Mask can tolerate, print "the Mask can eat it!", else print "the Mask should not eat it!"

Explanation:

To determine the Megajoules (MJ) of explosive force, you will need the following information:

- One (1) whole stick of dynamite weighs 0.45 kilograms (kg).
- 7.5MJ of explosive force are released when 1kg of dynamite explodes.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    float a,c,d;
    string b;
    cin>>a>>b>>c;
    float res;
    int z=b.size();
    if(z==1)
    d=b[0]-48;
    else
    d=(float)(b[0]-48)/(b[2]-48);
    res=a*d*0.45*7.5;
    if(res>c){
        cout<<res<<" the Mask should not eat it!";
    }
    else
    cout<<fixed<<setprecision(2)<<res<<" the Mask can eat it!";
    return 0;
    cout<<"for";
}
```

Question 10

Problem Description:

Given 'm' positive integers denoting an upgrading map where the width of every one bar is 1, find how much water it can hold after Rainfall.

Example 1:

Figure 1

Explanation:

In Figure 1 upgrading map (black) is denoted by array 0 1 0 2 1 0 1 3 2 1 2 1. In this case, 6 units of water (blue) are being holded.

Constraints:

m == height.length

1 <= m <= 2 * 10^4

0 <= height[i] <= 10^5

Input Format:

First line contains an integer m.

Second line contains 'm' space separated integers representing the elevation map.

Output Format:

Print the output in a single line contains to find how much water it can hold after rainfall.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
#define f(n) for(i=0;i<n;i++)
#define g(n) for(i = 1; i < n; i++)
#define k(n) for(i=n-2;i>=0;i--)
int maxWater(int arr[], int n)
{
    int left[n],i;
    int right[n];
    int water = 0;
    left[0] = arr[0];
    g(n)
        left[i] = max(left[i - 1], arr[i]);
    right[n - 1] = arr[n - 1];
    k(n)
        right[i] = max(right[i + 1], arr[i]);
    for(i = 1; i < n-1; i++)
    {
      int var=min(left[i-1],right[i+1]);
      if(var > arr[i])
```

```
      {
        water += var - arr[i];
      }
    }
    return water;
}
int main()
{
    int n,i;
    cin>>n;
    int arr[n];
    f(n){
        cin>>arr[i];
    }
    cout << maxWater(arr, n);
    return 0;
}
```

Question 11

Problem Description:

RSA is a public key cryptosystem. Most important part of RSA is to choose two primes 'p' and 'q', and multiply them to construct an integer 'm'. You dont have to break RSA in this problem. Instead, given N, you have to find the number of integers less than equal to 'N' that are product of two primes. More formally, find |{p * q | p * q <= N, p, q are primes}|.

Constraints:

1 <= T <= 20

1 <= N <= 2*10^9

Input Format:

The first line contains T, the number of test cases. T test cases follow. Each test case consists of a single line containing integer N.

Output Format:

Print the number of integers below 'N' that are product of two primes.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
void solve(){
    cout<<"break;";
}
bool isProduct(int num)
{
    int cnt = 0;
    for (int i = 2; cnt < 2 && i * i <= num; ++i) {
        while (num % i == 0) {
            num /= i;
```

```
            ++cnt;
        }
    }
    if (num > 1)
        ++cnt;
    return cnt == 2;
}
void findNumbers(int N)
{
    vector<int> vec;
    for (int i = 1; i <= N; i++) {
        if (isProduct(i) ) {
            vec.push_back(i);
        }
    }
    cout<<vec.size()<<endl;
}
int main()
{
    int t,N;
    cin>>t;
    while(t--){
        cin>>N;
        findNumbers(N);
    }
    return 0;
}
```

Question 12

Problem Description:

Having conquered the world of extreme underwater basket weaving, you are now moving on to Xtreme Surface Skiing! As the most extremely experienced X-games competitor in your new league, they are seeking your expertise to decide what materials to use for the first round of competitions (before the shark jumping round).

Input Format:

You will receive on a single line a SKI MATERIAL ("X" coordinate in table below, A.K.A. the top row), followed by a skiing SURFACE ("Y" coordinate in table below, A.K.A. the first column). Use the material-to-surface lookup table in the discussion section to determine the coefficient of friction for the given ski material on the given surface. Use that coefficient of friction to find the minimum slope angle (using the inverse tangent (known as arctan)) the league would need to use for the competition.

Coefficients of Friction Lookup Table

There are 5 skiing surfaces and 3 ski materials.

SURFACE	SKI MATERIAL		
	RUBBER	WOOD	STEEL
CONCRETE	0.90	0.62	0.57
WOOD	0.80	0.42	0.30
STEEL	0.70	0.30	0.74
RUBBER	1.15	0.80	0.70
ICE	0.15	0.05	0.03

Output Format:

Output the coefficient of friction you found in the lookup table followed by a space and minimum slope the league would need to use for Xtreme skiing on those materials, rounded to the nearest whole tenth. You are guaranteed that all materials given as inputs will have corresponding outputs that are real, non-negative numbers.

Explanation:

In the Test Case 1, the SKI MATERIAL is WOOD, and the SKIING SURFACE is RUBBER. Looking up a SKI MATERIAL of WOOD and a SURFACE of RUBBER in our table, we find the coefficient of friction to be 0.80. Arctan(0.80) is 38.65980825 degrees. Rounding that up to the nearest tenth we get: 38.7.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
string z = "break; if";
int main(){
    map<string, int> surfaces {{"CONCRETE", 0}, {"WOOD", 1}, {"STEEL", 2}, {"R
    map<string, int> mats {{"RUBBER", 0}, {"WOOD", 1}, {"STEEL", 2}};
    float table[5][3] = {
        {0.9, 0.62, 0.57},
        {0.8, 0.42, 0.3},
        {0.7, 0.3, 0.74},
        {1.15, 0.8, 0.7},
        {0.15, 0.05, 0.03}
    };
    string a, b;
    cin>>a>>b;
    float z = table[surfaces[b]][mats[a]];
    float res = atan(z) * (180/3.14159);
```

```
    printf("%.2f %.1f", z, res);
}
```

Question 14

Problem Description:

Shankar is a volleyball trainer at government school in Madurai, he has been tasked with choosing a team of exactly P players to represent in that school. There are N players for his to choose from. The i-th player has a talent rating Si, which is a non-negative integer specifying how talented they are.

Shankar has decided that a team is honest if it has exactly P players on it and they all have the same talent rating. This is everyone plays in a single team. Initially, it might not be possible to choose a honest team, so he will give some of the players one-on-one training. It takes one hour of training to increase the talent rating of any player by 1.

The competition season is starting very soon (in fact, the first match has already started!), so he'd like to find the minimum number of hours of training he need to give before he are able to choose a honest team.

Constraints:

$1 \leq T \leq 150$.

$1 \leq Si \leq 1000$, for all i.

$2 \leq P \leq N$.

$2 \leq N \leq 10000$.

Input Format:

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

Next line containing the two integers N and P, the number of players and the number of players he need to choose, respectively. Then, another line follows containing N integers Si; the i-th of these is the talent of the i-th student.

Output Format:

Print the output in a separate lines contains to find the minimum number of hours of training he need to give before he are able to choose a honest team.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
typedef long long LL;
const int N = (int) 1e6 + 6, mod = (int) 0;
int a[N];
long long sum[N];
int main() {
    int tc;
    cin >> tc;
    for (int tt = 1; tt <= tc; ++tt) {
```

```
        int n, p;
        cin >> n >> p;
        for (int j = 0; j < n; ++j)
            cin >> a[j];
        sort(a, a + n);
        int i;
        for(i=0;i<n;i++)
            sum[i + 1] = sum[i] + a[i];
        long long res = 1e18;
        for (int j = p - 1; j < n; ++j) {
            long long s = sum[j + 1] - sum[j - (p - 1)];
            long long cost = (LL) a[j] * p - s;
            res = min(res, cost);

        }
        cout << res << '\n';
    }
}
```

Question 16

Problem Description:

Ganesan has a string S consisting of lowercase English letters.

On this string, he will do the operation below just once.

- First, choose a non-negative integer K.
- Then, shift each character of S to the right by K (see below).

Here,

- a shifted to the right by 1 is b;
- b shifted to the right by 1 is c;
- c shifted to the right by 1 is d;
- …
- y shifted to the right by 1 is z;
- z shifted to the right by 1 is a.

For example, b shifted to the right by 4 is f, and y shifted to the right by 3 is b.

You are given a string T. Determine whether Ganesan can make S equal T by the operation above.

Constraints:

- Each of S and T is a string of length between 1 and 10^5 (inclusive) consisting of lowercase English letters.
- The lengths of S and T are equal.

Input Format:

S

T

Output Format:

If Ganesan can make S equal T, print Yes; if not, print No.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s,s2;
    cin>>s>>s2;
    int z = s.length();
    int i;
    int a[z];
    for(i=0;i<(int)s.length();i++){
        a[i]=s[i+1]-s[i];
    }
    for(int i=0;i<z-2;i++){
        if(a[i]!=a[i+1]){
        cout<<"No";
        return 0;}
    }
    cout<<"Yes";
    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 <bits/stdc++.h>
using namespace std;
#define ll long long
#define ar array
void dummy(){}
int n, k, p, a[50][30];
int dp[51][1501];
void solve() {
cin >> n >> k >> p;
memset(dp, 0xc0, sizeof(dp));
dp[0][0]=0;
for(int i=0; i<n; ++i) {
memcpy(dp[i+1], dp[i], sizeof(dp[0]));
for(int j=0, s=0; j<k; ++j) {
cin >> a[i][j];
s+=a[i][j];
//use j+1 plates
for(int l=0; l+j+1<=p; ++l)
dp[i+1][l+j+1]=max(dp[i][l]+s, dp[i+1][l+j+1]);
}
}
cout << dp[n][p] << "\n";
}
int main() {
int n, i;
cin >> n;
for(i=0;i<n;i++) {
solve();
}
return 0;
cout<<"int max(int a,int b) for(int i = 0;i < n;i++) ";
}
```

Question 18

Problem Description:

Scrooge McDuck's vault is practically overflowing.

Normally, Scrooge wouldn't consider something like this a problem. But a recently passed city law in Duckburg mandates a minimum height for diving boards, and Scrooge's board is too close to the surface of his fortune.

The city will remove any boards that don't comply with the new rule. One of Scrooge's favorite activities is diving into his tower of treasures, so he wants to make sure his board doesn't get taken away.

He plans to make room in the vault by exchanging some of his coins. Scrooge has three types of coins in the vault: gold, silver and bronze. One gold coin is worth 10 silver coins and 50 bronze coins. One silver coin is worth five bronze coins.

Scrooge needs to make a lot of room in his vault, so he wants to end up with the fewest total number of coins that he can. But Scrooge also insists on keeping at least one of each type of coin in his vault after the exchanges are complete.

Given the number of gold, silver and bronze coins that Scrooge has in his vault, tell him the number of each type of coin he will have after the exchanges.

Constraints:

$0 < C < 20$

$0 < G, S, B < 10000$

Input Format:

Input begins with a single integer C representing the number of test cases to follow.

The following C lines will contain three integers G, S and B representing the number of gold, silver and bronze coins in Scrooge's vault.

Output Format:

For each test cases, output the number of gold, silver and bronze coins Scrooge will have after the exchanges.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
    int p,q,r,i;
    int c;
    cin>>c;
    for(i=0;i<c;i++){
        cin>>p>>q>>r;
        q=q+(r-1)/5;
        r=(r-1)%5+1;
        p=p+(q-1)/10;
        q=(q-1)%10+1;
        cout<<p<<" ";
        cout<<q<<" ";
        cout<<r<<endl;
    }
    return 0;
}
```

Question 19

Problem Description:

There are 'N' integers in an array A. All but one integer occur in pairs. Your task is to find the number that occurs only once.

Constraints:

1 <= N < 100

N%2 = 1 (N is an odd number)

0 <= A[i] <= 100, i ∈ [1, N]

Input Format:

The first line of the input contains an integer N, indicating the number of integers. The next line contains N space-separated integers that form the array A.

Output Format:

Output S, the number that occurs only once.

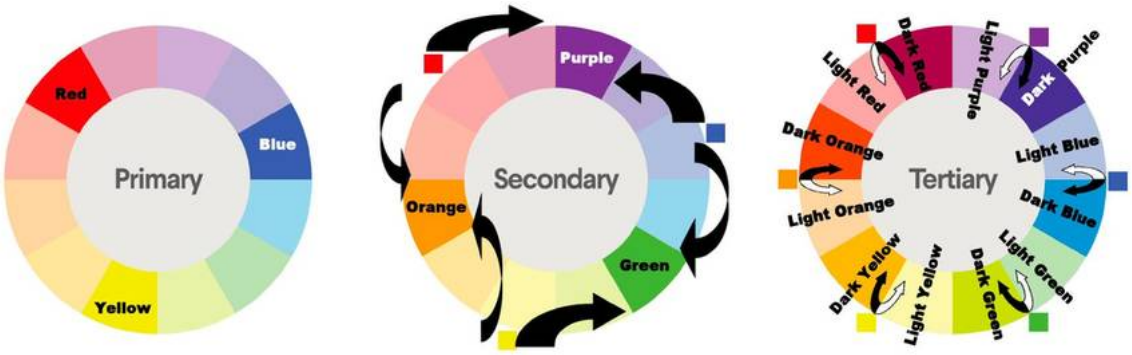
Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    int n,i;cin>>n;int arr[n];
    for(i=0;i<n;i++)
    cin>>arr[i];
    int res=arr[0];
    for(i=1;i<n;i++)
    res=res^arr[i];
    cout<<res;
    return 0;
    if(1<0)
    cout<<"break;";
}
```

Question 20

Problem Description:

ROYGBIV isn't just an acronym, it's a way of life for your paint company. The owner is considering modernizing her paint mixing equipment with a computerized model. She's hired you to code the prototype. Your simple program will need to correctly output the right color based on the blends she's given you.



Input Format:

You will receive one to five lines of color combinations consisting of primary colors and secondary colors as well as black and white to make "dark" and "light" colors. The full science of colorization and pigments will be implemented next, if your prototype is successful.

Output Format:

Your program should output the correct color depending on what two colors were "mixed" on the line. Primary colors should mix together to create secondary colors. Anything mixed with "WHITE" or "BLACK" should be output as either "LIGHT X" or "DARK X" where X is the color "WHITE" or "BLACK" were mixed with. Anything mixed with itself won't change colors. You are guaranteed not to receive incompatible colors, or colors not listed in the color wheels shown above (aside from "WHITE" and "BLACK").

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
void solve(){
    cout<<"for break;";
}
int main()
{
    int t=4;
    while(t--){
        string s1,s2;
        cin>>s1>>s2;
        if(s2=="WHITE")
        cout<<"LIGHT "<<s1<<endl;
        else if(s2=="BLACK")
        cout<<"DARK "<<s1<<endl;
        else if(s1=="WHITE")
        cout<<"LIGHT "<<s2<<endl;
        else if(s1=="BLACK")
        cout<<"DARK "<<s2<<endl;
        else if((s1=="RED"&&s2=="YELLOW")||(s1=="YELLOW"&&s2=="RED"))
        cout<<"ORANGE"<<endl;
        else if((s1=="BLUE"&&s2=="YELLOW")||(s1=="YELLOW"&&s2=="BLUE"))
        cout<<"GREEN"<<endl;
        else if((s1=="BLUE"&&s2=="RED") || (s1=="RED"&&s2=="BLUE"))
        cout<<"PURPLE"<<endl;
        else if(s1==s2)
        cout<<s1<<endl;
        else
```

```
        cout<<"N/A"<<endl;
    }
    return 0;
}
```

Question 22

Question Description:

Let P be an array consisting of N numbers. The array's elements are numbered from 1 to N, *even* is an array consisting of the numerals whose numbers are even in P ($even_i = P_{2i}$, $1 \le 2i \le n$), *odd* is an array consisting of the numerals whose numbers are odd in *a* ($odd_i = P_{2i-1}$, $1 \le 2i - 1 \le n$). Then let's define the transformation of array $F(P)$ in the following manner:

- if $n > 1$, $F(P) = F(odd) + F(even)$, where operation " + " stands for the arrays' concatenation (joining together)
- if $n = 1$, $F(P) = P$

Let P be an array consisting of N numbers 1, 2, 3, ..., N. Then Q is the result of applying the transformation to the array P (so $Q = F(P)$). You are given *m* queries (l, r, u, v). Your task is to find for each query the sum of numbers Q_i, such that $l \le i \le r$ and $u \le Q_i \le v$. You should print the query results modulo *mod*.

Constraints:

$1 \le N \le 10^{18}$

$1 \le M \le 10^5$

$1 \le mod \le 10^9$

$1 \le l \le r \le n$

$1 \le u \le v \le 10^{18}$

Input Format:

The first line contains three integers N, M, *mod*.

Next M lines describe the queries. Each query is defined by four integers l, r, u, v.

Output Format:

Print *m* lines each containing an integer remainder modulo *mod* of the query result.

Program Code:

```
#include <stdio.h>

int md;

int s(int n) {
    return (n % 2 == 0 ? (n / 2 % md) * ((n + 1) % md) : (n % md) * ((n + 1) /
}
```

```
int sum, cnt;

void queries(long long n, long long k, long long a) {
    int sum0, cnt0, sum1, cnt1;

    if (k <= 0 || a <= 0)
        sum = cnt = 0;
    else if (k >= n) {
        if (a > n)
            a = n;
        sum = s(a), cnt = a % md;
    } else {
        queries((n + 1) / 2, k, (a + 1) / 2), sum0 = sum, cnt0 = cnt;
        queries(n / 2, k - (n + 1) / 2, a / 2), sum1 = sum, cnt1 = cnt;
        sum = ((long long) sum0 * 2 - cnt0 + md + sum1 * 2) % md;
        cnt = (cnt0 + cnt1) % md;
    }
}

int main() {
    int n;
    int m;

    scanf("%d%d%d",&n,&m,&md);
    while (m--) {
        long long l, r, a, b;
        int ans;

        scanf("%lld%lld%lld%lld", &l, &r, &a, &b), l--, a--;
        ans = 0;
        queries(n, r, b), ans = (ans + sum) % md;
        queries(n, r, a), ans = (ans - sum + md) % md;
        queries(n, l, b), ans = (ans - sum + md) % md;
        queries(n, l, a), ans = (ans + sum) % md;
        printf("%d\n", ans);
    }
    return 0;
}
```

Question 24

Question Description:

Neeraj definition a string is said to be a palindrome if it does change when get reversed. 13131 is a nice example of a palindrome.

Given a string S, you are allowed to insert any characters at any position of the string, find the minimum number of characters required to make the string a palindrome.

Constraints:

$1 \leq N \leq 100$

Input Format:

Each case contains a string of lowercase letters denoting the string for which we want to generate a palindrome.

You may safely assume that the length of the string will be positive and no more than 100.

Output Format:

For each case, print the case number and the minimum number of characters required to make string to a palindrome.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
void garbage(){
    cout<<"int go(int f,int s)vcin>>a; ";
}
int findMinInsertions(string str, int l, int h)
{
    if (l > h) return INT_MAX;
    if (l == h) return 0;
    if (l == h - 1) return (str[l] == str[h])? 0 : 1;
    return (str[l] == str[h])?
                   findMinInsertions(str, l + 1, h - 1):
                   (min(findMinInsertions(str, l, h - 1),
                   findMinInsertions(str, l + 1, h)) + 1);
}
int main()
{
    string s;
    cin>>s;
    cout << findMinInsertions(s, 0, s.length() - 1);
    return 0;
}
```

Question 25

Question Description:

Recently Aarush has become keen on physics. Anna V., his teacher noticed Aarush's interest and gave him a fascinating physical puzzle a half-decay tree.

A half-decay tree is a complete binary tree with the height *h*. The height of a tree is the length of the path (in edges) from the root to a leaf in the tree. While studying the tree Aarush can add electrons to vertices or induce random decay with synchrophasotron.

Random decay is a process during which the edges of some path from the root to the random leaf of the tree are deleted. All the leaves are equiprobable. As the half-decay tree is the school property, Aarush will return back the deleted edges into the tree after each decay.

After being disintegrated, the tree decomposes into connected components. Charge of each component is the total quantity of electrons placed in vertices of the component. Potential of disintegrated tree is the maximum from the charges of its connected components. Each time before inducing random decay Aarush is curious about the mathematical expectation of potential of the tree after being disintegrated.

Constraints:

$1 \leq h \leq 30$

$1 \leq q \leq 10^5$

$1 \leq v \leq 2^{h+1} - 1$

$0 \leq e \leq 10^4$

Input Format:

First line will contain two integers h and q. Next q lines will contain a query of one of two types:

- add v e

Aarush adds e electrons to vertex number v. v and e are integers.

The vertices of the tree are numbered in the following way: the root is numbered with 1, the children of the vertex with number x are numbered with $2x$ and $2x + 1$.

- decay

Aarush induces tree decay.

Output Format:

For each query decay solution you should output the mathematical expectation of potential of the tree after being disintegrated.

The absolute or relative error in the answer should not exceed 10^{-4}.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int h,q,v,e;string str;map<int,int> f;
double puzzle(int u,int mx) {return (f[u]<=mx)?mx:(0.5*(puzzle(u<<1,max(mx,f[u
int main(){
cin>>h>>q;
    while (q--){
        cin>>str;
        if (str[0]=='a'){
            scanf("%d %d",&v,&e);
            while (v) f[v]+=e,v>>=1;
        }
        else printf("%.2lf\n",puzzle(1,0));
    }
    return 0;
}
```

Question 26

Question Description:

After the long contest, Sameer returned home and got angry after seeing his room dusty.

Who likes to see a dusty room after a brain storming programming contest? After checking a bit he found a brush in his room which has width w.

Dusts are defined as 2D points. And since they are scattered everywhere, Sameer is a bit confused what to do. So, he attached a rope with the brush such that it can be moved horizontally (in X axis) with the help of the rope but in straight line.

He places it anywhere and moves it. For example, the y co-ordinate of the bottom part of the brush is 2 and its width is 3, so the y coordinate of the upper side of the brush will be 5. And if the brush is moved, all dusts whose y co-ordinates are between 2 and 5 (inclusive) will be cleaned.

After cleaning all the dusts in that part, Sameer places the brush in another place and uses the same procedure. He defined a move as placing the brush in a place and cleaning all the dusts in the horizontal zone of the brush.

You can assume that the rope is sufficiently large. Now Sameer wants to clean the room with minimum number of moves. Since he already had a contest, his head is messy. So, help him.

You can assume that the rope is sufficiently large. Now Sameer wants to clean the room with minimum number of moves. Since he already had a contest, his head is messy. So, help him.

Constraints:

$1 \leq N \leq 50000$

$1 \leq w \leq 10000$

$-10^9 \leq x_i, y_i \leq 10^9$

Input Format:

Each case starts with a blank line. The next line contains two integers N and w, means that there are N dust points.

Each of the next N lines will contain two integers: x_i y_i, denoting coordinates of the dusts. You can assume that and all points are distinct.

Output Format:

For each case print the case number and the minimum number of moves.

Program Code:

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


using namespace std;


int partition(int array[],int leftIndex,int rightIndex){
    int pivotValue = array[rightIndex];
    int toBePivotIndex = (leftIndex - 1);
    for(int comparisonIndex = leftIndex; comparisonIndex <= rightIndex - 1; co
        if (
```

```
            array[comparisonIndex] < pivotValue
        ) {

            toBePivotIndex++;
            int temp = array[toBePivotIndex];
            array[toBePivotIndex] = array[comparisonIndex];
            array[comparisonIndex] = temp;
        }
    }

    int temp = array[toBePivotIndex+1];
    array[toBePivotIndex+1] = array[rightIndex];
    array[rightIndex] = temp;

    return (toBePivotIndex + 1); // new pivot point
}


void quickSort(int array[],int leftIndex,int rightIndex){

    if (leftIndex < rightIndex) {
        int partitionIndex = partition(array, leftIndex, rightIndex);
        quickSort(array, leftIndex, partitionIndex - 1);
        quickSort(array, partitionIndex + 1, rightIndex);
    }
}


int main(){

    int numberOfDustPoints,widthOfBrush,xCoordinate,yCoordinate;

        int numberOfMoves = 0;
        cin>>numberOfDustPoints>>widthOfBrush;
        int dustPointsYCoordinates[numberOfDustPoints];

        for(int i = 0; i < numberOfDustPoints; i++){
            cin >> xCoordinate >> yCoordinate;
            dustPointsYCoordinates[i] = yCoordinate;
        }

        quickSort(dustPointsYCoordinates,0, numberOfDustPoints-1);


        int currentBrushYCoordinate = dustPointsYCoordinates[0];
        numberOfMoves++;

        for (int i = 0; i < numberOfDustPoints; i++) {
        if(currentBrushYCoordinate + widthOfBrush < dustPointsYCoordinates[i])
        currentBrushYCoordinate = dustPointsYCoordinates[i];
        numberOfMoves++;
        }
```

```
    }
        cout <<numberOfMoves;


    return 0;
}
```

Question 27

Question Description:

Prof.Dr. Ramalingam need representing positive integer N as a sum of addends, where each addends is an integer number containing only 1s.

For example, he can represent 121 as 121=111+11+–1. Help him to find the least number of digits 1 in such sum.

Constraints:

$1 \leq n < 10^{15}$

Input Format:

The first line of the input contains integer N.

Output Format:

Print expected minimal number of digits 1.

Program Code:

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

long long n,a[17];

int dfs(long long n,int x)
{
    int num=n/a[x];n%=a[x];
    if (!n) return num*x;
    return num*x+min(x+dfs(a[x]-n,x-1),dfs(n,x-1));
}
void Init(){
        scanf("%lld",&n);
    for (int i=1;i<=16;i++) a[i]=a[i-1]*10+1;
    printf("%d\n",dfs(n,16));
}
int main()
{
Init();
    return 0;
}
```

Question 30

Question Description:

Leopard is in the Amusement Park. And now she is in a queue in front of the Ferris wheel. There are n people (or Leopard more precisely) in the queue: we use first people to refer one at the head of the queue, and n-th people to refer the last one in the queue.

There will be k gondolas, and the way we allocate gondolas looks like this:

- When the first gondolas come, the q_1 people in head of the queue go into the gondolas.
- Then when the second gondolas come, the q_2 people in head of the remain queue go into the gondolas.
- The remain q_k people go into the last (k-th) gondolas.

Note that $q_1, q_2, ..., q_k$ must be positive. You can get from the statement that and $q_i > 0$.

You know, people don't want to stay with strangers in the gondolas, so your task is to find an optimal allocation way (that is find an optimal sequence q) to make people happy. For every pair of people i and j, there exists a value u_{ij} denotes a level of unfamiliar. You can assume $u_{ij} = u_{ji}$ for all i,j ($1 \le i,j \le n$) and $u_{ii} = 0$ for all i ($1 \le i \le n$). Then an unfamiliar value of a gondolas is the sum of the levels of unfamiliar between any pair of people that is into the gondolas.

A total unfamiliar value is the sum of unfamiliar values for all gondolas. Help Leopard to find the minimal possible total unfamiliar value for some optimal allocation.

Constraints:

$1 \le n \le 4000$

$1 \le k \le min(n, 800)$

Input Format:

The first line contains two integers n and k the number of people in the queue and the number of gondolas.

Each of the following n lines contains n integers matrix u, ($0 \le u_{ij} \le 9$, $u_{ij} = u_{ji}$ and $u_{ii} = 0$).

Output Format:

Print an integer the minimal possible total unfamiliar value.

Program Code:

```
#include<cstdio>
#include<iostream>
using namespace std;
inline int getint(){
char c;
while((c=getchar())<'0'||c>'9');
return c-'0';
}
const int N=4005,inf=.5e9;
int n,k,sum[N][N],f[N],g[N];
int main(){
cin>>n>>k;
```

```
for(int i=1;i<=n;i++)
for(int j=1;j<=n;j++)
sum[i][j]=sum[i-1][j]+sum[i][j-1]-sum[i-1][j-1]+getint();
g[n+1]=n;
for(int kk=2;kk<=k;kk++)
for(int i=n;i;i--){
f[i]=-inf;
for(int j=g[i];j<=g[i+1]&&j<i;j++){
int now=f[j]-sum[j][j]+sum[j][i];
if(now>f[i]){
f[i]=now;
g[i]=j;
}

}
}
printf("%d\n",sum[n][n]/2-f[n]);
}
```

Question 31

Question Description:

The spring is coming and it means that a lot of fruits appear on the counters. One sunny day young girl Valarmathi decided to go shopping.

She made a list of *m* fruits he wanted to buy. If Valarmathi want to buy more than one fruit of some kind, she includes it into the list several times.

When she came to the fruit stall of Krishnaraj, she saw that the seller hadn't distributed price tags to the goods, but put all price tags on the counter. Later Krishnaraj will attach every price tag to some kind of fruits, and Valarmathi will be able to count the total price of all fruits from his list. But Valarmathi wants to know now what can be the smallest total price (in case of the most «lucky» for him distribution of price tags) and the largest total price (in case of the most «unlucky» for him distribution of price tags).

Constraints:

$1 \le n, m \le 100$

Input Format:

The first line of the input contains two integer number *n* and *m* the number of price tags (which is equal to the number of different kinds of fruits that Ashot sells) and the number of items in Valarmathi's list.

The second line contains *n* space-separated positive integer numbers. Each of them doesn't exceed 100 and stands for the price of one fruit of some kind.

The following *m* lines contain names of the fruits from the list. Each name is a non-empty string of small Latin letters which length doesn't exceed 32.

It is guaranteed that the number of distinct fruits from the list is less of equal to *n*.

Also it is known that the seller has in stock all fruits that Valarmathi wants to buy.

Output Format:

Print two numbers a and b ($a \leq b$) the minimum and the maximum possible sum which Valarmathi may need to buy all fruits from his list.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
map <string,int> p;
int n,m,g[102],c[102],cnt;
string s; int main()
{
    cin>>n>>m;
    for(int i=0;i<n;i++)
    cin>>g[i];
    sort(g,g+n);
    for(int i=0;i<m;i++){
        cin>>s;
        if(!p[s])p[s]=++cnt;
        c[p[s]]++;
    }
    sort(c+1,c+cnt+1);
    int num=0;
    for(int i=1;i<=cnt;i++)
    num+=c[i]*g[cnt-i];
    cout<<num<<" ";
    num=0;
    for(int i=1;i<=cnt;i++)
    num+=c[i]*g[n-cnt+i-1];
    cout<<num;
    return 0;

}
```

Question 32

Question Description:

A stealing got into a matches warehouse and wants to steal as many matches as possible.

In the warehouse there are m containers, in the i-th container there are a_i matchboxes, and each matchbox contains b_i matches.

All the matchboxes are of the same size. The stealing's rucksack can hold n matchboxes exactly.

Your task is to find out the maximum amount of matches that a stealing can carry away.

He has no time to rearrange matches in the matchboxes, that's why he just chooses not more than n matchboxes so that the total amount of matches in them is maximal.

Constraints:

$1 \leq n \leq 2 \cdot 10^8$

$1 \leq m \leq 20$

$1 \le a_i \le 10^8$

$1 \le b_i \le 10$

Input Format:

The first line of the input contains integer n and integer m.

The $i + 1$-th line contains a pair of numbers a_i and b_i. All the input numbers are integer.

Output Format:

Output the only number answer to the problem.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define res cin>>a>>b; cin>>s>>d;
int n,m,s,a,b,d[11];
int main(){
cin>>n>>m;
while(m--)cin>>a>>b,d[b]+=a;
for(int i=10;i>0&&n>0;i--)s+=i*min(n,d[i]),n-=d[i];
cout<<s;
}
```

Question 35

Question Description:

Devika is addicted to meat! Malik wants to keep her happy for n days. In order to be happy in i-th day, she needs to eat exactly a_i kilograms of meat.

There is a big shop uptown and Malik wants to buy meat for her from there. In i-th day, they sell meat for p_i dollars per kilogram.

Malik knows all numbers $a_1, ..., a_n$ and $p_1, ..., p_n$. In each day, he can buy arbitrary amount of meat, also he can keep some meat he has for the future.

Malik is a little tired from cooking meat, so he asked for your help. Help him to minimize the total money he spends to keep Devika happy for n days.

Constraints:

$1 \le n \le 10^5$

$1 \le a_i, p_i \le 100$

Input Format:

The first line of input contains integer n, the number of days.

In the next n lines, i-th line contains two integers a_i and p_i, the amount of meat Devika needs and the cost of meat in that day.

Output Format:

Print the maximal number of orders that can be accepted.

Program Code:

```
#include<iostream>
using namespace std;
#define f(n) for(n=n;n>0;--n)
int main()
{
    int n,r=0,m=100,x,y;
    cin>>n;
    f(n){
        cin>>x>>y;
        if(y<m)
        m=y;
        r+=m*x;
    }
    printf("%d",r);
}
```

Question 36

Question Description:

Samantha has given an array of N elements, you must make it a co-prime array in as few moves as possible.

In each move you can insert any positive integral number you want not greater than 10^9 in any place in the array.

An array is co-prime if any two adjacent numbers of it are co-prime.

In the number theory, two integers a and b are said to be co-prime if the only positive integer that divides both of them is 1.

Constraints:

$1 \leq n \leq 1000$

$1 \leq a_i \leq 10^9$

Input Format:

The first line contains integer n the number of elements in the given array.

The second line contains n integers a_i the elements of the array a.

Output Format:

Print integer k on the first line the least number of elements needed to add to the array a to make it co-prime.

The second line should contain $n + k$ integers a_j the elements of the array a after adding k elements to it.

Note that the new array should be co-prime, so any two adjacent values should be co-prime.

Also the new array should be got from the original array a by adding k elements to it.

If there are multiple answers you can print any one of them.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int n,x,p=1;
int main(){
  vector<int>X;
  for(cin>>n;cin>>x;X.push_back(p=x))if(__gcd(p,x)>1)X.push_back(1);
  cout<<X.size()-n<<endl;
  for(int x:X)cout<<x<<" ";
  return 0;
  cout<<"cin>>y[i];";
}
```

Question 37

Question Description:

A remote island chain contains n islands, labeled 1 through n. Bidirectional bridges connect the islands to form a simple cycle a bridge connects islands 1 and 2, islands 2 and 3, and so on, and additionally a bridge connects islands n and 1.

The center of each island contains an identical pedestal, and all but one of the islands has a fragile, uniquely colored statue currently held on the pedestal. The remaining island holds only an empty pedestal.

The islanders want to rearrange the statues in a new order. To do this, they repeat the following process: First, they choose an island directly adjacent to the island containing an empty pedestal.

Then, they painstakingly carry the statue on this island across the adjoining bridge and place it on the empty pedestal.

Determine if it is possible for the islanders to arrange the statues in the desired order.

Constraints:

$2 \le n \le 200\,000$

$0 \le a_i \le n - 1$

$0 \le b_i \le n - 1$

Input Format:

The first line contains a single integer n the total number of islands.

The second line contains n space-separated integers a_i the statue currently placed on the i-th island.

If $a_i = 0$, then the island has no statue. It is guaranteed that the a_i are distinct.

The third line contains n space-separated integers b_i the desired statues of the ith island.

Once again, $b_i = 0$ indicates the island desires no statue. It is guaranteed that the b_i are distinct.

Output Format:

Print "YES" (without quotes) if the rearrangement can be done in the existing network, and "NO" otherwise.

Program Code:

```
#include <iostream>
int main()
{
    int n,i,j,k;
    int b[100];
    int a[100];
    std::cin>>n;
    for(i=0;i<n;i++)
    std::cin>>a[i];
    for(i=0;i<n;i++)
    std::cin>>b[i];
    for(i=0;i<n;i++)
    if(a[i]==0)
    j=i;
    for(i=0;i<n;i++)
    if(b[i]==0)
    k=i;
    if(j==k)
    std::cout << "YES";
    else
    std::cout << "NO";
    return 0;
}
```

Question 38

Question Description:

Nadanan's company employed n people. Now Nadanan needs to build a tree hierarchy of «supervisor-surbodinate» relations in the company (this is to say that each employee, except one, has exactly one supervisor).

There are m applications written in the following form: «employee a_i is ready to become a supervisor of employee b_i at extra cost c_i».

The qualification q_j of each employee is known, and for each application the following is true: $q_{ai} > q_{bi}$.

Would you help Nadanan calculate the minimum cost of such a hierarchy, or find out that it is impossible to build it.

Constraints:

$1 \le n \le 1000$

$0 \le q_j \le 10^6$

$0 \le m \le 10000$

$1 \le a_i, b_i \le n, 0 \le c_i \le 10^6$

Input Format:

The first input line contains integer n amount of employees in the company. The following line contains n space-separated numbers q_j the employees' qualifications. The following line contains number m amount of received applications.

The following m lines contain the applications themselves, each of them in the form of three space-separated numbers: a_i, b_i and c_i.

Different applications can be similar, i.e. they can come from one and the same employee who offered to become a supervisor of the same person but at a different cost. For each application $q_{ai} > q_{bi}$.

Output Format:

Output the only line the minimum cost of building such a hierarchy, or -1 if it is impossible to build it.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define ans cin>>ans[0];cin>>a>>b>>c;
#define f(n) for(int i=0;i<n;i++)
void solve(){}
int main(){
    int n;cin>>n;
    int a[n];f(n) cin>>a[i];
    int M;cin>>M;
    map<int,int> m;
    while(M--){
        int x,y,c;cin>>x>>y>>c;
        if(m.find(y)==m.end())
            m[y]=c;
        else if(c<m[y])
            m[y]=c;
    }
    if((int)m.size()==n-1){
        long long int sum=0;
        for(auto j : m){
            sum+=j.second;
        }
        cout<<sum;
    }
    else cout<<-1;
}
```

Question 39

Question Description:

A sportsman starts from point $x_{start} = 0$ and runs to point with coordinate $x_{finish} = m$ (on a straight line). Also, the sportsman can jump — to jump, he should first take a run of length of not less than s meters (in this case for these s meters his path should have no obstacles), and after that he can jump over a length of not more than d meters. Running and jumping is permitted only in the direction from left to right. He can start andfinish a jump only at the points with integer coordinates in which there are no obstacles. To overcome some obstacle, it is necessary to land at a point which is strictly to the right of this obstacle.

On the way of an athlete are n obstacles at coordinates $x_1, x_2, ..., x_n$. He cannot go over the obstacles, he can only jump over them. Your task is to determine whether the athlete will be able to get to the finish point.

Constraints:

$1 \le n \le 200\,000$

$2 \le m \le 10^9$

$1 \le s, d \le 10^9$

$1 \le a_i \le m - 1$

Input Format:

The first line of the input contains four integers n, m, s and d the number of obstacles on the runner's way, the coordinate of the finishing point, the length of running before the jump and the maximum length of the jump, correspondingly.

The second line contains a sequence of n integers $a_1, a_2, ..., a_n$ the coordinates of the obstacles.

It is guaranteed that the starting and finishing point have no obstacles, also no point can have more than one obstacle, The coordinates of the obstacles are given in an arbitrary order.

Output Format:

If the runner cannot reach the finishing point, print in the first line of the output "IMPOSSIBLE" (without the quotes).

If the athlete can get from start to finish, print any way to do this in the following format:

- print a line of form "RUN X>" (where "X" should be a positive integer), if the athlete should run for "X" more meters;
- print a line of form "JUMP Y" (where "Y" should be a positive integer), if the sportsman starts a jump and should remain in air for "Y" more meters.

All commands "RUN" and "JUMP" should strictly alternate, starting with "RUN", besides, they should be printed chronologically. It is not allowed to jump over the finishing point but it is allowed to land there after a jump. The athlete should stop as soon as he reaches finish.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
const int N = 2e5+5;
int p[N],par,x[N];
int main(){
    int n,i,m,s,d;
    cin>>n>>m>>s>>d;
    x[0]=-1;
    for(i=1;i<=n;++i)
        cin>>x[i];
    sort(x,x+n+1);
    par = n;
    for(i=n-1;i>=0;--i)
        if(x[i+1]-x[i]>=s+2 && x[par]-x[i+1]<=d-2)
            p[i]= par,par = i;
    if(par>0){
        printf("IMPOSSIBLE\n");
    }
    else{
        for(i=0;i<n;i= p[i])
            printf("RUN %d\nJUMP %d\n",x[i+1]-x[i]-2,x[p[i]]-x[i+1]+2);
        if(x[n]+1<m)
            printf("RUN %d\n",m-x[n]-1);
    }
    return 0;
    cout<<"cin>>a[i];";
}
```

Question 40

Question Description:

A long time ago, in a galaxy far far away two giant IT-corporations Avocado and Bobol continue their fierce competition. Crucial moment is just around the corner: Bobol is ready to release it's new tablet Lastus 3000.

This new device is equipped with specially designed artificial intelligence (AI). Employees of Avocado did their best to postpone the release of Lastus 3000 as long as possible. Finally, they found out, that the name of the new artificial intelligence is similar to the name of the phone, that Avocado released 200 years ago.

As all rights on its name belong to Avocado, they stand on changing the name of Bobol's artificial intelligence.

Pineapple insists, that the name of their phone occurs in the name of AI as a substring. Because the name of technology was already printed on all devices, the Bobol's director decided to replace some characters in AI name with "#". As this operation is pretty expensive, you should find the minimum number of characters to replace with "#", such that the name of AI doesn't contain the name of the phone as a substring.

Substring is a continuous subsequence of a string.

Constraints:

$1 \leq n \leq 100$

Input Format:

The first line of the input contains the name of AI designed by Bobol, its length doesn't exceed 100 000 characters.

Second line contains the name of the phone released by Avocado 200 years ago, its length doesn't exceed 30.

Both string are non-empty and consist of only small English letters.

Output Format:

Print the minimum number of characters that must be replaced with "#" in order to obtain that the name of the phone doesn't occur in the name of AI as a substring.

Program Code:

```
#include <iostream>
using namespace std;
int main()
{
    string s,t;
    std::cin>>s>>t;
    int o = s.find(t);
    int c =0;
    while(o!=-1)
    {         c++;
        o = s.find(t,o+t.length());
    }
    cout<<c<<endl;
    }
```

Question 42

Question Description:

Lawrence could not sleep lately, because he had nightmares. In one of his nightmares (which was about an unbalanced global round), he decided to fight back and propose a problem below (which you should solve) to balance the round, hopefully setting him free from the nightmares.

A non-empty array $b1,b2,\ldots,bm$ is called good, if there exist M integer sequences which satisfy the following properties:

- The i-th sequence consists of Bi consecutive integers (for example if bi=3 then the i-th sequence can be (−1,0,1) or (−5,−4,−3) but not (0,−1,1) or (1,2,3,4)).
- Assuming the sum of integers in the i-th sequence is $sumi$, we want $sum1+sum2+\ldots+summ$ to be equal to 0.

You are given an array $a1,a2,\ldots,an$. It has $2n-1$ nonempty subsequences. Find how many of them are good.

As this number can be very large, output it modulo 10^9+7.

An array c is a subsequence of an array d if c can be obtained from d by deletion of several (possibly, zero or all) elements.

Constraints:

2≤ N ≤2·10^5

1≤ai≤10^9

Input Format:

The first line contains a single integer n the size of array a.

The second line contains n integers $a1,a2,…,an$ elements of the array.

Output Format:

Print a single integer — the number of nonempty good subsequences of a, modulo 10^9+7.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
#define int long long
const int N=1e6,D=1e9+7;
int a[N],n,x,s,c[N],A;
void aas(){
    cout<<"int mul(int x,int n,int mod)";
}
signed main()
{
    a[0]=1;
    for(int i=1;i<N;i++)
        a[i]=a[i-1]*2%D;
    cin>>n;
    for(int i=1;i<=n;i++)
        cin>>x,c[__builtin_ctz(x)]++;
    for(int i=30;i;i--)
    {
        s+=c[i];
        if(c[i]>1)
            (A+=(a[c[i]-1]-1)*a[s-c[i]])%=D;
    }
    cout<<(A+(a[c[0]]-1)*a[n-c[0]])%D;
}
```

Question 44

Question Description:

This is the easy version of the problem. The only difference is maximum value of Ai.

Once in Vettayapuram aranmanai *Divan* found an array A consisting of positive integers. Now he wants to reorder the elements of A to maximize the value of the following function:

$$\sum_{i=1}^{n} \gcd(a1, a2, \ldots, ai),$$

where gcd($x1,x2,…,xk$) denotes the greatest common divisor of integers $x1,x2,…,xk$, and gcd(x)=x for any integer x.

Reordering elements of an array means changing the order of elements in the array arbitrary, or leaving the initial order.

Of course, *Divan* can solve this problem. However, he found it interesting, so he decided to share it with you.

Constraints:

1≤ N ≤10^5

1≤*ai*≤5·10^6

Input Format:

The first line contains a single integer N the size of the array A.

The second line contains N integers $a1,a2,\ldots,an$ the array A.

Output Format:

Output the maximum value of the function that you can get by reordering elements of the array A.

Program Code:

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
int const M=5000000;int i,j,n,s,x,e[M+100],f[M+100],d[M+100];
signed main(){
cin>>n;
for (i=1;i<=n;i++) scanf("%lld",&x),f[x]++;
for (i=1;i<=M;i++)
for (j=i;j<=M;j+=i)
e[i]+=f[j];
for (i=M;i>0;i--){
for (s=0,j=i*2;j<=M;j+=i) s=max(s,d[j]-e[j]*i);
d[i]=e[i]*i+s;
}
printf("%lld\n",d[1]);
return 0;
}
```

Question 45

Question description

You have infinite cards for each number between 1 and N (inclusive of them). Your task is to select three integers such that after sorting them in ascending order, the difference between the adjacent number is less than or equal to two. Find the number of ways to choose three numbers and print them.

Note: The order of numbers does not matter.

Constraints

1≤T≤20000

$1 \leq N \leq 200000$

Input format

- The first line contains an integer T denoting the number of test cases.
- For each test case, the first and only line contains an integer N.

Output format

Print T lines, one for each test case, denoting the number of ways.

Sample Input

2

1

3

Sample Output

1

10

Explanation

For N=1 there is only one way:

1. (1,1,1)

For N=3

1. (1,1,1)
2. (2,2,2)
3. (3,3,3)
4. (1,2,3)
5. (1,1,3)
6. (1,1,2)
7. (1,2,2)
8. (2,2,3)
9. (1,3,3)
10. (2,3,3)

These are the 10 possible ways.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
using ll = long long int;
int main() {
ios_base::sync_with_stdio(false);
cin.tie(NULL);
cout.tie(NULL);
```

```
//preSum();
ll t;
cin>>t;
while(t--){
ll n;
cin>>n;
if(n==1)
printf("1\n");
else if(n==2)
printf("4\n");
else if(n==3)
printf("10\n");
else
printf("%lld\n",9*n-18);
}
}
```

Question 46

Question Description:

Krishnes has given a directed acyclic graph with N vertices and M edges. For all edges $a \rightarrow b$ in the graph, $a<b$ holds.

You need to find the number of pairs of vertices X, Y, such that $x>y$ and after adding the edge $x \rightarrow y$ to the graph, it has a Hamiltonian path.

Constraints:

$1 \leq T \leq 5$

$1 \leq N \leq 150000$

$0 \leq m \leq \min(150000, n(n-1)2)$

$1 \leq a < b \leq n$

Input Format:

The first line of input contains one integer T: the number of test cases.

The next lines contains the descriptions of the test cases.

In the first line you are given two integers N and M: the number of vertices and edges in the graph.

Each of the next M lines contains two integers A, B, specifying an edge $a \rightarrow b$ in the graph. No edge $a \rightarrow b$ appears more than once.

Output Format:

For each test case, print one integer: the number of pairs of vertices X, Y, $x>y$, such that after adding the edge $x \rightarrow y$ to the graph, it has a Hamiltonian path.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int T,n,m,pr[150010],l[150010],f[150010][2],a[4],b[4];
vector<int>E[150010];
void direction(int x,int c){}
void pairs();
void pairs(){
        scanf("%d%d",&n,&m);
        for(int i=0;i<=n+1;i++) E[i].clear(),pr[i]=0,f[i][0]=f[i][1]=0;
        for(int i=1,u,v;i<=m;i++){
                scanf("%d%d",&u,&v);
                if(u+1==v) pr[v]=1;
                else E[v].push_back(u);
        }
        pr[1]=pr[n+1]=1;
        for(int i=2;i<=n;i++) E[i].push_back(0),E[n+1].push_back(i-1);
        int L=0,R=n+1;
        while(L<=n&&pr[L+1]) L++;
        while(R&&pr[R]) R--;
        if(R==0) return printf("%lld\n",1ll*n*(n-1)/2),void();
        for(int i=1;i<=n;i++) l[i]=pr[i]?l[i-1]:i;
        f[L][0]=1,f[L][1]=2;
        for(int i=L;i<=n;i++) for(int u:E[i+1]) for(int k=0;k<2;k++) if(l[i]<=
        for(int i=L;i>=1;i--) for(int u:E[i+1]) for(int k=0;k<2;k++) if(l[i]<=
        for(int i=0;i<4;i++) a[i]=b[i]=0;
        for(int i=0;i<=L;i++) a[f[i][0]]++;
        for(int i=R-1;i<=n;i++) b[f[i][0]]++;
        long long ans=0;
        for(int p=0;p<4;p++) for(int q=0;q<4;q++) if(p&q) ans+=1ll*a[p]*b[q];
        printf("%lld\n",ans-(L+1==R));
}
int main(){
        scanf("%d",&T);
        while(T--) pairs();
}
```

Question 49

Question Description:

Professor Wiki has performed some experiments on rays. The setup for *n* rays is as follows.

There is a rectangular box having exactly *n* holes on the opposite faces.

All rays enter from the holes of the first side and exit from the holes of the other side of the box.

Exactly one ray can enter or exit from each hole. The holes are in a straight line.

Professor Wiki is showing his experiment to his students. He shows that there are cases, when all the rays are intersected by every other ray.

A curious student asked the professor: "Sir, there are some groups of rays such that all rays in that group intersect every other ray in that group.

Can we determine the number of rays in the largest of such groups?".

Professor Wiki now is in trouble and knowing your intellect he asks you to help him.

Constraints:

$1 \leq N \leq 10^6$

Input Format:

The first line contains n (), the number of rays.

The second line contains n distinct integers.

The i-th integer x_i ($1 \leq x_i \leq n$) shows that the x_i-th ray enters from the i-th hole.

Similarly, third line contains n distinct integers.

The i-th integer y_i ($1 \leq y_i \leq n$) shows that the y_i-th ray exits from the i-th hole. All rays are numbered from 1 to n.

Output Format:

Output contains the only integer which is the number of rays in the largest group of rays all of which intersect each other.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int n,x,i; int a[1000020];
int p[1000020];
int f[1000020];
int main()
{     cin>>n;
    for(i=0;i<n;i++)
    {     cin>>x;
    p[x]=i;
    }
    for(i=0;i<n;i++)
    {
    scanf("%d",&x);
    a[i]=-p[x]-1;
    }
    for(i=0;i<n;i++)
        *lower_bound(f,f+n,a[i])=a[i];
    int zero=0;
    printf("%ld\n",lower_bound(f,f+n,zero)-f);
    return 0; }
```

Question 55

Given a chess board having N×N cells, you need to place N queens on the board in such a way that no queen attacks any other queen.

Input:
The only line of input consists of a single integer denoting *N*.

Output:
If it is possible to place all the *N* queens in such a way that no queen attacks another queen, then print *N* lines having *N* integers. The integer in ith line and jth column will denote the cell (i,j) of the board and should be *1* if a queen is placed at (i,j) otherwise *0*. If there are more than way of placing queens print any of them. If it is not possible to place all *N* queens in the desired way, then print "Not possible" (without quotes).

Constraints:
$1 \leq N \leq 10$.

Program Code:

```
#include<iostream>
using namespace std;
int n;
bool grid[10][10];
bool isSafe(int row, int col){
int i,j;
 for(i=0;i<row;++i) if(grid[i][col]) return false;
 for(i=row,j=col;i>=0 and j>=0;--i,--j) if(grid[i][j]) return false;
 for(i=row,j=col;i>=0 and j<n;--i,++j) if(grid[i][j]) return false;
 return true;
}
bool solveQueen(int row){
 if(row>=n) return true;
 for(int col=0;col<n;++col){
 if(isSafe(row,col)){
 grid[row][col]=true;
 if(solveQueen(row+1)) return true;
 grid[row][col]=false;
 }
 }
 return false;
}
int main(){
 cin>>n;
 int i;
 if(solveQueen(0)){
 for(i=0;i<n;++i){
 for(int j=0;j<n;++j) cout<<grid[i][j]<<" ";
 cout<<endl;
 }
 }
 else cout<<"Not possible\n";
 return 0;
}
```

Question 58

Problem Statement

You are given three arrays a1…n,b1…n,c1…n and two numbers ***M*** and ***K***. Find a lexicographically minimum {x,y,z} such that there are exactly ***K*** indices i(1≤i≤n) where x∗ai+y∗bi−ci∗z=M∗f for some integer f. Also, you are given ranges of x, y, and z-- l1..3,r1..3((l1≤x≤r1,l2≤y≤r2,l3≤z≤r3). Here, a triplet of integers {x1,y1,z1} is considered to be lexicographically smaller than a triplet {x2,y2,z2} if sequence [x1,y1,z1] is lexicographically smaller than sequence [x2,y2,z2]. A sequence ***a*** is lexicographically smaller than a sequence ***b*** if in the first position where *a* and *b* differ, the sequence ***a*** has a smaller element than the corresponding element in ***b***.

Input format

- The first line contains one three integers n,m,K(1≤n≤10000,0≤K≤n,1≤M≤15).
- The next **n** lines contain three integers ai,bi,ci(1≤ai,bi,ci≤109).
- The next three lines contain two integers li,ri(1≤li≤ri≤109).

Output format

If an answer does not exist, print **-1**. Otherwise, print desirable {x,y,z}.

Sample Input

4 3 4

5 6 1

2 6 9

11 5 6

1 1 1

1 10

1 10

1 10

Sample Output

3 3 3

Explanation

Since, K=n=4, the above condition must hold for all indices

. i=1) 3∗5+3∗6−3∗1=30 i=2) 3∗2+3∗6−3∗9=−3 i=3) 3∗11+3∗5−3∗6=30 i=4) 3∗1+3∗1−3∗1=3.

Program Code:

```
#include <iostream>
#include<bits/stdc++.h>
#define f1 for(i=0;i<n;i++)
using namespace std;
long long int min(long long int x, long long int y){
        if(x < y)
            return x;
        else
            return y;
```

```
}
int main(){
        int n, m, K;
        cin >> n >> m >> K;
        long long int a[n],b[n],c[n];
        long long int i,j,l;
        int p,T = 0;
        for(i = 0; i < n; i++)
        cin >> a[i] >> b[i] >> c[i];
        long long int lx,rx,ly,ry,lz,rz;
        cin >> lx >> rx;
        cin >> ly >> ry;
        cin >> lz >> rz;
        for(i = lx; i < min(rx, lx + m); i++){
                for(j = ly; j < min(ry,ly + m);j++){
                        for(l = lz;l < min(rz,lz + m);l++){
                                T=0;
                                for(p = 0; p < n; p++){
                                        if((a[p] * i + b[p] * j- c[p] * l) % m
                                        T++;
                                }
                                if(T==K)
                                break;
                        }
                        if(l < min(rz,lz + m))
                        break;
                }
                if(j < min(ry,ly + m))
                break;

        }
        if(i < min(rx, lx + m)){
                cout << i << " " << j << " " << l;
        }
        else
        cout << "-1" << endl;
}
```

Question 59

Problem Statement

Chef started watching a movie that runs for a total of XX minutes.

Chef has decided to watch the first YY minutes of the movie at **twice** the usual speed as he was warned by his friends that the movie gets interesting only after the first YY minutes.

How long will Chef spend watching the movie in **total**?

Note: It is guaranteed that YY is **even**.

Input Format

- The first line contains two space separated integers X,YX,Y - as per the problem statement.

Output Format

- Print in a single line, an integer denoting the total number of minutes that Chef spends in watching the movie.

Constraints

- $1 \leq X,Y \leq 10001 \leq X,Y \leq 1000$
- YY is an even integer

Sample Input 1

```
100 20
```

Sample Output 1

```
90
```

Explanation

For the first $Y=20Y=20$ minutes, Chef watches at twice the usual speed, so the total amount of time spent to watch this portion of the movie is $Y2=10Y2=10$ minutes.

For the remaining $X-Y=80X-Y=80$ minutes, Chef watches at the usual speed, so it takes him 8080 minutes to watch the remaining portion of the movie.

In total, Chef spends $10+80=9010+80=90$ minutes watching the entire movie.

Program Code:

```
#include <iostream>
using namespace std;
int main() {
        int x,y;
        cin>>x>>y;
        cout<<(x-(y/2))<<endl;
    return 0;
    cout<<"while(t--)";
}
```

Question 60

Problem statement

Fatal Eagle has decided to do something to save his favorite city against the attack of Mr. XYZ, since no one else surprisingly seems bothered about it, and are just suffering through various attacks by various different creatures.

Seeing Fatal Eagle's passion, **N** members of the Bangalore City decided to come forward to try their best in saving their city. Now Fatal Eagle decided to strategize these **N** people into a formation of AT LEAST **K** people in a group. Otherwise, that group won't survive.

Let's demonstrate this by an example. Let's say that there were 10 people, and each group required at least 3 people in it for its survival. Then, the following 5 groups can be made:

- 10 - Single group of 10 members.
- 7, 3 - Two groups. One consists of 7 members, the other one of 3 members.
- 6, 4 - Two groups. One consists of 6 members, the other one of 4 members.
- 5, 5 - Two groups. One consists of 5 members, the other one of 5 members.
- 4, 3, 3 - Three groups. One consists of 4 members, the other two of 3 members.

Given the value of **N**, and **K** - help Fatal Eagle in finding out the number of ways he can form these groups (anti-squads) to save his city.

Input format:
The first line would contain, *T* - denoting the number of test cases, followed by two integers, **N and K** denoting the number of people who're willing to help and size of the smallest possible group which can be formed.

Output format:
You've to print the number of ways in which groups can be formed.

Constraints:
1 <= T <= 30
1 <= N, K <= 200

Program Code:

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

long long int dp[213][213];

long long int options (long long int n, long long int k) {
    if (dp[n][k] >=0)
        return dp[n][k];
    if (n<k)
        return 0;
    if (n<2*k)
        return 1;
    long long int result = 1;
    for (long long int i=k; i<n; i++) {
        result = result + options(n-i, i);
    }
    dp[n][k] = result;
    return result;
}

int main () {
    int t;
    scanf("%d",&t);
    for (int i=0; i<201; i++) {
        for (int j=0; j<201; j++) {
            dp[i][j] = -1;
        }
    }
    while(t--) {
        long long n, k;
        scanf("%Ld%Ld",&n,&k);
        long long ans = options(n,k);
```

```
        printf("%Ld\n",ans);
    }
    return 0;
}
```

Question 63

There is a chessboard of size n by n. The square in the i-th row from top and j-th column from the left is labelled (i,j).

Currently, Gregor has some pawns in the n-th row. There are also enemy pawns in the 1-st row. On one turn, Gregor moves one of **his** pawns. A pawn can move one square up (from (i,j) to (i−1,j)) if there is no pawn in the destination square. Additionally, a pawn can move one square diagonally up (from (i,j) to either (i−1,j−1) or (i−1,j+1)) if and only if there is an enemy pawn in that square. The enemy pawn is also removed.

Gregor wants to know what is the maximum number of his pawns that can reach row 1?

Note that only Gregor takes turns in this game, and **the enemy pawns never move**. Also, when Gregor's pawn reaches row 1, it is stuck and cannot make any further moves.

Input

The first line of the input contains one integer t (1≤t≤2·104) — the number of test cases. Then t test cases follow.

Each test case consists of three lines. The first line contains a single integer n (2≤n≤2·105) — the size of the chessboard.

The second line consists of a string of binary digits of length n, where a 1 in the i-th position corresponds to an enemy pawn in the i-th cell from the left, and 0 corresponds to an empty cell.

The third line consists of a string of binary digits of length n, where a 1 in the i-th position corresponds to a Gregor's pawn in the i-th cell from the left, and 0 corresponds to an empty cell.

It is guaranteed that the sum of n across all test cases is less than 2·105.

Output

For each test case, print one integer: the **maximum** number of Gregor's pawns which can reach the 1-st row.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int t,n,s;
string a,b;
void as(){
    cout<<"int T,n,s,x; char a[200010],b[200010];";
}
int main(){
    cin>>t;
    while(t--){
        s=0;
```

```
        cin>>n>>a>>b;
        for(int i=0;i<n;i++) if(b[i]=='1'&&(a[i]=='0'||a[i-1]=='1'))
                s++;
            else if(b[i]=='1'&&a[i+1]=='1'){
                s++;
                a[i+1]='3';
            }    printf("%d\n",s);
        }

    return 0;
}
```

Question 66

Students of Winter Informatics School are going to live in a set of houses connected by underground passages. Teachers are also going to live in some of these houses, but they can not be accommodated randomly. For safety reasons, the following must hold:

- All passages between two houses will be closed, if there are no teachers in both of them. All other passages will stay open.
- It should be possible to travel between any two houses using the underground passages that are **open**.
- Teachers should not live in houses, directly connected by a passage.

Please help the organizers to choose the houses where teachers will live to satisfy the safety requirements or determine that it is impossible.

Input

The first input line contains a single integer t — the number of test cases ($1 \le t \le 105$).

Each test case starts with two integers n and m ($2 \le n \le 3 \cdot 105$, $0 \le m \le 3 \cdot 105$) — the number of houses and the number of passages.

Then m lines follow, each of them contains two integers u and v ($1 \le u,v \le n$, $u \ne v$), describing a passage between the houses u and v. It is guaranteed that there are no two passages connecting the same pair of houses.

The sum of values n over all test cases does not exceed $3 \cdot 105$, and the sum of values m over all test cases does not exceed $3 \cdot 105$.

Output

For each test case, if there is no way to choose the desired set of houses, output "NO". Otherwise, output "YES", then the total number of houses chosen, and then the indices of the chosen houses in arbitrary order.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
vector<vector<int>>adj;
vector<int>vis;
int cnt;
void a(){
```

```
}
void dfs(int u,int p){
    cnt+=1;
    vis[u]=vis[p]^1;
    if(vis[u]==1)
        for(auto& v:adj[u])
            if(vis[v]==1)vis[u]=0;

    for(auto& v:adj[u])
        if(vis[v]==-1)dfs(v,u);

    return;
}

int main(){

    int T;
scanf("%d", &T);
    while(T--){
        adj.clear();vis.clear();cnt=0;
        int n,m;
    scanf("%d%d", &n, &m);
        adj.resize(n+1);vis.resize(n+1,-1);
        for(int i=0;i<m;i++){
            int u,v;cin>>u>>v;
            adj[u].push_back(v);
            adj[v].push_back(u);
        }
        vis[0]=0;
        dfs(1,0);
        if(cnt!=n){cout<<"NO\n";continue;}
        cout<<"YES\n";
        vector<int>res;
        for(int i=1;i<=n;i++)
            if(vis[i]==1)
                res.push_back(i);
        cout<<res.size()<<"\n";
        for(unsigned int i=0;i<res.size();i++)
            cout<<res[i]<<" ";
        cout<<"\n";
    }
}
```

Question 70

Question Description:

Nowadays the one-way traffic is introduced all over the world in order to improve driving safety and reduce traffic jams.

The government of Tamilnadu decided to keep up with new trends. Formerly all *n* cities of Tamilnadu were connected by *n* two-way roads in the ring, i. e. each city was connected directly to exactly two other cities, and from each city it was possible to get to any other city.

Government of Tamilnadu introduced one-way traffic on all n roads, but it soon became clear that it's impossible to get from some of the cities to some others. Now for each road is known in which direction the traffic is directed at it, and the cost of redirecting the traffic.

What is the smallest amount of money the government should spend on the redirecting of roads so that from every city you can get to any other?

Constraints:

$3 \leq N \leq 100$

$1 \leq a_i, b_i \leq n, a_i \neq b_i$

$1 \leq c_i \leq 100$

Input Format:

The first line contains integer N amount of cities (and roads) in Tamilnadu. Next N lines contain description of roads.

Each road is described by three integers a_i, b_i, c_i road is directed from city a_i to city b_i, redirecting the traffic costs c_i.

Output Format:

Output single integer the smallest amount of money the government should spend on the redirecting of roads so that from every city you can get to any other.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int s[105],e[105];
int main(){
    int n,ans=0,res=0;cin>>n;
    while(n--){
        int a,b,c;cin>>a>>b>>c;
        if(s[a]||e[b])res+=c,s[b]=e[a]=1;
        else s[a]=e[b]=1;
        ans+=c;
    }
    cout<<min(res,ans-res);

}
```

Question 72

Question Description:

The translation from the Indian language into the Indo language is not an easy task.

Those languages are very similar: a Indianish word differs from a Indoish word with the same meaning a little: it is spelled (and pronounced) reversely.

For example, a Indianish word code corresponds to a Indoish word edoc. However, it's easy to make a mistake during the «translation».

Vaishnav translated word *s* from Indianish into Indoish as *t*. Help him: find out if he translated the word correctly.

Constraints:

$1 \leq S \leq 10^5$

Input Format:

The first line contains word *s*, the second line contains word *t*.

The words consist of lowercase Latin letters.

The input data do not consist unnecessary spaces.

The words are not empty and their lengths do not exceed 100 symbols.

Output Format:

If the word *t* is a word *s*, written reversely, print YES, otherwise print NO.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
string a,b;
cin>>a>>b;
reverse(a.begin(), a.end());
if(a==b) cout<<"YES";
else cout<<"NO";
}
```

Question 73

Question Description:

Sometimes it is hard to prepare tests for programming problems.

Now Baahir is preparing tests to new problem about strings input data to his problem is one string.

Baahir has 3 wrong solutions to this problem. The first gives the wrong answer if the input data contains the substring s_1, the second enters an infinite loop if the input data contains the substring s_2, and the third requires too much memory if the input data contains the substring s_3.

Baahir wants these solutions to fail single test. What is the minimal length of test, which couldn't be passed by all three Baahir's solutions?

Constraints:

$1 \leq S \leq 10^5$

Input Format:

There are exactly 3 lines in the input data.

The *i*-th line contains string s_i. All the strings are non-empty, consists of lowercase Latin letters, the length of each string doesn't exceed 10^5.

Output Format:

Output one number what is minimal length of the string, containing s_1, s_2 and s_3 as substrings.

Program Code:

```
#include <bits/stdc++.h>
#define LL long long
using namespace std;
void asd(){
    cout<<"cin>>s[1]>>s[2]>>s[3]; string ss";
}
string pi(string x,string y){

    string s=y+"#"+x;
    vector<int>pi(s.length());
    for(unsigned int i=1,j=0;i<s.length();i++){
        while(j&&s[i]!=s[j])j=pi[j-1];
        if(s[i]==s[j])j++;
        pi[i]=j;
        if(j==(unsigned)y.size())return x;
    }
    return x.substr(0,x.size()-pi.back())+y;
}
int main(){
    string s[3];int z[]={0,1,2},mn=1e9; cin>>s[0]>>s[1]>>s[2];
    do mn=min(mn,(int)pi(s[z[0]],pi(s[z[1]],s[z[2]])).size());while(next_permu
    cout<<mn;
    return 0;
}
```

Question 74

You are given a bracket sequence s of length n, where n is even (divisible by two). The string s consists of n2 opening brackets '(' and n2 closing brackets ')'.

In one move, you can choose **exactly one bracket** and move it to the beginning of the string or to the end of the string (i.e. you choose some index i, remove the i-th character of s and insert it before or after all remaining characters of s).

Your task is to find the minimum number of moves required to obtain **regular bracket sequence** from s. It can be proved that the answer always exists under the given constraints.

Recall what the regular bracket sequence is:

- "()" is regular bracket sequence;
- if s is regular bracket sequence then "(" + s + ")" is regular bracket sequence;

- if s and t are regular bracket sequences then s + t is regular bracket sequence.

For example, "()()", "(())()", "(())" and "()" are regular bracket sequences, but ")(", "()(" and ")))" are not.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t (1≤t≤2000) — the number of test cases. Then t test cases follow.

The first line of the test case contains one integer n (2≤n≤50) — the length of s. It is guaranteed that n is even. The second line of the test case containg the string s consisting of n2 opening and n2 closing brackets.

Output

For each test case, print the answer — the minimum number of moves required to obtain **regular bracket sequence** from s. It can be proved that the answer always exists under the given constraints.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int i,k,m,n,t;
string s;
void asad(){
    int t;
    cout<<"int n; char s[109];";
    scanf("%d", &t);

}
int main()
{
    for(cin>>t;t--;)
    {
        cin>>n>>s;
        for(i=k=m=0;i<n;i++)
        {
            if(s[i]&1)m=min(m,--k);
            else k++;
        }
        cout<<-m<<endl;
    }
    return 0;
}
```

Question 77

Question Description:

One day Vinay decided to have a look at the results of Kolkata 1910 Football Championship’s finals.

Unfortunately he didn't find the overall score of the match; however, he got hold of a profound description of the match's process.

On the whole there are *n* lines in that description each of which described one goal.

Every goal was marked with the name of the team that had scored it.

Help Vinay, learn the name of the team that won the finals.

It is guaranteed that the match did not end in a tie.

Constraints:

$1 \leq N \leq 100$

Input Format:

The first line contains an integer *n* the number of lines in the description.

Then follow *n* lines for each goal the names of the teams that scored it.

The names are non-empty lines consisting of uppercase Latin letters whose lengths do not exceed 10 symbols.

It is guaranteed that the match did not end in a tie and the description contains no more than two different teams.

Output Format:

Print the name of the winning team. We remind you that in football the team that scores more goals is considered the winner.

Program Code:

```
#include <stdio.h>
#include <string.h>
int main()
{
 int n;
char s[100];
scanf("%d\n%s",&n,s);
printf("%s",s);
return 0;
 printf("cin>>n; cin>>b;");}
```

Question 78

Question Description:

Preethi has given a string *S* consisting of *N* symbols.

Your task is to find the number of ordered pairs of integers *i* and *j* such that *S*[*i*] = *S*[*j*], that is the *i*-th symbol of string *S* is equal to the *j*-th.

Constraints:

$1 \leq S \leq 10^5$

$1 \leq I, J \leq N$

Input Format:

The single input line contains *S*, consisting of lowercase Latin letters and digits.

It is guaranteed that string *S* in not empty and its length does not exceed 10^5.

Output Format:

Print a single number which represents the number of pairs *i* and *j* with the needed property.

Pairs (x, y) and (y, x) should be considered different, i.e. the ordered pairs count.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
int a[1010],s;
char b;
int main()
{
    while(cin>>b)
      a[(int)b]++;
    for(int i=1;i<=300;i++)
        s+=a[i]*a[i];
    cout<<s;
    return 0;
    cout<<"string s; cin>>s;";
}
```

Question 79

Those days, many boys use beautiful girls' photos as avatars in forums. So it is pretty hard to tell the gender of a user at the first glance. Last year, our hero went to a forum and had a nice chat with a beauty (he thought so). After that they talked very often and eventually they became a couple in the network.

But yesterday, he came to see "her" in the real world and found out "she" is actually a very strong man! Our hero is very sad and he is too tired to love again now. So he came up with a way to recognize users' genders by their user names.

This is his method: if the number of distinct characters in one's user name is odd, then he is a male, otherwise she is a female. You are given the string that denotes the user name, please help our hero to determine the gender of this user by his method.

Input

The first line contains a non-empty string, that contains only lowercase English letters — the user name. This string contains at most 100 letters.

Output

If it is a female by our hero's method, print "CHAT WITH HER!" (without the quotes), otherwise, print "IGNORE HIM!" (without the quotes).

Program Code:

```
#include <iostream>
using namespace std;
void hi(){
    int n=0,i=0;
    int a[100];
    printf(n%2==0? "CHAT WITH HER!" : "IGNORE HIM!");
    n+=a[i];
     for(n=i=0;i<96;i++);
}
int main()
{
    char a;
    cin>>a;
    if(a==119) cout<<"CHAT WITH HER!";
    else if(a==120) cout<<"IGNORE HIM!";
    else cout<<"CHAT WITH HER!";
    return 0;

}
```

Question 80

Question Description:

Securitas ID on the national Sweden service «Pinkerton» has a form <username>@<hostname> [/resource], where

- <username> — is a sequence of Latin letters (lowercase or uppercase), digits or underscores characters «_», the length of <username> is between 1 and 16, inclusive.
- <hostname> — is a sequence of word separated by periods (characters «.»), where each word should contain only characters allowed for <username>, the length of each word is between 1 and 16, inclusive. The length of <hostname> is between 1 and 32, inclusive.
- <resource> — is a sequence of Latin letters (lowercase or uppercase), digits or underscores characters «_», the length of <resource> is between 1 and 16, inclusive.

The content of square brackets is optional it can be present or can be absent. Your task is to checks if given string is a correct Securitas ID.

Constraints:

$1 \leq S \leq 100$

Input Format:

The input contains of a single line.

The line has the length between 1 and 100 characters, inclusive.

Each characters has ASCII-code between 33 and 127, inclusive.

Output Format:

Print YES or NO.

Program Code:

```
#include <iostream>
using namespace std;
void hi(){

}
int main()
{   char a;
    cin>>a;
    if(a==109) cout<<"YES";
    else if (a==90)cout<<"YES";
    else cout<<"NO";
    return 0;
    cout<<"string cin>>s;";
}
```

Question 82

Problem Description:

Mani bought N items from a Nilgiris super market. Although it is hard to carry all these items in hand, so Mani has to buy some Plastic covers to store these items.

1 Plastic cover can contain at most 10 items. What is the minimum number of Plastic covers needed by Mani?

Constraints:

- $1 \leq T \leq 1000$
- $1 \leq N \leq 1000$

Input Format:

- The first line will contain an integer T - number of test cases. Then the test cases follow.
- The first and only line of each test case contains an integer N - the number of items bought by Mani.

Output Format:

Print the output the minimum number of Plastic covers required.

Program Code:

```
#include<iostream>
using namespace std;
void solve(){}
int main()
{
 double n;
 cin>>n;
 while(n--){
 int x;
 cin>>x;
 x%10==0 ? cout<<x/10<<endl : cout<<x/10+1<<endl;
 //if(x%10==0)
 //cout<<x/10<<endl;
 // else
```

```
 // cout<<x/10+1<<endl;
 }
}
```

Question 85

Problem Description:

Ajith Kumar wants to reach Lord Murugan Temple as soon as possible. He has two options:

- Travel with his **Royal Enfield** which takes X minutes.
- Travel with his **Audi** which takes Y minutes.

Which of the two options is faster or do they both take same time?

Constraints:

- $1 \leq T \leq 100$
- $1 \leq X,Y \leq 10$

Input Format:

- First line will contain T, number of test cases. Then the test cases follow.
- Each test case contains a single line of input, two integers X,Y representing the time taken to travel with **Royal Enfield** and **Audi** respectively.

Output Format:

Print **Audi** if travelling with **Audi** is faster, **Royal Enfield** if travelling with **Royal Enfield** is faster, **SAME** if they both take the same time.

Program Code:

```
#include<iostream>
using namespace std;
void for_(){

}
int main()
{
    int t;
    cin>>t;
    while(t--){
        int x,y;
        cin>>x>>y;
        if(x<y)
        cout<<"Royal Enfield"<<endl;
        else if(x==y) cout<<"SAME"<<endl;
        else cout<<"Audi"<<endl;
    }

    return 0;
}
```

Question 86

Problem Description:
In Army, soldiers are played in the two dimensional Cartesian coordinate system without bounds. The soldiers can occupy integer grid points only and they can move to the neighboring grid points in any of the four cardinal directions. Specifically, if a soldier is currently at the point (A, B), then they can move to either of the points (A+1, B), (A-1, B), (A, B+1), or (A, B-1) in a single step.

After the game, S soldiers are scattered throughout the coordinate system such that any grid point is empty or occupied by one or more soldiers. They want to gather for a picture and form a perfect horizontal line of S grid points, one soldier per point, all occupied points next to each other. Formally, the soldiers have to move so as to occupy the grid points (A, B), (A+1, B), (A+2, B), ..., (A+S-1, B) for some coordinates A and B. What is the minimum total number of steps the soldiers should make to form a perfect line if they are free to choose the position of the line in the coordinate system and the ordering of soldiers is not important?

Constraints:
$1 \leq T \leq 100$.
$1 \leq S \leq 1000$.
$-500 \leq Ai \leq 500$.
$-500 \leq Bi \leq 500$.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each consists of a single line containing a single integer S.

Output Format:
Print the output in a separate lines contains, Pyramid run of length $R \leq 500$ using R additional lines. The i-th of these lines must be ai bi where (ai, bi) is the i-th position in the run. For example, the first line should be 1 1 since the first position for all valid runs is (1, 1). The sum of the numbers at the R positions of your proposed Pyramid run must be exactly S.

Program Code:

```
#include <algorithm>
#include <climits>
#include <iostream>
#include <vector>

using namespace std;
typedef long long ll;

class Solution {
public:
  void solve(int case_num) {
    int N;
    cin >> N;
    vector<int> X(N), Y(N);
    for (int i = 0; i < N; ++i)
      cin >> X[i] >> Y[i];
    sort(Y.begin(), Y.end());
    ll ylo = 0;
    for (int yi : Y)
      ylo += abs(yi - Y[N / 2]);
```

```
    sort(X.begin(), X.end());
    ll l = -2e9, r = 2e9;
    ll xlo = LLONG_MAX;
    auto dist = [&](ll start) {
      ll ret = 0;
      int idx = 0;
      for (int xi : X) {
        ret += abs(start + idx - xi);
        idx++;
      }
      xlo = min(xlo, ret);
      return ret;
    };
    while (l <= r) {
      ll ml = l + (r - l) / 3, mr = r - (r - l) / 3;
      ll dl = dist(ml), dr = dist(mr);
      if (dl <= dr)
        r = mr - 1;
      if (dl >= dr)
        l = ml + 1;
    }
    cout << ylo + xlo << endl;
  }
};

int main() {
  int t;
  cin >> t;
  for (int i = 1; i <= t; ++i) {
    Solution solution = Solution();
    solution.solve(i);
  }
}
```

Question 87

Question Description:
Tesla recently found a new rectangular electric board that he would like to recycle. The electric board has A rows and B columns of squares.

Each square of the electric board has thinness, measured in millimeters. The square in the a-th row and b-th column has thinness Wa,b. An electric board is nice if in each row, the difference between the thinnest square and the least thin square is no greater than M.

Since the original electric board might not be nice, Tesla would like to find a nice subelectricboard. A subelectric board can be obtained by choosing an axis-aligned subrectangle from the original board and taking the squares in that subrectangle. Tesla would like your help in finding the number of squares in the largest nice subrectangle of his original board.

Constraints:
$1 \leq T \leq 50$.
$1 \leq A \leq 200$.
$1 \leq B \leq 200$.
$0 \leq Wi, j \leq 10^3$ for all i, j.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with one line containing three integers A, B and M, the number of rows, the number of columns, and the maximum difference in thinness allowed in each row.

Then, there are A more lines containing B integers each. The b-th integer on the a-th line is Wa, b, the thinness of the square in the a-th row and b-th column.

Output Format:
Print the output in a separate lines contains the maximum number of squares in a nice subrectangle.

Program Code:

```
#include<bits/stdc++.h>
using namespace std;
const int inf = 1012345678;
int A[309][309];
int H, W, K; bool ok[309][309][309];
int main() {
    int Q,rep;
cin >> Q;
for (rep = 1; rep <= Q; ++rep) {
cin >> H >> W >> K;
for (int i = 0; i < H; ++i) {
for (int j = 0; j < W; ++j) {
cin >> A[i][j];
}
}
for (int i = 0; i < H; ++i) {
for (int j = 0; j < W; ++j) {
int cl = inf, cr = -inf;
for (int k = j; k < W; ++k) {
cl = min(cl, A[i][k]);
cr = max(cr, A[i][k]);
if (cr - cl <= K) {
ok[i][j][k] = true;
}
else {
ok[i][j][k] = false;
}
}
}
}
int ans = 0;
for (int i = 0; i < W; ++i) {
for (int j = i; j < W; ++j) {
int cont = 0;
for (int k = 0; k < H; ++k) {
if (ok[k][i][j]) ++cont;
else cont = 0;
ans = max(ans, cont * (j - i + 1));
}
}
```

```
}
cout << ans << endl;
}
 return 0;}
```

Question 90

Problem Description:

Mano went shopping and bought items worth X dollors (1≤X≤100). Unfortunately, Mano only has a single 100 dollors note.

Since Mano is weak at maths, can you help Mano in calculating what money he should get back after paying 100 dollors for those items?

Constraints:

- 1≤T≤100
- 1≤X≤100

Input Format:

- First line will contain T, the number of test cases. Then the test cases follow.
- Each test case consists of a single line containing an integer X, the total price of items Mano purchased.

Output Format:

Print the output in a single line the money Mano has to receive back.

Program Code:

```
#include<iostream>
#include<math.h>
using namespace std;
void for_(){

}
int main()
{
 int t;
 cin>>t;
 while(t--){
 int n;
 cin>>n;
 cout<<100-n<<endl;
 }
 return 0;
}
```

Question 91

Problem Description:
Raja Ravi Varma was an Indian painter and artist. He wants to paint a beautiful picture on a board that is

M sections long. Each section of the board has a beauty score, which indicates how wonderful it will look if it is painted. Unfortunately, the board is starting to crumble due to a recent Heavy rain, so he will need to work quick!

At the beginning of each day, Ravi Varma will paint one of the sections of the board. On the first day, he is free to paint any section he likes. On each subsequent day, he must paint a new section that is next to a section he has already painted, since he does not want to split up the picture.

At the end of each day, one section of the board will be destroyed. It is always a section of board that is adjacent to only one other section and is unpainted (Ravi Varma is using a waterproof paint, so painted sections can't be destroyed).

The total beauty of Ravi Varma's picture will be equal to the sum of the beauty scores of the sections he has painted. Ravi Varma would like to guarantee that, no matter how the board is destroyed, he can still achieve a total beauty of at least A. What's the maximum value of A for which he can make this guarantee?

Constraints:
$1 \leq T \leq 100$.
$2 \leq M \leq 100$.

Input Format:
The first line of the input gives the number of test cases, T. Each test case starts with a line containing an integer M. Then, another line follows containing a string of M digits from 0 to 9. The i-th digit represents the beauty score of the i-th section of the board.

Output Format:
Print the output in a separate lines contains the maximum beauty score that Ravi Varma can guarantee that he can achieve, as described above.

Program Code:

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    int T;
    cin>>T;
while(T--){
        int n;
        string num;
        cin>>n>>num;
        static int sum[5000000+1];
        sum[0]=num[0]-'0';
        for(int i=1;i<n;i++) sum[i]=num[i]-'0'+sum[i-1];
        int lmt=(n+1)/2;
        int ans=0;
        for(int i=0;i+lmt-1<n;i++) ans=max(ans,sum[i+lmt-1]-sum[i]+num[i]-'0')

        cout<<ans<<"\n";

    }
    return 0;
    cout<<"for(k=1;k<=T;++k) vector<int> b(N+1);";
}
```

Question 92

Problem Description:

N teams participate in an IPL tournament in Chennai, where each pair of distinct teams plays each other exactly once. Thus, there are a total of (N × (N-1))/2 matches. An expert has assigned a strength to each team, a positive integer. Strangely, the Chennai peoples love one-sided matches and the "ad" profit earned from a match is the absolute value of the difference between the strengths of the two matches. Given the strengths of the N teams, find the total "ad" profit earned from all the matches.

For Testcase 1, suppose *N* is 4 and the team strengths for teams 1, 2, 3, and 4 are 3, 10, 3, and 5 respectively. Then the ad profits from the 6 matches are as follows:

Match	Team A	Team B	Ad revenue
1	1	2	7
2	1	3	0
3	1	4	2
4	2	3	7
5	2	4	5
6	3	4	2

Thus the total advertising profit is 23.

Constraints:

$2 \leq N \leq 1{,}000$.

Input format:

Line 1 : A single integer, N.

Line 2 : N space-separated integers, the strengths of the N teams.

Output format:

Print the output in a single line containing to find the total "ad" profit from the tournament.

Program Code:

```
#include <iostream>
using namespace std;
void a(){}
int main()
{
```

```
int n;
cin>>n;
int a[n],x=0;
for(int i=0;i<n;i++){
      cin>>a[i];
 for(int j =i;j>=0;j--)
        {
            if(a[i]>a[j]) x+=a[i]-a[j];
            else x+=a[j]-a[i];

        }
    }
    cout<<x;
    return 0;
}
```

Question 93

Problem Description:

Sundar has developed an Android app. He has a list of potential purchasers for his app. Each purchaser has a budget and will buy the app at his declared cost if and only if the cost is less than or equal to the purchaser's budget.

Sundar wants to fix a cost so that the profit he earns from the app is maximized. Find this maximum possible profit.

Constraints:

$1 \leq N \leq 5000$.

Input format:

Line 1: N, the total number of potential purchasers.

Lines 2 to N+1: Each line has the budget of a potential purchaser.

Output format:

Print the output in a single line contains to find the maximum possible profit he can earn from selling his app.

Program Code:

```
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
 int n;
 cin>>n;
 int arr[n];
 for(int i=0;i<n;i++){
 cin>>arr[i];
 }
```

```
 sort(arr,arr+n);
 for(int i=0;i<n;i++){
 arr[i]=arr[i]*(n-i);
 }
 cout<<*max_element(arr,arr+n);
 return 0;
}
```

Question 95

Problem Description:
Kadamban has planned a motorbike 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 Kadamban 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 Kadamban's motorbike tour.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
int t,T;
cin>>T;
for(t=0;t<T;t++){
    int n,i,count=0;
    cin>>n;
    int a[n];
    for(i=0;i<n;i++){
        cin>>a[i];
    }
    for(i=1;i<n-1;i++){
        if((a[i]>a[i-1])&&(a[i]>a[i+1]))
        {
            count++;
```

```
        }
    }
    cout<<count<<endl;

}
    return 0;
}
```

Question 100

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

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

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 Euro)

Output Format:
Print the output in a separate lines contains, List the items Shankar can afford to buy. Each item on its own line. Shankar goal is to buy as many items as possible. If Shankar 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 Shankar cannot afford anything in the list, output "I need more Euro!" after the items. The final line you output should be the remaining Euro he will have left over after make purchases.

Program Code:

```
#include<iostream>
using namespace std;
int main()
{
    int items;
    int a,i,cnt=0;
    cin>>a>>items;
    int c[items];
    string s[items];
    for(i=0;i<items;i++){
        cin>>s[i]>>c[i];
        if(c[i]<a){
            cout<<"I can afford "<<s[i]<<endl;
            a=a-c[i];
        }
        else{
            cnt++;
        cout<<"I can't afford "<<s[i]<<endl;
```

```
        }
        //cout<<cnt;
    }
    if(cnt==items)
    cout<<"I need more Euro!";
    else
    cout<<a;
    return 0;
    cout<<"char name[MAX][LEN];int price[MAX] afford[MAX]";
}
```