**MARATONES DE PROGRAMACIÓN**

**ACIS- REDIS / ACM-ICPC**

**Cartilla de ejercicios resueltos, algoritmos de solución y ejemplos**

**Grupo programación competitiva**

**Universidad ECCI**

**Ingeniería de sistemas**

**Tecnología de desarrollo informático**

**2015 - 2019**

**Contenido**

-----------------------------------------------------------------------------------------------------------------------------------

[**AUTOMATIC POETRY (String)** 7](#_Toc2500730)

[**MINIMUN LAND PRICES (Math)** 8](#_Toc2500731)

[**OUTWITTING THE WEIGHING MACHINE (Math)** 8](#_Toc2500732)

[**THE COLLATZ SECUENCE ( Math)** 9](#_Toc2500733)

[**LONGEST COMMON SUBSEQUENCE (String)** 10](#_Toc2500734)

[**CUTTING PIZZA (Math)** 11](#_Toc2500735)

[**SUB EXPESION COUNTING C++ (String)** 11](#_Toc2500736)

[**EXPANDING RODS C++ (Math)** 12](#_Toc2500737)

[**EXPANDING RODS JAVA (Math)** 13](#_Toc2500738)

[**FEYNMAN (Geometry)** 13](#_Toc2500739)

[**I LOVE PIZZA C++ (String)** 14](#_Toc2500740)

[**PEANOLAND CONTACTING GAUSSLAND (Math)** 15](#_Toc2500741)

[**STERN-BROCOT TREE (Tree)** 17](#_Toc2500742)

[**JUDGING TROUBLES (String)** 17](#_Toc2500743)

[**TRIANGLE WAVE (String)** 18](#_Toc2500744)

[**BYE-BYE CAKES (Math)** 19](#_Toc2500745)

[**HASHMAT THE BRAVE WARRIOR (Math)** 20](#_Toc2500746)

[**NEXT PALINDROMIC NUMBERS (Big numbers)** 20](#_Toc2500747)

[**ECOLOGICAL PREMIUM (Math)** 21](#_Toc2500748)

[**JUST THE FACTS ( Big Numbers)** 22](#_Toc2500749)

[**BUSINESS CENTER (Math)** 22](#_Toc2500750)

[**ASCII AREA (Geometry)** 23](#_Toc2500751)

[**CANDY ( Dynamic )** 24](#_Toc2500752)

[**LUCKY THIEF C++ (Math)** 25](#_Toc2500753)

[**AUTOMATIC ANSWER (Math)** 25](#_Toc2500754)

[**THE DECODER C++ (String)** 26](#_Toc2500755)

[**KINDERGARTEN COUNTING GAME (String)** 26](#_Toc2500756)

[**ALARM CLOCK (Math)** 27](#_Toc2500757)

[**A MULTIPLICATION GAME (Pattern)** 27](#_Toc2500758)

[**MAGIC FORMULA (Math)** 28](#_Toc2500759)

[**CARMICHAEL NUMBERS (Math )** 29](#_Toc2500760)

[**COWS AND CARS (Probability)** 29](#_Toc2500761)

[**SEARCHING FOR NESSY (Geometry)** 30](#_Toc2500762)

[**BIG CHOCOLATE (Geometry)** 30](#_Toc2500763)

[**FACTSTONE BENCHMARK (Math)** 30](#_Toc2500764)

[**BILLIARD (Geometry)** 31](#_Toc2500765)

[**KNIGHT MOVES (Graph)** 31](#_Toc2500766)

[**ORDERING TASKS (Graph)** 32](#_Toc2500767)

[**TRAFFIC VOLUME (Math)** 33](#_Toc2500768)

[**CHESS (math)** 34](#_Toc2500769)

[**ELECTRICITY (Math)** 34](#_Toc2500770)

[**BENDER B. RODRÍGUEZ PROBLEM (Geometry)** 36](#_Toc2500771)

[**AUTOMATE THE GRADES (Math)** 37](#_Toc2500772)

[**SQUARE ROOT (Big Numbers)** 38](#_Toc2500773)

[**ONE-TWO-THREE (String)** 39](#_Toc2500774)

[**FIBONACCI WORDS (String)** 39](#_Toc2500775)

[**SETTLES OF CATAN ( Graph)** 41](#_Toc2500776)

[**SMOOTH FACTOR (Math)** 42](#_Toc2500777)

[**AUDIOPHOBIA (Graph)** 43](#_Toc2500778)

[**TENNIS ROUNDS (Math)** 45](#_Toc2500779)

[**PHILIP J. FRY PROBLEM (Queqe)** 45](#_Toc2500780)

[**SCORE (String)** 46](#_Toc2500781)

[**POWER OF CRYPTOGRAPHY (Big numbers)** 47](#_Toc2500782)

[**DECODE THE TAPE (String)** 48](#_Toc2500783)

[**PRODUCT (Big numbers)** 48](#_Toc2500784)

[**INTEGER INQUIRY (Big numbers)** 49](#_Toc2500785)

[**OVERFLOW (String)** 49](#_Toc2500786)

[**EXPONENTIATION (Big numbers)** 50](#_Toc2500787)

[**IF WE WERE A CHILD AGAIN (Big numbers)** 51](#_Toc2500788)

[**DEGREES OF SEPARATION (Graph)** 51](#_Toc2500789)

[**NUMERICAL SURPRISES (Big numbers)** 53](#_Toc2500790)

[**POWERS ET AL. (Modular math)** 53](#_Toc2500791)

[**ANDY'S FIRST DICTIONARY (String)** 53](#_Toc2500792)

[**BOX OF BRICKS (Math)** 54](#_Toc2500793)

[**CRAZY TEA PARTY (Math)** 55](#_Toc2500794)

[**GCD LCM (Math)** 56](#_Toc2500795)

[**DIGIT COUNTING (String)** 56](#_Toc2500796)

[**MOLAR MASS (String)** 57](#_Toc2500797)

[**WHAT'S CRYPTANALYSIS? (String-Sort)** 58](#_Toc2500798)

[**EDITOR (Dynamic)** 59](#_Toc2500799)

[**SKEW BINARY (Math)** 59](#_Toc2500800)

[**HE IS OFFSIDE (Math)** 60](#_Toc2500801)

[**SHRINKING POLYGONS (Geometry)** 61](#_Toc2500802)

[**IDENTIFYING TEA (Math)** 62](#_Toc2500803)

[**HANGMAN JUDGE (String)** 63](#_Toc2500804)

[**D AS IN DAEDALUS (Math)** 63](#_Toc2500805)

[**ACIS, A Contagious vIruS (Geometry)** 64](#_Toc2500806)

[**FUNNY CARDIOLOGIST (Geometry)** 65](#_Toc2500807)

[**ECLIPSING GIANIK STAR (Geometry )** 66](#_Toc2500808)

[**ODD OR EVEN (Math)** 67](#_Toc2500809)

[**THE MONKEY AND THE OILED BAMBOO (Math)** 67](#_Toc2500810)

[**BIG MOD (Big Numbers)** 68](#_Toc2500811)

[**HUMIDEX (String/Math)** 69](#_Toc2500812)

[**I LOVE BIG NUMBERS (Big numbers)** 70](#_Toc2500813)

[**BABELFISH (STRING)** 70](#_Toc2500814)

[**TOUR DE FRANCE (Math)** 72](#_Toc2500815)

[**SCROLLING SIGN (String)** 72](#_Toc2500816)

[**THE HOTEL WITH INFINITE ROOMS (Math)** 73](#_Toc2500817)

[**THIS IS INTEGRATION ? (Geometry)** 74](#_Toc2500818)

[**TROUBLE WITH A PENTAGON (Geometry)** 74](#_Toc2500819)

[**SIMPLE BASE CONVERSION (Big numbers)** 75](#_Toc2500820)

[**COUNTING STARS (String)** 76](#_Toc2500821)

[**GCD (Math)** 77](#_Toc2500822)

[**SQUARE NUMBERS (Math)** 78](#_Toc2500823)

[**ROBOT INSTRUCCIONS (String)** 78](#_Toc2500824)

[**HANOI TOWER TROUBLES AGAIN! (Math)** 79](#_Toc2500825)

[**THE 3N+1 PROBLEM (Math)** 79](#_Toc2500826)

[**BASE -2 (Math)** 80](#_Toc2500827)

[**THE BUS DRIVER PROBLEM (Sort)** 81](#_Toc2500828)

[**CACHO (String)** 82](#_Toc2500829)

[**SODA SURPLER (Recursive math)** 82](#_Toc2500830)

[**BOUNDING BOX (Geometry)** 83](#_Toc2500831)

[**FORWARDING EMAILS (Graph)** 85](#_Toc2500832)

[**BEE (Math)** 86](#_Toc2500833)

[**COLA (Math)** 86](#_Toc2500834)

[**RELATIONAL OPERATOR (Math )** 87](#_Toc2500835)

[**TERNARY (Math)** 88](#_Toc2500836)

[**BRICK WALL PATTERNS (Math)** 88](#_Toc2500837)

[**BOILED EGGS (Math)** 89](#_Toc2500838)

[**COMMANDO WAR (Sorting)** 90](#_Toc2500839)

[**THE BLOCKS PROBLEM (String-Stack)** 91](#_Toc2500840)

[**ECOLOGICAL BIN PACKING (Math-graph)** 93](#_Toc2500841)

[**TEX QUOTES (String)** 95](#_Toc2500842)

[**MACHINED SURFACES (Math)** 96](#_Toc2500843)

[**ROTATING SENTENCES (String)** 96](#_Toc2500844)

[**ARTIFICIAL INTELLIGENCE? (String)** 97](#_Toc2500845)

[**THE DECODER JAVA (String)** 100](#_Toc2500846)

[**LINEAL CELULAR AUTOMATA (String)** 101](#_Toc2500847)

[**FACTORS AND FACTORIALS (Math/PrintFormat)** 102](#_Toc2500848)

[**SAFE SALUTATIONS (Math)** 104](#_Toc2500849)

[**GRAMMAR EVALUATIONS (STRING)** 105](#_Toc2500850)

[**LIST OF CONQUEST (String)** 105](#_Toc2500851)

[**BEAT THE SPREAD! (Math)** 106](#_Toc2500852)

[**RAT ATTACK (Data Structure)** 107](#_Toc2500853)

[**MACARONS (Graph)** 108](#_Toc2500854)

[**SCARECROW (String)** 111](#_Toc2500855)

[**GREY CODES (Math-bits)** 111](#_Toc2500856)

[**BE EFICCIENT (Math)** 112](#_Toc2500857)

[**EIGENSECUENCE (Graph)** 112](#_Toc2500858)

[**ARRANGE SOME MARBLES (Graph)** 113](#_Toc2500859)

[**ZEROES III (Graph)** 115](#_Toc2500860)

[**COST CUTTING (Math)** 116](#_Toc2500861)

[**MINESWEEPER (String)** 117](#_Toc2500862)

[**THE SNAIL (Math)** 119](#_Toc2500863)

[**HORROR DASH (Sort)** 119](#_Toc2500864)

[**DIVISION OF NLOGONIA (Geometric)** 120](#_Toc2500865)

[**PESKY PALINDROMES(String)** 121](#_Toc2500866)

[**COWS IN A ROW (Brute force)** 122](#_Toc2500867)

[**REQUEST FOR PROPOSAL (Math)** 123](#_Toc2500868)

[**EVENT PLANING (Math)** 124](#_Toc2500869)

[**LOANSOME CAR BUYER (Math)** 125](#_Toc2500870)

[**EXPRESIONS (Tree)** 126](#_Toc2500871)

[**PARENTHESES BALANCE (String-Queue)** 128](#_Toc2500872)

[**COLLECTING BEEPERS (Graph)** 129](#_Toc2500873)

[**ENCODER AND DECODER (String)** 130](#_Toc2500874)

[**JOKING WITH FERMAT´S LAST THEOREM (Math)** 131](#_Toc2500875)

[**N+NODN (Math)** 132](#_Toc2500876)

[**PROFESOR (Math)** 133](#_Toc2500877)

[**PUZ (Math)** 134](#_Toc2500878)

[**ROTATING DRUMS (Combinatory)** 134](#_Toc2500879)

[**SHUFFLING CARDS (Math)** 135](#_Toc2500880)

[**SCOCCER CHAMPIONSHIP (String)** 136](#_Toc2500881)

[**STRETAN (Math)** 139](#_Toc2500882)

[**TABOVI (Math)** 140](#_Toc2500883)

[**TIMSKO (Math)** 140](#_Toc2500884)

[**COUNTING CHAOS (Dynamic)** 141](#_Toc2500885)

[**BUBBLE MAPS (String)** 142](#_Toc2500886)

[**LAZY JUMPING FROG (Graph)** 145](#_Toc2500887)

[**TORNADO! (Dynamic)** 146](#_Toc2500888)

[**IS THIS INTEGRATION? (Geometry)** 147](#_Toc2500889)

[**THE TWO NOTE RAG (Brute-force)** 148](#_Toc2500890)

[**RECURSIVELY PALINDROMIC PARTITIONS (Math)** 148](#_Toc2500891)

[**ALMOST SHORTEST PATH (Graph)** 149](#_Toc2500892)

[**ALL START THREE POINT CONTEST (String)** 152](#_Toc2500893)

[**DIVISORS (Math)** 153](#_Toc2500894)

# **AUTOMATIC POETRY (String)**

import java.util.Scanner;

public class Main {

   public Main (){

       Scanner s = new Scanner (System.in);

       int n =s.nextInt();

       s.nextLine();

       for (int i = 0; i < n; i++) {

           String linea1 =s.nextLine();

           String s1,s2,s3,s4,s5;

           s1= linea1.substring(0,linea1.indexOf("<"));

           s2= linea1.substring(linea1.indexOf("<"),linea1.indexOf(">"));

           s3= linea1.substring(linea1.indexOf(">"),linea1.lastIndexOf("<"));

           s4= linea1.substring (linea1.lastIndexOf("<"),linea1.lastIndexOf(">"));

           s5= linea1.substring (linea1.lastIndexOf(">"));

           String linea2= s.nextLine();

           linea2 =linea2.replace("...", s4+s3+s2+s5);

           linea1=linea1.replace("<","");

           linea1=linea1.replace(">","");

           linea2=linea2.replace("<","");

           linea2=linea2.replace(">","");

           System.out.println(linea1);

           System.out.println(linea2);

       }

   }

    public static void main(String[] args) {

        new Main();

    }

}

# **MINIMUN LAND PRICES (Math)**

import java.util.ArrayList;

import java.util.Collections;

import java.util.Scanner;

public class Main {

   public  Main()

   {

       Scanner s = new Scanner(System.in);

       int t = s.nextInt();

       for (int i = 0; i < t; i++) {

           ArrayList<Integer> prices=new ArrayList<Integer>();

           int li= s.nextInt();

           while (li!=0){

           prices.add(li);

           li=s.nextInt();

           }

          Collections.sort(prices);

          int totalprice =0;

          int pow =1;

           for (int j = prices.size()-1; j >=0; j--) {

               totalprice+= 2\*Math.pow(prices.get(j),pow);

               pow++;

               if (totalprice>5000000)

               break;

              }

           if (totalprice>5000000){

               System.out.println("Too expensive");

           }

           else

               {System.out.println(totalprice);

                       }

           }

       }

   public static void main (String[] args)

       { new Main();}

   }

# [**OUTWITTING THE WEIGHING MACHINE**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=4709) **(Math)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main {

    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        int t = Integer.parseInt(br.readLine());

        for (int i = 0; i < t; i++)

        {

            int w []=new int[10];

            int wtotal=0;

            String arr[] = br.readLine().split(" ");

            for (int j = 0; j < 10; j++)

                {

                w[j] = Integer.parseInt(arr[j]);

                wtotal+=w[j];

                }

            wtotal/=4;

            int a,b,c,d,e;

            c=wtotal-w[0]-w[9];

            a=w[1]-c;

            e=w[8]-c;

            b=w[0]-a;

            d=w[9]-e;

            System.out.println("Case "+(i+1)+": "+a+" "+b+" "+c+" "+d+" "+e);

        }

    }

# **THE COLLATZ SECUENCE ( Math)**

#include<stdio.h>

int main()

{

long long a,numero,limite,i,j=1;

while(scanf("%lld%lld",&numero,&limite)==2)

//%lld significa conversion a long long int, cada %lld va con una variable en orden de izquierda a derecha.

//El primero con numero, el segundo con limite.

{

if(numero<0 && limite<0)

break;

a=numero;

for(i=1;i!=0;i++)

{

if(a==1)

break;

if(a%2==0)

a=a/2;

else

a=a\*3+1;

if(a>limite)

break;

}

printf("Case %lld: A = %lld, limit = %lld, number of terms = %lld\n",j,numero,limite,i);

j++;

}

}

# **LONGEST COMMON SUBSEQUENCE (String)**

import java.util.Scanner;

import static java.lang.Math.max;

class Main {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

        String first, second;

        //Puede tener un espacio y jode el next

        while (in.hasNextLine()) {

            first = in.nextLine();

            second = in.nextLine();

            System.out.println(lcs(first, second,1 ));

        }

    }

    public static int lcs(String s1, String s2, int k) {

        s1 = "#" + s1;

        s2 = "#" + s2;

        int C[][] = new int[s1.length()][s2.length()];

        for (int a = k; a < s1.length(); a++) {

            for (int b = k; b < s2.length(); b++) {

                C[a][b] = max(C[a][b - 1], C[a - 1][b]);

                /\*for (int c = 0; c <= a; c++) {// or c <=b

                    if (c >= k) {

                        C[a][b] = max(C[a][b], C[a - c][b - c] + c);

                    }

                    if (s1.charAt(a - c) != s2.charAt(b - c)) {

                        break;

                    }

                }\*/

                if (s1.charAt(a)==s2.charAt(b)){

                    C[a][b]=C[a-1][b-1]+1;

                }

            }

        }

        return C[s1.length() - 1][s2.length() - 1];

    }

}

# **CUTTING PIZZA (Math)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main {

public static void main(String[] args)throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

long n= Long.parseLong(br.readLine());

while (n>=0){

System.out.println((n\*(n+1)/2)+1);

n=Long.parseLong(br.readLine());

}

}

}

# **SUB EXPESION COUNTING C++ (String)**

#include <iostream>

using namespace std;

int main(int argc, char \*argv[]) {

string cad1, cad2;

while(cin>>cad1>>cad2){

for(int i=0; i<cad1.length(); i++){

if((cad1[i]>47&&cad1[i]<58)||(cad1[i]>97&&cad1[i]<123)){

cad1[i] = 'x';

if(i>0 && cad1[i-1]=='x')

cad1.replace(i--,1,"");

}

if(cad1[i] == '-' || cad1[i] == '\*' || cad1[i] == '/'){

cad1[i] = '+';

}

}

for(int i=0; i<cad2.length(); i++){

if((cad2[i]>47&&cad2[i]<58)||(cad2[i]>97&&cad2[i]<123)){

cad2[i] = 'x';

if(i>0 && cad2[i-1]=='x')

cad2.replace(i--,1,"");

}

if(cad2[i] == '-' || cad2[i] == '\*' || cad2[i] == '/'){

cad2[i] = '+';

}

}

int contador=0;

while(cad2.find(cad1)<cad2.length()){

contador++;

cad2.replace(cad2.find(cad1), cad1.length(),"");

}

cout<<contador<<endl;

}

return 0;

# **EXPANDING RODS C++ (Math)**

#include <stdio.h>

#include <math.h>

int main() {

double L, C, n;

//longitud inicial, coeficiente, grados

const double pi = acos(-1);//3.1416.....

while(scanf("%lf %lf %lf", &L, &C, &n) == 3) {

//lectura en long double

if(n < 0)//si es menor cierra

break;

double LL = (1 + n\*C)\*L;//longitud final

double l = 0, r = pi/2, theta, R;

// radio, ángulo de elevación, altura del triángulo invisible

int cnt = 0;

while(l <= r && cnt < 50) {

theta = (l+r)/2;

R = L/2/sin(theta);

if(R\*2\*theta < LL)

l = theta;

else

r = theta;

cnt++;

}

printf("%.3lf\n", R-R\*cos(theta));

//impresión del resultado con tres decimales

}

return 0;

}

# **EXPANDING RODS JAVA (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

double L, C, n, LL;

Scanner s = new Scanner(System.in);

while (s.hasNext()) {

L = s.nextDouble();

C = s.nextDouble();

String ns=s.next();

n = Double.parseDouble(ns);

if (n < 0) {

break;

}

LL = (1 + n \* C) \* L;

double l = 0, r = Math.PI/ 2, theta = 0, R = 0;

int cnt = 0;

while (l <= r && cnt < 50) {

theta = (l + r) / 2;

R = L / 2 / Math.sin(theta);

if (R \* 2 \* theta < LL) {

l = theta;

} else {

r = theta;

}

cnt++;

}

double res = R - R\*Math.cos(theta);

System.out.printf("%.3f\n", res);

}

}

}

# [**FEYNMAN**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=243&page=show_problem&problem=3301) **(Geometry)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

new Main();

}

public Main() {

Scanner sc = new Scanner(System.in);

int n=1, i,suma=0;

while (sc.hasNextInt() && n != 0) {

n = sc.nextInt();

if(n==0)

break;

for (int j = n; j >=1; j--) {

i=(int) Math.pow(j,2);

suma+=i;

}

System.out.println(suma);

suma=0;

}

}

}

# **I LOVE PIZZA C++ (String)**

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <math.h>

#include <string.h>

int min(int x,int y){

if(x<y)

return x;

return y;

}

int main()

{

int n=0;

scanf("%d",&n);

while(n--){

int M=0,A=0,R=0,G=0,I=0,T=0,i=0,z=0;

char str[600];

scanf("%s",str);

z=strlen(str);

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

if(str[i]=='M')

M++;

else if(str[i]=='A')

A++;

else if(str[i]=='R')

R++;

else if(str[i]=='G')

G++;

else if(str[i]=='I')

I++;

else if(str[i]=='T')

T++;

}

int sol=min(min(min(min(min(M,A/3),R/2),G),I),T);

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

}

return 0;

}

# [**PEANOLAND CONTACTING GAUSSLAND**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=4825) **(Math)**

import java.lang.Math; import java.util.ArrayList;

import java.util.Scanner;

public class Main

{

public static void main(String[] args)

{

int a;

Scanner esc =new Scanner(System.in);

while(esc.hasNextInt()){

a=esc.nextInt();

System.out.println(convert(a));

} }

static class Gauss {

public int a, b;

@Override

public String toString() {

String res = "" + a + " " + b;

return res;

}

public void inc(Gauss g) {

a += g.a;

b += g.b;

}

}

static Gauss gpow(int k) {

Gauss r = new Gauss();

int n = k / 2;

int z = n % 4;

if (k == 0) {

r.a = 1;

r.b = 0;

}

else if (k == 1) {

r.a = -1;

r.b = 1;

}

else if (k % 2 == 0) {

if (n % 2 == 0) {

r.a = (int) Math.pow(2, n);

r.b = 0;

}

else {

r.a = 0;

r.b = (int) Math.pow(2, n);

}

if (z == 1) {

r.b = -r.b;

}

else if (z == 2) {

r.a = -r.a;

}

}

else {

r.a = (int) Math.pow(2, n);

r.b = r.a;

if (z == 0) {

r.a = -r.a;

}

else if (z == 2) {

r.b = -r.b;

}

else if (z == 3) {

r.a = -r.a;

r.b = -r.b;

}

}

return r;

}

static ArrayList<Integer> binario(int p) {

ArrayList<Integer> res = new ArrayList<Integer>();

while (p > 0) {

res.add(p % 2);

p /= 2;

}

return res;

}

static Gauss convert(int p) {

Gauss r = new Gauss();

int i = 0;

for (int n : binario(p)) {

if (n == 1) {

Gauss t = gpow(i);

r.inc(t);

}

i++;

}

return r;

}

}

# **STERN-BROCOT TREE (Tree)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class brocot {

public brocot() throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int nc = Integer.parseInt(br.readLine());

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

String path = br.readLine();

long n1=0,n2=1,n3=1;

long d1=1,d2=1,d3=0;

for (int j = 0; j < path.length(); j++) {

if(path.charAt(j)=='L'){

n3=n2;

d3=d2;

n2=n3+n1;

d2=d3+d1;

}else{

n1=n2;

d1=d2;

n2=n3+n1;

d2=d3+d1;

}

}

System.out.println(n2+"/"+d2);

}

}

public static void main(String[] args) {

try {

new brocot();

} catch (IOException ex) {

}

}

}

# **JUDGING TROUBLES (String)**

import java.util.HashMap;

import java.util.Map;

import java.util.Scanner;

public class Judging {

public Judging() {

Scanner s = new Scanner(System.in);

while(s.hasNextInt()){

int sub = s.nextInt();

HashMap<String, Integer> lista1 = new HashMap<String,Integer>();

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

String pal = s.next();

if(lista1.containsKey(pal)){

lista1.put(pal, lista1.get(pal)+1);

}else{

lista1.put(pal, 1);

}

}

HashMap<String, Integer> lista2 = new HashMap<String,Integer>();

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

String pal = s.next();

if(lista2.containsKey(pal)){

lista2.put(pal, lista2.get(pal)+1);

}else{

lista2.put(pal, 1);

}

}

int iguales=0;

for (Map.Entry<String, Integer> entrySet : lista1.entrySet()) {

String key = entrySet.getKey();

Integer value = entrySet.getValue();

if(lista2.containsKey(key)){

iguales += Math.min(value, lista2.get(key));

}

}

System.out.println(iguales);

}

}

public static void main(String[] args) {

new Judging();

}

}

# [**TRIANGLE WAVE**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=429) **(String)**

#include<iostream>

using namespace std;

void printWave(int amplitude)

{

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

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

                        cout<<i;

                }

                cout<<endl;

        }

        for(int j = amplitude-1; j >= 1; j--){

                for(int jj = 1; jj<=j ; jj++){

                        cout<<j;

                }

                cout<<endl;

        }

}

int main ()

{

        int cases , amp , freq;

        cin>>cases;

        for(int i=0 ; i<cases ; i++){

              if(i>0) cout<<endl;

                cin>>amp>>freq;

                for(int j=0 ; j<freq; j++){

                        if(j>0)cout<<endl;

                        printWave(amp);

                }

        }

        return 0;

}

# **BYE-BYE CAKES (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc =new Scanner(System.in);

int a,b,c,d,aa,bb,cc,dd,at,bt,ct,dt,mx,res1,res2,res3,res4;

while(sc.hasNextInt()){

a=sc.nextInt();

b=sc.nextInt();

c=sc.nextInt();

d=sc.nextInt();

aa=sc.nextInt();

bb=sc.nextInt();

cc=sc.nextInt();

dd=sc.nextInt();

if (a<0)

break;

at=(a+aa-1)/aa;

bt=(b+bb-1)/bb;

ct=(c+cc-1)/cc;

dt=(d+dd-1)/dd;

//if en caliente

mx=at;

mx=mx>bt? mx:bt;

mx= mx>ct? mx:ct;

mx=mx>dt? mx:dt;

res1=aa\*mx-a;

res2=bb\*mx-b;

res3=cc\*mx-c;

res4=dd\*mx-d;

System.out.println(res1+" "+res2+" "+res3+" "+res4);

}

}

}

# **HASHMAT THE BRAVE WARRIOR (Math)**

import java.util.Scanner;

public class Main {

    public Main(){

        long a,b,dife;

        Scanner sc = new Scanner(System.in);

        while(sc.hasNext()){

           a = sc.nextLong();

           b = sc.nextLong();

           if (a<=b){

               dife=b-a;

           }else

               dife=a-b;

        System.out.println(dife);

    }

    }

    public static void main(String[] args) {

       new Main();

    }

}

# **NEXT PALINDROMIC NUMBERS (Big numbers)**

import java.math.BigInteger;

import java.util.Scanner;

public class Main {

public Main() {

Scanner s = new Scanner(System.in);

while (s.hasNext()) {

int n = Integer.parseInt(s.next());

BigInteger d = new BigInteger(s.next());

BigInteger medio;

boolean par = false;

if (d.toString().length() % 2 == 1) {

medio = new BigInteger(d.toString().substring(0, d.toString().length() / 2 + 1));

} else {

medio = new BigInteger(d.toString().substring(0, d.toString().length() / 2));

par = true;

}

//System.out.println(n+" "+d+" "+medio);

while (n > 0) {

String numero = "";

if (par) {

numero = medio + (new StringBuilder(medio.toString())).reverse().toString();

} else {

numero = medio + (new StringBuilder(medio.toString().substring(0, medio.toString().length() - 1))).reverse().toString();

}

BigInteger numeroAImprimir = new BigInteger(numero);

if (numeroAImprimir.compareTo(d) > 0) {

System.out.println(numeroAImprimir);

n--;

}

if (medio.add(BigInteger.ONE).toString().length() > medio.toString().length()) {

if (medio.compareTo(BigInteger.TEN) < 0 && !par) {

medio = BigInteger.ZERO;

}

par = !par;

}

medio = medio.add(BigInteger.ONE);

}

}

}

public static void main(String[] args) {

new Main();

}

}

# **ECOLOGICAL PREMIUM (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc =new Scanner(System.in);

int n,f,metros, animales,ambiente;

n=sc.nextInt();

for (int i = n; i >0; i--) {

f=sc.nextInt();

int suma=0;

for (int j = f; j >0; j--) {

metros=sc.nextInt();

animales=sc.nextInt();

ambiente=sc.nextInt();

suma+=(metros\*ambiente);

}

System.out.println(suma);

}

}

}

# **JUST THE FACTS ( Big Numbers)**

#include<cstdio>

//Libreria predeterminada de C puro para lectura y escritura

int N, ld[10010];

void lnd(int n) {

long long m = 1;

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

m = (m \* i);

while(m % 10 == 0) m /= 10;

m = m % 100000;

ld[i] = m % 10;

}

}

int main() {

lnd(10000);

while(scanf("%d", &N) == 1) {

//scanf es lectura en C

printf("%5d -> %d\n", N, ld[N]);

/printd es impresion en C

}

}

# **BUSINESS CENTER (Math)**

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        new Main();

    }

    public Main() {

        Scanner sc = new Scanner(System.in);

        double n,  u, d;

        int m,aux,x,p;

        while (sc.hasNext()) {

            aux = Integer.MAX\_VALUE;

            n = sc.nextDouble();

            m = sc.nextInt();

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

                u = sc.nextDouble();

                d = sc.nextDouble();

                x =(int) Math.ceil((n \* d) / (u + d));

                p=(int) ((x\*u)-((n-x)\*d));

                if (p==0){

                    x++;

                 p= (int) ((x\*u)-((n-x)\*d));

                }

                if (p < aux) {

                    aux = p;

                }

            }

            System.out.println((int)aux);

        }

    }

}

# **ASCII AREA (Geometry)**

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int h, w,area;

        String entrada;

        while (sc.hasNextLine()) {

            int medios = 0, completos = 0;

            h = sc.nextInt();

            w = sc.nextInt();

            sc.nextLine();

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

                entrada = sc.nextLine();

                for (int j = 0; j < entrada.length(); j++) {

                    if (entrada.charAt(j) == '/' || entrada.charAt(j) == '\\') {

                        medios += 1;

                    }

                    if (entrada.charAt(j) == '.' && medios % 2 == 1) {

                        completos += 1;

                    }

                }

            }

            medios=(int) (medios\*0.5);

            area=medios+completos;

            System.out.println(area);

        }

    }

}

# **CANDY ( Dynamic )**

import java.io.\*;

import java.util.\*;

public class Main {

public static int max(int a, int b) { return (a>b)? a: b; }

public static int pint(String n) { return Integer.parseInt(n); }

public static void main(String[] args) throws Throwable {

BufferedReader in=new BufferedReader(new InputStreamReader(System.in));

PrintWriter out=new PrintWriter(new BufferedWriter(new OutputStreamWriter(System.out)));

int m,n,x,tmp,prevLine1,prevLine2,prevNum1,prevNum2;

String line; String[] tok;

tok=in.readLine().split("\\s+"); m=pint(tok[0]); n=pint(tok[1]);

while (m+n!=0) {

prevLine1=prevLine2=0;

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

prevNum1=prevNum2=0; tok=in.readLine().split("\\s+");

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

x=pint(tok[j]); tmp=prevNum1;

prevNum1=max(prevNum1, x+prevNum2);

prevNum2=tmp;

}

tmp=prevLine1;

prevLine1=max(prevLine1, prevNum1+prevLine2);

prevLine2=tmp;

}

out.printf("%d\n", prevLine1);

tok=in.readLine().split("\\s+"); m=pint(tok[0]); n=pint(tok[1]);

}

out.close();

in.close();

}

}

# **LUCKY THIEF C++ (Math)**

#include<iostream>

using namespace std;

int main (int argc, char \*argv[]) {

long m,n, resultado;

long t;

cin>>t;

if (t<0)

return 0;

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

cin>>n;

cin>>m;

resultado=(-n\*(1-2\*m+n))/2;

cout<<resultado<<endl;

}

return 0;

}

# **AUTOMATIC ANSWER (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int t,numero, salida;

String solo;

t=sc.nextInt();

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

numero=sc.nextInt();

salida=(((((numero\*567)/9)+7492)\*235)/47)-498;

solo=String.valueOf(salida);

System.out.println(solo.charAt(solo.length()-2));

}

}

}

# [**THE DECODER**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=399) **C++ (String)**

#include<iostream>

#include <string>

using namespace std;

int main (int argc, char \*argv[]) {

char c;

char linea [255];

while (cin>>linea) {

for(int i=0;linea[i]!='\0';i++){

c=linea[i]-7;

cout<<c;

}

cout<<endl;

}

return 0;

}

# **KINDERGARTEN COUNTING GAME (String)**

import java.util.Scanner;

class Main {

public static void main(String args[]) {

Scanner sc = new Scanner(System.in);

char c = ' ';

String linea;

int contador = 0;

while ((sc.hasNext())) {

linea = sc.nextLine();

boolean eraChar = false;

for (int i = 0; i < linea.length(); i++) {

if(!((linea.charAt(i)<123&&linea.charAt(i)>96)||(linea.charAt(i)<91&&linea.charAt(i)>64))){

if(eraChar){

contador++;

eraChar = false;

}

}else{

if(!eraChar)

eraChar=true;

}

}

if(eraChar)

contador++;

System.out.println(contador);

contador=0;

}

}

}

# [**ALARM CLOCK**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=2724) **(Math)**

#include <iostream>

using namespace std;

int main()

{

int h1,m1,h2,m2,resultado;

cin>>h1;

cin>>m1;

cin>>h2;

cin>>m2;

while (!(h1==0&&h1==0&&m1==0&&m2==0)){

if(h1==h2&&m1==m2)

resultado=0;

if (h1<h2||(h1==h2&&m1<m2))

resultado = ((h2\*60)+m2)-((h1\*60)+m1);

if (h1>h2||(h1==h2&&m1>m2) )

resultado=1440-(((h1\*60)+m1)-((h2\*60)+m2));

cout<<resultado<<endl;

resultado=0;

cin>>h1;

cin>>m1;

cin>>h2;

cin>>m2;

}

return 0;

}

# **A MULTIPLICATION GAME (Pattern)**

#include <iostream>

#include <cstdio>

#include <cmath>

using namespace std;

int main()

{

    long long int n;

    while(scanf("%lld",&n) == 1)

    {

         long long int lo;

         long double lg = logl(n)/logl(18);

        lo = floorl(lg);

        if (lg - lo < 0.00000000001) {cout << "Ollie wins.\n";continue;}

        if ( n/powl(18.0,lo) > 9.0 ) cout << "Ollie wins.\n";

        else cout << "Stan wins.\n";

    }

    return 0;

}

# **MAGIC FORMULA (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

double a,b,c,d,l,resultado;

int contador=0;

a=sc.nextDouble();

b=sc.nextDouble();

c=sc.nextDouble();

d= sc.nextDouble();

l=sc.nextDouble();

while (a!=0 ||b!=0|| c!=0||d!=0||l!=0){

for (int x = 0; x <=l; x++) {

resultado =(a\*Math.pow(x, 2))+(b\*x)+c;

if (resultado%d==0){

contador++;

}

}

System.out.println(contador);

contador=0;

a=sc.nextInt();

b=sc.nextInt();

c=sc.nextInt();

d= sc.nextInt();

l=sc.nextInt();

}

}

}

# **CARMICHAEL NUMBERS (Math )**

import java.util.Arrays;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

int[] ar = { 561, 1105, 1729, 2465, 2821, 6601, 8911, 10585, 15841, 29341, 41041, 46657, 52633, 62745, 63973, 75361 };

int n=-1;

while((n =in.nextInt())!=0)

System.out.println((Arrays.binarySearch(ar, n) >=0? "The number "+ n + " is a Carmichael number." : n + " is normal."));

}

}

# **COWS AND CARS (Probability)**

#include <iostream>

#include <iomanip>

using namespace std;

int main()

{

int NCOWS, NCARS, NSHOW;

while (cin >> NCOWS >> NCARS >> NSHOW)

{

// Dos casos.

// 1.Seleciona una vaca y cambia a un carro:

// NCOWS / (NCOWS + NCARS)

// \* NCARS / (NCOWS + NCARS - NSHOW - 1)

// 2.Selecciona un carro y cambia a un carro:

// NCARS / (NCOWS + NCARS)

// \* (NCARS - 1) / (NCOWS + NCARS - NSHOW - 1)

cout << setprecision(5)<< fixed<< ((double)NCARS \* (NCOWS + NCARS - 1)) / ((NCOWS + NCARS) \* (NCOWS + NCARS - NSHOW - 1))

<< endl;

}

return 0;

}

# [**SEARCHING FOR NESSY**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=1985) **(Geometry)**

#include <stdio.h>

using namespace std;

int main() {

int casos, alto,ancho;

scanf("%d",&casos);

while(casos--){

scanf("%d %d",&alto,&ancho);

printf("%d\n",(alto/3)\*(ancho/3));

}

return 0;

}

# [**BIG CHOCOLATE**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=1911) **(Geometry)**

#include <iostream>

using namespace std;

int main() {

int m,n;

while(cin>>m>>n){

cout<<m\*n-1<<endl;

}

return 0;

}

# [**FACTSTONE BENCHMARK**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=1857) **(Math)**

#include<stdio.h>

#include<math.h>

int main()

{

int year;

while (scanf("%d", &year), year)

{

int n = (year - 1940) / 10;

double k = pow ( 2, n) \* log10(2), sum = 0;

for (int i = 1; ; i++)

{

sum += log10(i);

if (sum > k)

{

printf("%d\n", i - 1);

break;

}

}

}

return 0;

}

# **BILLIARD (Geometry)**

#include <stdio.h>

#include <math.h>

int main() {

double a, b, s, m, n;

const double pi = acos(-1);

while(scanf("%lf %lf %lf %lf %lf", &a, &b, &s, &m, &n) == 5) {

if(a == 0 && b == 0 && s == 0 && m == 0)

break;

double M = a\*m, N = b\*n;

double A = atan(N/M)\*180/pi;

double v = sqrt(M\*M + N\*N)/s;

printf("%.2lf %.2lf\n", A, v);

}

return 0;

}

# [**KNIGHT MOVES**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=380) **(Graph)**

#include <vector>

#include <string>

#include <algorithm>

#include <iostream>

#include <queue>

using namespace std;

int dx[]={1,1,-1,-1,2,2,-2,-2};

int dy[]={2,-2,2,-2,1,-1,1,-1};

struct nodo

{

int ind1;

int ind2;

int costo;

nodo(int a, int b, int c)

{

ind1=a;

ind2=b;

costo=c;

}

};

int dfs(int a, int b, int m, int n)

{

queue<nodo> Q;

Q.push(nodo(a,b,0));

while(!Q.empty())

{

nodo aux=Q.front();

Q.pop();

if(aux.ind1==m && aux.ind2==n) return aux.costo;

for(int i=0;i<8;i++)

{

int I=aux.ind1+dx[i];

int J=aux.ind2+dy[i];

if(I>=1 && J>=1 && I<=8 && J<=8)

Q.push(nodo(I,J,aux.costo+1));

}

}

}

int main()

{

string s1,s2;

int x1,y1,x2,y2;

while(cin>>s1>>s2)

{

x1=s1[0]-'a'+1;

y1=s1[1]-'0';

x2=s2[0]-'a'+1;

y2=s2[1]-'0';

cout<<"To get from "<<s1<<" to "<<s2<<" takes "<<dfs(x1,y1,x2,y2)<<" knight moves."<<endl;

}//To get from e2 to e4 takes 2 knight moves.

return 0;

}

# [**ORDERING TASKS**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=1246) **(Graph)**

#include <cstdio>

#include <vector>

#include <stack>

using namespace std;

void visit(vector<vector<int> > &adj, vector<int> &list,

vector<bool> &seen, int u) {

for (int i = 0; i < adj[u].size(); i++) {

if (!seen[adj[u][i]]) visit(adj, list, seen, adj[u][i]);

}

seen[u] = true;

list.insert(list.begin(), u + 1);

}

int main() {

int n, m, u, v;

vector<vector<int> > adj;

for (;;) {

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

if (n == 0 && m == 0) break;

adj.clear();

adj.resize(n);

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

scanf("%d %d", &u, &v);

adj[u - 1].push\_back(v - 1);

}

// Topological sort.

vector<int> list;

vector<bool> seen;

seen.clear();

seen.resize(n);

for (int i = 0; i < n; i++) seen[i] = false;

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

if (!seen[i]) {

visit(adj, list, seen, i);

}

}

printf("%d", list[0]);

for (int i = 1; i < list.size(); i++)

printf(" %d", list[i]);

printf("\n");

}

}

# [**TRAFFIC VOLUME**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=1634) **(Math)**

#include<stdio.h>

#include<math.h>

int main()

{

double vit,volume;

int l,f;

while(1)

{

scanf("%d %d",&l,&f);

if(l==0&& f==0)

break;

vit=sqrt(l\*f\*2.0);

volume=(vit\*3600.0)/(2.0\*l);

printf("%0.8lf %0.8lf\n",vit,volume);

}

return 0;

}

# **CHESS (math)**

#include<cstdio>

#include<cstring>

#include<algorithm>

using namespace std;

int main(){

int t;

scanf("%d\n", &t);

char c;

int M, N;

int answer = 0;

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

scanf("%c %d %d\n", &c, &M, &N);

answer = 0;

if(c == 'r'){

answer = min(M, N);

}

else if(c == 'Q'){

answer = min(M, N);

}

else if(c == 'k'){

answer = (((M + 1)/2) \* ((N + 1)/2)) + (((M)/2) \* ((N)/2));

}

else if(c == 'K'){

answer = (((M + 1)/2) \* ((N + 1)/2));

}

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

}

}

# **ELECTRICITY (Math)**

import java.util.Scanner;

public class Main {

public static int N;

public static int sum;

public static int nDays;

public static boolean isLeap(int y) {

return (((y % 4) == 0 && (y % 100) != 0) || (y % 400) == 0);

}

public static boolean consecutive(int d1, int m1, int y1, int d2, int m2, int y2) {

if (m1 == 1 || m1 == 3 || m1 == 5 || m1 == 7 || m1 == 8 || m1 == 10 || m1 == 12) {

if (d1 <= 30 && d2 == d1 + 1 && m2 == m1 && y2 == y1) {

return true;

}

if (d1 == 31 && m1 < 12 && d2 == 1 && m2 == m1 + 1 && y2 == y1) {

return true;

}

if (d1 == 31 && m1 == 12 && d2 == 1 && m2 == 1 && y2 == y1 + 1) {

return true;

}

} else if (m1 == 4 || m1 == 6 || m1 == 9 || m1 == 11) {

if (d1 <= 29 && d2 == d1 + 1 && m2 == m1 && y2 == y1) {

return true;

}

if (d1 == 30 && d2 == 1 && m2 == m1 + 1 && y2 == y1) {

return true;

}

} else if (m1 == 2) {

if ((d1 <= 27 || (d1 == 28 && isLeap(y1))) && d2 == d1 + 1 && m2 == m1 && y2 == y1) {

return true;

}

if (((d1 == 29 && isLeap(y1)) || (d1 == 28 && !isLeap(y1))) && d2 == 1 && m2 == 3 && y2 == y1) {

return true;

}

}

return false;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int d1, m1, y1, d2, m2, y2;

int c1, c2;

N = sc.nextInt();

while (N != 0) {

nDays = 0;

sum = 0;

d1 = sc.nextInt();

m1 = sc.nextInt();

y1 = sc.nextInt();

c1 = sc.nextInt();

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

d2 = sc.nextInt();

m2 = sc.nextInt();

y2 = sc.nextInt();

c2 = sc.nextInt();

if (consecutive(d1, m1, y1, d2, m2, y2)) {

nDays++;

sum += (c2 - c1);

}

d1 = d2;

m1 = m2;

y1 = y2;

c1 = c2;

}

System.out.println(nDays + " " + sum);

N = sc.nextInt();

}

}

}

# **BENDER B. RODRÍGUEZ PROBLEM (Geometry)**

#include <cstdio>

using namespace std;

bool change(char cur, char c1, char c2){

if(cur == 'x' && c1 == '-') return true;

if(cur == 'y' && c2 == 'y' && c1 == '+') return true;

if(cur == 'z' && c2 == 'z' && c1 == '+') return true;

return false;

}

int main(){

int L;

char s[3];

while(true){

scanf("%d",&L);

if(L == 0) break;

--L;

int sign = 1;

char c = 'x';

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

scanf("%s",s);

if(s[0] == 'N') continue;

if(change(c,s[0],s[1])) sign = -sign;

if(c == 'x') c = s[1];

else if(c == 'y' && s[1] == 'y') c = 'x';

else if(c == 'z' && s[1] == 'z') c = 'x';

}

printf("%c%c\n",sign == 1? '+' : '-',c);

}

return 0;

}

# **AUTOMATE THE GRADES (Math)**

import java.util.Scanner;

public class grades {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int t1, t2, fin, asistencia, nota1, nota2, nota3, notatotal, aux1 = 0, aux2 = 0, casos, salida, pasada = 1;

casos = sc.nextInt();

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

t1 = sc.nextInt();

t2 = sc.nextInt();

fin = sc.nextInt();

asistencia = sc.nextInt();

nota1 = sc.nextInt();

nota2 = sc.nextInt();

nota3 = sc.nextInt();

if (nota1 > nota2 || nota1 > nota3) {

aux1 = nota1;

if (nota2 > nota3) {

aux2 = nota2;

} else {

aux2 = nota3;

}

} else {

aux1 = nota2;

aux2 = nota3;

}

notatotal = (aux1 + aux2) / 2;

salida = t1 + t2 + fin + asistencia + notatotal;

if (salida >= 90) {

System.out.println("Case " + pasada + ": A");

}

if (salida >= 80 && salida < 90) {

System.out.println("Case " + pasada + ": B");

}

if (salida >= 70 && salida < 80) {

System.out.println("Case " + pasada + ": C");

}

if (salida >= 60 && salida < 70) {

System.out.println("Case " + pasada + ": D");

}

if (salida < 60) {

System.out.println("Case " + pasada + ": F");

}

pasada++;

}

}

}

# [**SQUARE ROOT**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=964) **(Big Numbers)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.math.BigInteger;

public class Main {

public static void main(String[] args) throws IOException {

BigInteger bg=BigInteger.ZERO;

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

StringBuffer sb = new StringBuffer("");

String m = "";

int cases=Integer.parseInt(br.readLine());

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

br.readLine();

if(i>0)

sb.append("\n");

bg=new BigInteger(br.readLine());

sb.append(bigsqrt(bg)).append("\n");

}

System.out.print(sb);

}

//Implementando raiz en BigInt

static BigInteger bigsqrt(BigInteger A)

{

boolean sirve ;

BigInteger temp = A.shiftRight(BigInteger.valueOf(A.bitLength()).shiftRight(1).intValue()), result = null;

while (true)

{

result = temp.add(A.divide(temp)).shiftRight(1);

if (!temp.equals(result))

temp = result;

else

break ;

}

sirve = false ;

if (result.multiply(result).equals(A))

sirve = true ;

return result;

}

}

# [**ONE-TWO-THREE**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=3710) **(String)**

#include <stdio.h>

#include <string.h>

int main ()

{

int testCase; scanf ("%d", &testCase);

while ( testCase-- )

{

char a [10]; scanf ("%s", a);

if ( strlen (a) == 5 ) printf ("3\n");

else

{

int cnt = 0;

if ( a [0] == 'o' ) cnt++;

if ( a [1] == 'n' ) cnt++;

if ( a [2] == 'e' ) cnt++;

if ( cnt >= 2 ) printf ("1\n");

else printf ("2\n");

}

}

return 0;

}

# **FIBONACCI WORDS (String)**

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.Arrays;

class Main {

static String[] prefix = new String[102];

static String[] sufix = new String[102];

static long[] ans = new long[102];

static char[] pattern;

static char[] text;

static int[] f = new int[100005];

static int n, m;

static void kmpProcess() {

int i = 0, j;

j = f[0] = -1;

while (i < m) {

while (j >= 0 && pattern[i] != pattern[j])

j = f[j];

i++;

j++;

f[i] = j;

}

}

static int kmpSerach() {

int res = 0;

int i = 0, j = 0;

n = text.length;

while (i < n) {

while (j >= 0 && text[i] != pattern[j])

j = f[j];

i++;

j++;

if (j == m) {

res++;

j = f[j];

}

}

return res;

}

static void makePrefix(int i) {

prefix[i] = prefix[i - 1];

int remaining = m - 1 - prefix[i].length();

if (remaining > 0) {

int last = remaining > prefix[i - 2].length() ? prefix[i - 2]

.length() : remaining;

prefix[i] += prefix[i - 2].substring(0, last);

}

}

static void makeSufix(int i) {

sufix[i] = sufix[i - 2];

int remaining = m - 1 - sufix[i].length();

if (remaining > 0) {

int first = remaining > prefix[i - 1].length() ? 0 : prefix[i - 1]

.length() - remaining;

sufix[i] = sufix[i - 1].substring(first) + sufix[i];

}

}

public static void main(String[] args) throws Exception {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

StringBuilder out = new StringBuilder();

String l;

prefix[0] = sufix[0] = "0";

prefix[1] = sufix[1] = "1";

int c = 1;

while ((l = br.readLine()) != null) {

Arrays.fill(ans, 0);

int N = Integer.parseInt(l);

pattern = br.readLine().toCharArray();

m = pattern.length;

kmpProcess();

if (pattern.length == 1)

ans[pattern[0] - '0']++;

int i = 2;

while (i <= N) {

ans[i] = ans[i - 1] + ans[i - 2];

if (m > 1) {

makePrefix(i);

makeSufix(i);

text = (sufix[i - 1] + prefix[i - 2]).toCharArray();

ans[i] += kmpSerach();

}

i++;

}

out.append("Case " + c++ + ": " + ans[N] + "\n");

}

System.out.print(out);

}

}

# **SETTLES OF CATAN ( Graph)**

import java.util.\*;

public class Main {

public static int[][] Matrix;

public static int NumberOfNodes, NumberOfEdges;

public static boolean[][] Visited;

public static int First, Second;

public static void main(String[] args) {

Scanner Input = new Scanner(System.in);

while(Input.hasNext()) {

NumberOfNodes = Input.nextInt();

NumberOfEdges = Input.nextInt();

if(NumberOfEdges == 0 && NumberOfNodes == 0) break;

Matrix = new int[NumberOfNodes + 1][NumberOfNodes + 1];

Visited = new boolean[NumberOfNodes + 1][NumberOfNodes + 1];

int Answer = 0;

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

First = Input.nextInt();

Second = Input.nextInt();

Matrix[First][Second] = Matrix[Second][First] = 1;

}

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

Answer = Math.max(Answer, Solve(i));

}

System.out.println(Answer);

}

}

public static int Solve(int Node) {

int Result = 0;

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

if(Matrix[Node][i] > 0 && !Visited[Node][i]) {

Visited[Node][i] = Visited[i][Node] = true;

Result = Math.max(Result, Solve(i) + 1);

Visited[Node][i] = Visited[i][Node] = false;

}

}

return Result;

}

}

# [**SMOOTH FACTOR**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=4653) **(Math)**

import java.util.\*;

public class Main {

static Scanner sc = new Scanner(System.in);

static int[] arreglo;

static int tamano, entrada, smooth;

public static void main(String[] args) {

while (sc.hasNext()) {

tamano = sc.nextInt();

arreglo = new int[tamano];

for (int i = 0; i < arreglo.length; i++) {

arreglo[i] = entrada = sc.nextInt();

}

smooth = 1;

int cnt = 1, c = 0, pre = 0;

for (int i = 1; i <= tamano - 1; i++) {

if (arreglo[i - 1] <= arreglo[i]) {

cnt++;

} else {

c++;

if (c == 2) {

c--;

cnt = i - pre + 1;

pre = i;

} else {

cnt++;

pre = i;

}

}

if (smooth < cnt) {

smooth = cnt;

}

}

System.out.println(smooth);

}

}

}

# **AUDIOPHOBIA (Graph)**

import java.util.Scanner;

class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int testCase=0;

while(true)

{

testCase++;

// cruces, calles y cantidad

int cross=sc.nextInt(),street=sc.nextInt(),query=sc.nextInt();

if(cross==0 && street==0 && query==0) break;

// matriz de adyacencia

int weight[][] = new int[cross+1][cross+1];

int r,c;

for(int i=1;i<=cross;i++)

for(int j=1;j<=cross;j++)

//inicializado en inf

weight[i][j] = Integer.MAX\_VALUE;

for(int i=1;i<=street;i++)

{

r = sc.nextInt();

c = sc.nextInt();

weight[r][c] = weight[c][r] = sc.nextInt();

}

for(int k=1; k<=cross; k++)

for(int i=1; i<=cross; i++)

for(int j=1;j<=cross;j++)

weight[i][j] = weight[j][i] = Math.min(weight[i][j], Math.max(weight[i][k], weight[k][j]));

if(testCase>1) System.out.println("");

System.out.printf("Case #%d%n",testCase);

for(int i=1;i<=query;i++)

{

r = sc.nextInt();

c = sc.nextInt();

if(weight[r][c] == Integer.MAX\_VALUE) System.out.println("no path");

else System.out.printf("%d%n", weight[r][c]);

}

}

}

}

# **TENNIS ROUNDS (Math)**

import java.util.Scanner;

class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n,i,j,elevado,mitad=0,mayor,menor;

while (sc.hasNextInt()){

n= sc.nextInt();

i=sc.nextInt();

j=sc.nextInt();

if (i<j){

menor =i;

mayor =j;

}else {

menor =j;

mayor =i;

}

elevado=(int)Math.pow(2, n-1);

mitad =elevado;

for (int k = n; k > 0; k--) {

elevado =(int)Math.pow(2,k-2);

if (menor<=mitad && mayor >mitad){

System.out.println(k);

break;

}else{

if (menor>mitad){

mitad+=elevado;

}else {

mitad-=elevado;

}

}

}

}

}

}

# **PHILIP J. FRY PROBLEM (Queqe)**

import java.util.Comparator;

import java.util.PriorityQueue;

import java.util.Scanner;

public class Main {

public Main() {

int trips;

Scanner s = new Scanner(System.in);

trips = s.nextInt();

while (trips != 0) {

int[] tiempos = new int[trips];

int[] bolas = new int[trips];

int suma = 0;

PriorityQueue<Integer> valores = new PriorityQueue<Integer>(new Comparator<Integer>() {

@Override

public int compare(Integer o1, Integer o2) {

return o2.compareTo(o1);

}

});

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

tiempos[i] = s.nextInt();

bolas[i] = s.nextInt();

}

for (int i = trips - 1; i >= 0; i--) {

if (bolas[i] != 0) {

int temp = bolas[i];

while (temp > 0 && !valores.isEmpty()) {

suma -= valores.poll() / 2;

temp--;

}

}

valores.add(tiempos[i]);

suma += tiempos[i];

}

System.out.println(suma);

trips = s.nextInt();

}

}

public static void main(String[] args) {

new Main();

}

}

# **SCORE (String)**

import java.util.Scanner;

class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int casos, ceros = 0, acumulado = 0;

String entrada;

char toma;

casos = sc.nextInt();

for (int j = 0; j < casos; j++) {

acumulado = 0;

ceros = 0;

entrada = sc.next();

for (int i = 0; i < entrada.length(); i++) {

toma = entrada.charAt(i);

if (toma == 'O') {

ceros++;

acumulado += ceros;

} else {

ceros = 0;

}

}

System.out.println(acumulado);

}

}

}

# **POWER OF CRYPTOGRAPHY (Big numbers)**

import java.util.Scanner;

class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

while (sc.hasNext()) {

double elevado = sc.nextDouble();

double raiz = sc.nextDouble();

double res = Math.pow(raiz, 1 / elevado);

res = Math.round(res);

System.out.printf("%d\n", (int) res);

}

}}

# **DECODE THE TAPE (String)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

boolean entrando= true;

int claves=0;

String entrada;

char salida;

byte[] caracter ;

while (entrando){

entrada = sc.nextLine();

if (entrada.equals("\_\_\_\_\_\_\_\_\_\_\_"))

claves++;

if (claves==2){

entrando = false;

}

entrada= entrada.replace("|", "");

entrada=entrada.replace(".", "");

entrada=entrada.replace("o", "1");

entrada= entrada.replace(" ", "0");

// entrada="0"+entrada;

if (!entrada.equals("\_\_\_\_\_\_\_\_\_\_\_")){

salida=(char) Integer.parseInt(entrada,2);

System.out.print(salida);

}

}

}

}

# **PRODUCT (Big numbers)**

import java.math.BigInteger;

import java.util.Scanner;

public class Main2 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

while (sc.hasNext()) {

BigInteger numero1, numero2, salida;

numero1 = sc.nextBigInteger();

numero2 = sc.nextBigInteger();

salida = numero1.multiply(numero2);

System.out.println(salida);

}

}

}

# **INTEGER INQUIRY (Big numbers)**

import java.math.BigInteger;

import java.util.Scanner;

public class Main2 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

BigInteger numero1, suma = BigInteger.ZERO, salida;

while (sc.hasNext()) {

String copy;

numero1 = sc.nextBigInteger();

copy = numero1.toString();

if (copy.equals("0")) {

System.out.println(suma);

break;

}

suma= suma.add(numero1);

}

}

}

# **OVERFLOW (String)**

import java.math.BigInteger;

import java.util.Scanner;

class Main2 {

static final BigInteger MX = new BigInteger("2147483647");

public static void main(String args[]) {

Scanner in = new Scanner(System.in);

boolean a, b, c;

while (in.hasNext()) {

String line = in.nextLine();

String splt[] = line.split(" ");

BigInteger numero1 = new BigInteger (splt[0]);

BigInteger numero2 = new BigInteger (splt[2]);

System.out.println(line);

if (check(numero1) == false) {

System.out.println("first number too big");

}

if (check(numero2) == false) {

System.out.println("second number too big");

}

if (splt[1].equals("+")) {

numero1 = numero1.add(numero2);

} else {

numero1 = numero1.multiply(numero2);

}

if (check(numero1) == false) {

System.out.println("result too big");

}

}

}

static boolean check(BigInteger big) {

if (big.compareTo(MX) == 1) {

return false;

}

return true;

}

}

# **EXPONENTIATION (Big numbers)**

import java.util.Scanner;

import java.math.BigDecimal;

public class Main2

{

public static void main(String[] args)

{

Scanner cin = new Scanner(System.in);

while (cin.hasNext())

{

BigDecimal num = cin.nextBigDecimal();

int n = cin.nextInt();

num = num.pow(n);

String s = num.toPlainString();

if (s.charAt(0) == '0')

s = s.substring(1);

int end = s.length();

while (s.charAt(end - 1) == '0')

end--;

if (s.charAt(end - 1) == '.')

end--;

s = s.substring(0, end);

System.out.println(s);

}

}

}

# **IF WE WERE A CHILD AGAIN (Big numbers)**

import java.math.BigInteger;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

BigInteger num1, num2, salida;

String operador;

String[] separar;

while (sc.hasNext()) {

num1 = sc.nextBigInteger();

operador = sc.next();

num2 = sc.nextBigInteger();

if (operador.equals("/")) {

salida = num1.divide(num2);

} else {

salida = num1.mod(num2);

}

System.out.println(salida);

}

}

}

# **DEGREES OF SEPARATION (Graph)**

#include<stdio.h>

#include<iostream>

#include<string.h>

#include<stdlib.h>

using namespace std;

const int MaxP=50;

int P,R,cases,i,j,k,sum,ans;

string name[MaxP],s1,s2;

int a[MaxP][MaxP];

int find(string s) {

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

if(name[i]==s)

return i;

name[sum]=s;sum++;

return sum-1;

}

bool init() {

scanf("%d %d",&P,&R);

if(P==0&R==0)

return false;

sum=0;

memset(a,0,sizeof(a));

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

cin>>s1>>s2;

j=find(s1);

k=find(s2);

a[j][k]=a[k][j]=1;

}

return true;

}

int main() {

cases=0;

while(init())

{

for(k=0;k<P;k++)

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

for(j=0;j<P;j++)

if(a[i][k]&&a[k][j])

if(a[i][j]==0||a[i][j]>a[i][k]+a[k][j]) a[i][j]=a[j][i]=a[i][k]+a[k][j];

ans=0;

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

for(j=i+1;j<P;j++)

if(a[i][j]==0)

ans=P+10;

else if(a[i][j]>ans)

ans=a[i][j];

if(ans>P)

printf("Network %d: DISCONNECTED\n\n",++cases);

else printf("Network %d: %d\n\n",++cases,ans);

}

return 0;

}

# **NUMERICAL SURPRISES (Big numbers)**

**Python**

numTest = int(input())

for itertest in range(numTest):

N = int(input())

P = int(input().strip())

remains = 0

remains =P % N

print (remains)

# **POWERS ET AL. (Modular math)**

import java.math.BigInteger;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in);

String m = sc.next();

BigInteger n = new BigInteger(sc.next());

while (!m.equals("0")&&!n.equals(BigInteger.ZERO)){

int mat [][] ={{0},{1},{6,2,4,8},{1,3,9,7},{6,4},{5},{6},{1,7,9,3},{6,8,4,2},{1,9}};

int a= Integer.parseInt(m.substring(m.length()-1));

BigInteger mod = new BigInteger(mat[a].length+"");

n=n.mod(mod);

System.out.println(mat[a][n.intValue()]);

m=sc.next();

n= new BigInteger(sc.next());

}

# **ANDY'S FIRST DICTIONARY (String)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.ArrayList;

import java.util.Collections;

import java.util.HashSet;

public class Main {

public static void main(String[] args) throws IOException

{

BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

String line, aux;

HashSet<String> dictionary = new HashSet<String>();

ArrayList<String> list = new ArrayList<String>();

int i;

while ((line = in.readLine()) != null)

{

aux = "";

for (i=0; i<line.length(); i++)

{

if (Character.isLetter(line.charAt(i))){

aux += line.charAt(i);

}else{

aux = aux.toLowerCase();

if (aux.length() > 0){

dictionary.add(aux);

}

aux = "";

}

}

if (aux.length()> 0){

aux = aux.toLowerCase();

dictionary.add(aux);

}

}

list.addAll(dictionary);

Collections.sort(list);

for (i=0; i<list.size(); i++){

System.out.println(list.get(i));

}

}

}

}

}

# **BOX OF BRICKS (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n, casos, suma = 0, prom, salida = 0,c=1

;

int[] alturas;

n = sc.nextInt();

while (n != 0) {

suma=0;

salida=0;

prom=0;

alturas = new int[n];

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

alturas[i] = sc.nextInt();

suma += alturas[i];

}

prom = suma / n;

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

if (alturas[j]>prom)

salida += alturas[j] - prom;

}

System.out.println("Set #" + c);

System.out

.printf("The minimum number of moves is %d.\n\n", salida);

c++;

n = sc.nextInt();

}

}

}

# **CRAZY TEA PARTY (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n, entrada, salida;

n = sc.nextInt();

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

entrada = sc.nextInt();

salida= (entrada/2)\*((entrada-1)/2);

System.out.println(salida);

}

}

}

# **GCD LCM (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int gcd, lcm, t, i;

t = sc.nextInt();

for (i = 1; i <= t; i++) {

gcd = sc.nextInt();

lcm = sc.nextInt();

if (lcm % gcd == 0) {

System.out.println(gcd + " " + lcm);

} else {

System.out.println("-1");

}

}

}

}

# **DIGIT COUNTING (String)**

#include<bits/stdc++.h>

using namespace std;

int main(){

int T,n;

cin >> T;

for(int t = 0; t < T; t++){

int cnt[10] = {0};

cin >> n;

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

int tmp = i;

while(tmp != 0){

cnt[tmp%10]++;

tmp/=10;

}

}

for(int i = 0; i <= 9; i++){

cout << cnt[i];

if(i < 9)cout << " ";

}

cout << endl;

}

return 0;

}

# **MOLAR MASS (String)**

#include<stdio.h>

static char s[128];

#define H 1.008

#define C 12.01

#define O 16.0

#define N 14.01

int main(){

register unsigned int n, i;

unsigned int i1, i2;

scanf("%u", &n);

while(n--){

scanf("%s", &s);

float sum = 0;

for(i = 0; s[i]; ++i){

i1 = s[i + 1];

i2 = s[i + 2];

unsigned i1check = i1 >= '0' && i1 <= '9';

switch(s[i]){

case 67:

if(i1check && (i2 >= '0' && i2 <= '9')){

sum += 120.1 \* i1 + C \* i2 - 6341.28;

break;

}else if(i1check){

sum += C \* i1 - 576.48;

break;

}else{

sum += C;

break;

}

case 72:

if(i1check && (i2 >= '0' && i2 <= '9')){

sum += 10.08 \* i1 + H \* i2 - 532.224;

break;

}else if(i1check){

sum += H \* i1 - 48.384;

break;

}else{

sum += H;

break;

}

case 78:

if(i1check && (i2 >= '0' && i2 <= '9')){

sum += 140.1 \* i1 + N \* i2 - 7397.28;

break;

}else if(i1check){

sum += N \* i1 - 196.14;

break;

}else{

sum += N;

break;

}

case 79:

if(i1check && (i2 >= '0' && i2 <= '9')){

sum += 160.0 \* i1 + O \* i2 - 8448.0;

break;

}else if(i1check){

sum += O \* i1 - 768.0;

break;

}else{

sum += O;

break;

}

}

}

printf("%.3f\n", sum);

}

return 0;

}

# **WHAT'S CRYPTANALYSIS? (String-Sort)**

#include <stdio.h>

int main()

{

int T, num[128]={0}, Max = 0;

char str[10000];

//freopen("test.txt", "r", stdin);

scanf("%d", &T);

gets(str);

while(T--){

gets(str);

for(int i = 0; str[i]; i++){

if(str[i] >= 97 && str[i] <= 122) str[i] -= 32;

num[str[i]]++;

if(num[str[i]] > Max) Max = num[str[i]];

}

}

for(int i = Max; i >= 1; i--)

for(int j = 65; j <= 90; j++)

if(num[j] == i) printf("%c %d\n", j, i);

return 0;

}

# **EDITOR (Dynamic)**

#include <iostream>

#include <string>

#include <cstring>

#define MAX 5001

using namespace std;

int T[MAX][MAX];

int main() {

int t;

cin >> t;

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

string s;

cin>>s;

int sz = s.size();

int maxx = 0;

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

for(int j=1; j<=sz; j++) {

if (s[i-1] == s[j-1] && i!=j)

maxx = max(maxx, T[i][j] = T[i-1][j-1]+1);

else

T[i][j] = 0;

}

}

cout << maxx << endl;

}

return 0;

}

# **SKEW BINARY (Math)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.ArrayList;

/\*\*

\* Created by AhmedAmr on 10/29/15.

\*/

public class Main {

public static void main(String[] args) throws IOException {

BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

ArrayList<String> line = readStrings(in.readLine());

while(!line.get(0).equals("0")){

char[] num = line.get(0).toCharArray();

int size = num.length;

int res = 0;

int fact = 2;

for (int i = size-1; i >=0; i--) {

if(num[i]!='0'){

res+=Integer.parseInt(String.valueOf(num[i]))\*(fact-1);

}

fact\*=2;

}

System.out.println(res);

line = readStrings(in.readLine());

}

}

public static ArrayList<String> readStrings(String line) {

String[] arr = line.split(" ");

ArrayList<String> res = new ArrayList<>();

for (String s : arr) {

if (!s.isEmpty()) res.add(s);

}

return res;

}

}

# **HE IS OFFSIDE (Math)**

#include <stdio.h>

#include <algorithm>

int a,d;

int b[11],c[11];

using namespace std;

int main (){

int i;

scanf("%d%d",&a,&d);

while (a||d){

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

scanf("%d",b+i);}

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

scanf("%d",c+i);

}

sort(b,b+a);

sort(c,c+d);

printf("%c\n",b[0]<c[1]?'Y':'N');

scanf ("%d%d",&a,&d);

}

return 0;

}

# **SHRINKING POLYGONS (Geometry)**

#include <stdio.h>

#define MAXVERTICES 100000

#define TRUE 1

#define FALSE 0

int distancia[MAXVERTICES+1];

int lado, circunferencia;

int n;

int check(int k){

int i,j,jj,in,next,numlados,distanciaActual;

lado=circunferencia/k;

in= distanciaActual=0;

distanciaActual=distancia[in];

while (distanciaActual<lado){

in++;

distanciaActual+=distancia[in];

}

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

numlados=distanciaActual=0;

next=lado;

j=i;

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

distanciaActual+=distancia[j];

if (distanciaActual==next){

next+=lado;

numlados++;

}else if (distanciaActual>next)

break;

j=(j+1)%n;

}

if (numlados==k)

return TRUE;

}

return FALSE;

}

int main (void){

int i;

while (scanf("%d",&n)==1&&n){

circunferencia=0;

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

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

circunferencia+=distancia[i];

}

for (i=n;i>2;i--)

if (circunferencia%i==0&&check(i)){

printf("%d\n",n-i);

break;

}

if (i==2)

puts("-1");

}

return 0;

}

# **IDENTIFYING TEA (Math)**

import java.util.Scanner;

public class Main {

static int[] respuesta = new int[5];

static int t, contador = 0;

public static final void main(String[] args) {

Scanner sc = new Scanner(System.in);

while (sc.hasNext()) {

contador = 0;

t = sc.nextInt();

for (int i = 0; i < 5; i++) {

respuesta[i] = sc.nextInt();

}

for (int i = 0; i < 5; i++) {

if (t == respuesta[i]) {

contador++;

}

}

System.out.println(contador);

}

}

}

# **HANGMAN JUDGE (String)**

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

int main()

{

size\_t round;

while (cin >> round, round != -1)

{

string solution, guess;

cin >> solution >> guess;

int stroke = 0;

for (size\_t i = 0; i < guess.size()

&& stroke <= 6

&& solution != ""; ++i)

{

if (solution.find(guess[i]) != string::npos)

{

solution.erase(remove(solution.begin(), solution.end(),

guess[i]),

solution.end());

}

else

++stroke;

}

cout << "Round " << round << endl;

if (stroke == 7)

cout << "You lose." << endl;

else if (solution == "")

cout << "You win." << endl;

else

cout << "You chickened out." << endl;

}

return 0;

}

# **D AS IN DAEDALUS (Math)**

#include <stdio.h>

int main()

{

int p, r, q;

while(scanf("%d%d", &p, &r)!=EOF)

{

q = p-1;

int ara[q], bnk, b, cnt=0, sum=0, i, j, nit;

for(i=1; i<=r; i++)

{

sum = 0;

scanf("%d", &bnk);

scanf("%d", &b);

for(j=0; j<q; j++)

{

scanf("%d", &ara[j]);

sum += ara[j];

}

if(sum>=bnk)

continue;

if(sum+b > bnk) b=0;

nit = bnk - sum;

if(bnk-sum>=10000) cnt += (10000 - b);

else if((bnk-sum)>=1000) cnt += (1000 - b);

else if((bnk-sum)>=100) cnt += (100 - b);

else if((bnk-sum)>=10) cnt += (10 - b);

else if((bnk-sum)>=1) cnt += (1- b);

}

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

}

return 0;

}

# **ACIS, A Contagious vIruS (Geometry)**

import java.awt.geom.Line2D;

import java.awt.geom.Point2D;

import java.math.RoundingMode;

import java.text.DecimalFormat;

import java.util.Scanner;

public class Main {

public static String cambio (double num){

DecimalFormat formato = new DecimalFormat();

formato.setMinimumFractionDigits(3);

formato.setMaximumFractionDigits(3);

formato.setRoundingMode(RoundingMode.HALF\_UP);

return formato.format(num).replace(",", ".");

}

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

String limit = s.next();

while (!limit.equals("\*")) {

int temp = Integer.parseInt(limit);

double min = 0;

Point2D center = new Point2D.Double(s.nextDouble(), s.nextDouble());

Line2D line2D = new Line2D.Double(s.nextDouble(), s.nextDouble(), s.nextDouble(), s.nextDouble());

Point2D inicio = line2D.getP1();

min = line2D.ptSegDist(center);

for (int i = 2; i < temp; i++) {

Point2D p = line2D.getP2();

line2D = new Line2D.Double(p, new Point2D.Double(s.nextDouble(), s.nextDouble()));

if (min > line2D.ptSegDist(center)) {

min = line2D.ptSegDist(center);

}

}

Point2D p = line2D.getP2();

line2D = new Line2D.Double(p, inicio);

if (min > line2D.ptSegDist(center)) {

min = line2D.ptSegDist(center);

}

System.out.println(cambio(min));

limit = s.next();

}

}

}

# **FUNNY CARDIOLOGIST (Geometry)**

#include <bits/stdc++.h>

using namespace std;

const double INF = 1e+30;

double dp[260][260] = {};

int main() {

int n, m;

while (scanf("%d %d", &n, &m) == 2) {

int x[260], y[260];

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

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

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

dp[i][1] = i == 0 ? 0 : INF;

for (int j = 2; j <= n- m; j++)

dp[i][j] = INF;

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

double dist = hypot(x[i]-x[k], y[i]-y[k]);

for (int j = 2; j <= n-m; j++)

dp[i][j] = min(dp[i][j], dp[k][j-1] + dist);

}

}

double ret = dp[n-1][n-m];

printf("%.3lf\n", ret);

}

return 0;

}

# **ECLIPSING GIANIK STAR (Geometry )**

#include <bits/stdc++.h>

using namespace std;

int main() {

int n;

while (scanf("%d", &n) == 1) {

int a[512], b[512], cnt[360] = {};

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

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

a[i] = (a[i]%360 + 360)%360;

cnt[a[i]]++;

}

int ret = -1;

for (int time = 0; time <= 360 && ret == -1; time++) {

for (int i = 0; i < 360; i++) {

if (cnt[i] > 1)

ret = time;

cnt[i] = 0;

}

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

a[i] = a[i] + b[i];

a[i] = (a[i]%360 + 360)%360;

cnt[a[i]]++;

}

}

if (ret == -1)

printf("GIANIK IS NEVER ECLIPSED\n");

else

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

}

return 0;

}

# **ODD OR EVEN (Math)**

import java.util.\*;

public class Main {

public static void main(String[] args) throws Exception {

int juan = 0, maria = 0, x, n;

Scanner sc = new Scanner(System.in);

n = sc.nextInt();

while ((n > 0)) {

juan = 0;

maria = 0;

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

x = sc.nextInt();

if ((x & 1) == 1) {

maria++;

}

}

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

x = sc.nextInt();

if ((x & 1) == 0) {

juan++;

}

}

System.out.println(juan > maria ? juan - maria : maria - juan);

n=sc.nextInt();

}

}

}

# **THE MONKEY AND THE OILED BAMBOO (Math)**

#include <algorithm>

#include <cstdio>

using namespace std;

const int n\_max = 100000;

int rungs[n\_max];

int main()

{

int T;

scanf("%d", &T);

for (int t = 1; t <= T; t++) {

int n;

scanf("%d", &n);

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

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

int k = rungs[0];

for (int i = 1; i < n; i++)

k = max(k, rungs[i] - rungs[i - 1]);

int j = k;

if (rungs[0] == j)

j--;

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

int d = rungs[i] - rungs[i - 1];

if (d == j)

j--;

else if (d > j) {

k++; break;

}

}

printf("Case %d: %d\n", t, k);

}

return 0;

}

# **BIG MOD (Big Numbers)**

import java.math.BigInteger;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner lea = new Scanner (System.in);

BigInteger B,P,M;

while (lea.hasNext()){

B = lea.nextBigInteger();

P = lea.nextBigInteger();

M = lea.nextBigInteger();

System.out.println(B.modPow(P, M));

}

}

}

# **HUMIDEX (String/Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner lea = new Scanner(System.in);

while(lea.hasNext()){

char letra1 = lea.next().charAt(0);

if(letra1 == 'E'){

break;

}

boolean T = true, H = true, D = true;

double numero1 = lea.nextDouble();

char letra2 = lea.next().charAt(0);

double numero2 = lea.nextDouble();

double t = 0, h = 0, d = 0;

if(letra1 == 'T' && letra2 == 'D'){

T = false;

D = false;

t = numero1;

d = numero2;

}

else if(letra1 == 'D' && letra2 == 'T'){

T = false;

D = false;

t = numero2;

d = numero1;

}

else if(letra1 == 'T' && letra2 == 'H'){

T = false;

H = false;

t = numero1;

h = numero2;

}

else if(letra1 == 'H' && letra2 == 'T'){

T = false;

H = false;

t = numero2;

h = numero1;

}

else if(letra1 == 'D' && letra2 == 'H'){

D = false;

H = false;

d = numero1;

h = numero2;

}

else if(letra1 == 'H' && letra2 == 'D'){

D = false;

H = false;

d = numero2;

h = numero1;

}

double e = 6.11 \* Math.exp(5417.7530\*((1/273.16)-(1/(d + 273.16))));

double h1 = (0.5555) \* (e - 10.0);

if(T){

t = h - h1;

}

else if(H){

h = t + h1;

}

else if(D){

h1 = h - t;

e = (h1 / 0.5555) + 10;

d = (-1/(((Math.log(e/6.11))/5417.7530) - (1/273.16))) - 273.16;

}

System.out.println("T "+Math.rint(t\*10)/10+" D "+Math.rint(d\*10)/10+" H "+Math.rint(h\*10)/10);

}

}

}

# **I LOVE BIG NUMBERS (Big numbers)**

import java.math.BigInteger;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner lea = new Scanner(System.in);

while(lea.hasNext()){

int n = lea.nextInt();

BigInteger factorial = BigInteger.ONE;

for(long i = n; i >= 1; i--){

factorial = factorial.multiply(BigInteger.valueOf(i));

}

BigInteger p = factorial;

BigInteger suma = BigInteger.ZERO;

while(!p.equals(BigInteger.ZERO)){

suma = suma.add(p.mod(BigInteger.TEN));

p = p.divide(BigInteger.TEN);

}

System.out.println(suma);

}

}

}

# **BABELFISH (STRING)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.\*;

public class Main{

    public static void main(String[] args) throws IOException {

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        String S = br.readLine();

        Map words = new HashMap();

        while (!S.matches("")) {

            String[] Split = S.split(" ");

            words.put(Split[1], Split[0]);

            S = br.readLine();

        }

        S = br.readLine();

        boolean found = false;

        while (S != null) {

            found = false;

            if (words.containsKey(S)) {

                found = true;

                System.out.println(words.get(S));

            }

            if (!found)

                System.out.println("eh");

            S = br.readLine();

        }

    }

}

**C++**

#include <iostream>

#include <string>

#include <map>

#include <cstdio>

#include <cstring>

using namespace std;

int main(){

long i,j,k;

char s[11],ss[11],line[25];

map<string,string>m;

//freopen("i.txt","r",stdin);

while( gets(line) ){

if( strcmp(line,"") ){

sscanf(line,"%s %s",&s,&ss);

if( m[ss] == "" )

m[ss] = s;

}

else{

while( gets(line) ){

if( m[line] != "" )

cout << m[line] << '\n';

else

cout << "eh\n";

}

}

}

}

# **TOUR DE FRANCE (Math)**

import java.io.\*;

import java.util.Arrays;

import java.util.Scanner;

public class Main {

public static void main(String[] args) throws IOException {

Scanner sc = new Scanner(System.in);

StringBuilder out = new StringBuilder();

int f,r,c;

double max;

int [] front , rear;

double [] ratio;

while((f = sc.nextInt()) != 0){

r = sc.nextInt();

front = new int[f];

rear = new int[r];

ratio = new double[r \* f];

c = 0;

max = 0;

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

front[i] = sc.nextInt();

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

rear[i] = sc.nextInt();

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

for (int j = 0; j < front.length; j++)

ratio[c++] = rear[i] \* 1.0 / front[j];

Arrays.sort(ratio);

for (int i = 1; i < ratio.length; i++) {

max = Math.max(max, ratio[i] / ratio[i-1]);

}

String ans = String.format("%.2f", max);

out.append(ans + "\n");

}

System.out.print(out);

}

}

# **SCROLLING SIGN (String)**

#include <iostream>

#include <cstdio>

#include <algorithm>

#include <cstring>

#include <string>

#define INF\_MAX 2147483647

#define INF\_MIN -2147483647

#define pi acos(-1.0)

#define N 1000000

#define LL long long

#define For(i, a, b) for ( int i = (a); i < (b); i++ )

#define Fors(i, sz) for ( size\_t i = 0; i < sz.size (); i++ )

#define Set(a, s) memset (a, s, sizeof (a))

using namespace std;

string str;

string total;

int k, w;

bool isMatch (int pos)

{

for ( int i = pos; i < total.size (); i++ ) {

if ( total [i] != str [i - pos] ) return false;

}

return true;

}

void minimize ()

{

int pos = total.size () - k;

for ( int i = pos; i < total.size (); i++ ) {

if ( isMatch (i) ) {

for ( int j = total.size () - i; j < str.size (); j++ )

total += str [j];

return;

}

}

total += str;

}

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

int testCase;

scanf ("%d", &testCase);

while ( testCase-- ) {

scanf ("%d %d", &k, &w);

total = "";

For (i, 0, w) { cin >> str; minimize (); }

cout << total.size() << endl;

}

return 0;

}

# **THE HOTEL WITH INFINITE ROOMS (Math)**

#include<stdio.h>

int main()

{

long s,d,i,n;

while(scanf("%ld%ld",&s,&d)==2)

{

n=0;

for(i=s;;i++)

{

n+=i;

if(n>d || n==d)

{

printf("%ld\n",i);

break;

}

}

}

return 0;

}

# **THIS IS INTEGRATION ? (Geometry)**

#include<iostream>

#include<cmath>

#include<iomanip>

#define PI acos(-1)

using namespace std;

int main(){

double a;

while(cin>>a){

cout<<fixed<<setprecision(3)<<a\*a\*(1-sqrt(3)+PI/3)<<" "<<a\*a\*(2\*sqrt(3)-4+PI/3)<<" "<<a\*a\*(4-2\*PI/3-sqrt(3))<<endl;

}

return 0;

}

# **TROUBLE WITH A PENTAGON (Geometry)**

#include <stdio.h>

#include <math.h>

#define pi acos(-1)

#define Sin(x) sin(x\*pi/180.0)

int main() {

double f;

while(scanf("%lf", &f) == 1) {

printf("%.10lf\n", Sin(108)/Sin(63)\*f);

}

return 0;

}

# **SIMPLE BASE CONVERSION (Big numbers)**

import java.util.\*;

import java.math.\*;

class Main{

public static void main(String args[]){

Scanner sc = new Scanner (System.in);

String s;

while(sc.hasNext()){

s=sc.next();

if(s.charAt(0)=='-') break;

if(s.charAt(0)=='0'&&s.charAt(1)=='x'){

// number is in hexadecimal

BigInteger number = new BigInteger(s.substring(2),16);

System.out.println(number);

}

else{

BigInteger number = new BigInteger(s);

s=number.toString(16);

System.out.println("0x"+s.toUpperCase());

}

}

}

}

**C**

#include<stdio.h>  
#include<stdlib.h>  
int main()  
{  
    long a;  
    char b[35];  
    while(gets(b))  
    {  
        if(b[0]=='0' && b[1]=='x')  
            {  
                a=strtol(b,NULL,16);  
                printf("%ld\n",a);  
            }  
        else  
        {  
            a=atol(b);  
            if(a<0)  
                break;  
            else printf("0x%X\n",a);  
        }  
    }  
    return 0;  
}

# **COUNTING STARS (String)**

#include<iostream>

using namespace std;

int main(){

int f;

int c;

int stars;

bool star=true;

cin>>f>>c;

while(f!=0){

stars = 0;

char matriz [f][c];

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

for(int j=0;j<c;j++){

cin>>matriz[i][j];

}

}

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

for(int j=0;j<c;j++){

if(matriz[i][j] == 42){

for(int k=0;k<3;k++){

for(int l=0;l<3;l++){

if(((i+k-1)<f) && ((j+l-1)<c)){

if(((i+k-1)>-1) && ((j+l-1)>-1)){

if((matriz[i+k-1][j+l-1] == 42) && !((i+k-1)== i && (j+l-1)==j)){

star = false;

}

}

}

}

}

if(star == true){

stars++;

}

}

star= true;

}

}

cout<<stars<<endl;

cin>>f>>c;

}

return 0;

}

# **GCD (Math)**

#include <cstdio>

#include <vector>

using namespace std;

vector<vector<int> > gcds(501, vector<int>(501, 0));

int gcd(int x, int y)

{

int &g = gcds[x][y];

if (g == 0)

{

if (x == 0)

g = y;

else

g = gcd(y % x, x);

gcds[y][x] = g;

}

return g;

}

int GetG(int n)

{

int G = 0;

for(int i= 1; i < n; ++i)

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

G += gcd(i, j);

return G;

}

int main()

{

int n;

while (scanf("%d", &n), n)

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

}

# **SQUARE NUMBERS (Math)**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

int bot, top;

cin >> bot >> top;

while (bot)

{

int count(0);

int num = sqrt(bot);

if (num\*num < bot)

++num;

for (;num\*num <= top; ++num, ++count);

cout << count << '\n';

cin >> bot >> top;

}

}

# **ROBOT INSTRUCCIONS (String)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int T = sc.nextInt();

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

int N = sc.nextInt();

int[] moves = new int[N];

int ans = 0;

for (int j = 0; j < N; j++) {

String input = sc.next();

if (input.equals("LEFT")) {

moves[j] = -1;

} else if (input.equals("RIGHT")) {

moves[j] = 1;

} else {

sc.next();

moves[j] = moves[sc.nextInt() - 1];

}

ans += moves[j];

}

System.out.println(ans);

}

}

}

# **HANOI TOWER TROUBLES AGAIN! (Math)**

#include <stdio.h>

#include <math.h>

int main() {

int n, i, j;

scanf("%\*d");

while(scanf("%d", &n) == 1) {

int f[50] = {}, tmp;

f[0] = 1;

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

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

tmp = f[j]+i;

tmp = (int)sqrt(tmp);

if(tmp\*tmp == f[j]+i || f[j] == 0) {

f[j] = i;

break;

}

}

if(j == n) break;

}

printf("%d\n", i-1);

}

return 0;

}

# **THE 3N+1 PROBLEM (Math)**

#include <cstdio>

#include <algorithm>

#define long long LL

using namespace std;

int main ()

{

int i, j;

while ( scanf ("%d %d", &i, &j) != EOF ) {

int temp\_i = i;

int temp\_j = j;

if ( i > j ) swap (i, j);

int max\_cycle\_length = 0;

int cycle\_length;

while ( i <= j ) {

unsigned int n = i;

cycle\_length = 1;

while ( n != 1 ) {

if ( n % 2 == 1 ) n = 3 \* n + 1;

else n /= 2;

cycle\_length++;

}

if ( cycle\_length > max\_cycle\_length )

max\_cycle\_length = cycle\_length;

i++;

}

printf ("%d %d %d\n", temp\_i, temp\_j, max\_cycle\_length);

}

return 0;

}

# **BASE -2 (Math)**

import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.InputStreamReader;  
  
public class Main {  
  
    public static void main(String[] args) throws IOException {  
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));  
        StringBuffer sb = new StringBuffer("");  
        int cases=Integer.parseInt(br.readLine());  
        for(int i=1;i<cases+1;i++) {  
            int x=Integer.parseInt(br.readLine());  
            sb.append("Case #").append(i).append(": ").append(binary(x)).append("\n");  
        }  
        System.out.print(sb);  
    }  
      
static String binary(long n) {  
    StringBuilder temp=new StringBuilder();  
        while(true){  
            if(n==0){  
                temp.append("0");  
                break;  
            }  
            if(n==1){  
                temp.append("1");  
                break;  
            }  
            if(n==-1){  
                temp.append("11");  
                break;  
            }  
            if(n%2==0){  
               n/=-2;  
               temp.append("0");  
            }else{  
               n--;  
               n/=-2;  
               temp.append("1");   
            }  
        }  
        return temp.reverse().toString();  
    }  
  
}

# **THE BUS DRIVER PROBLEM (Sort)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.\*;

public class Main {

public static int NumberOfBusDrivers, Distance, Over;

public static int[] Morning;

public static int[] Evening;

public static void main(String args[]) throws NumberFormatException, IOException{

Scanner Input = new Scanner(System.in);

while(true){

NumberOfBusDrivers = Input.nextInt();

Distance = Input.nextInt();

Over = Input.nextInt();

if(NumberOfBusDrivers == 0 && Distance == 0 && Over == 0) break;

Morning = new int[NumberOfBusDrivers];

Evening = new int[NumberOfBusDrivers];

for(int i = 0; i < NumberOfBusDrivers; i++) Morning[i] = Input.nextInt();

for(int i = 0; i < NumberOfBusDrivers; i++) Evening[i] = -Input.nextInt();

Arrays.sort(Morning); Arrays.sort(Evening);

int Value = 0;

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

int Result = Morning[i] + (-Evening[i]);

if(Result > Distance)

Value += (Result - Distance) \* Over;

}

System.out.println(Value);

}

}

}

# **CACHO (String)**

#include <bits/stdc++.h>

using namespace std;

int main() {

int testcase;

scanf("%d", &testcase);

while (testcase--) {

int ret = 1, x, p;

for (int i = 0; i < 5; i++) {

scanf("%d", &x);

if (i)

ret &= x == p+1;

p = x;

}

puts(ret ? "Y" : "N");

}

return 0;

}

# **SODA SURPLER (Recursive math)**

import java.util.Scanner;

class uva11689{

public static void main(String args[]){

Scanner sc=new Scanner(System.in);

int cases=sc.nextInt();

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

int e=sc.nextInt(),f=sc.nextInt(),c=sc.nextInt();

int total=e+f,count=0;

while((total/c)!=0){

count=count+total/c; //???????

total=total/c+total%c; //???????

}

//Output

System.out.println(count);

}

}

}

# **BOUNDING BOX (Geometry)**

#include <cstdio>

#include <cstdlib>

#include <cmath>

#include<iostream>

#include <algorithm>

#define EPS 1e-9

using namespace std;

const double HALF\_PI = acos(0.0);

const double PI = 2\*HALF\_PI;

struct Vector2d{

double x,y;

Vector2d(){x=y=0;};

Vector2d(double \_x,double \_y):x(\_x),y(\_y){}

Vector2d rotate(double radians){

double COS = cos(radians);

double SIN = sin(radians);

return Vector2d((COS\*x-SIN\*y),

(SIN\*x+COS\*y));

}

};

struct Point{

double x,y;

Point(){x=y=0;}

Point(double \_x,double \_y):x(\_x),y(\_y){};

double det(Point P0,Point P1){

return (P1.x-P0.x)\*(x-P0.x)-(P1.y-P0.y)\*(y-P0.y);

}

bool isLeft(Point P0,Point P1){

return det(P0,P1) >= 0;

}

};

struct Line{

double A,B,C;

Line(double x1,double y1,double x2,double y2){

A = y1-y2;

B = x2-x1;

C = ((-A)\*x1)-(B\*y1);

}

Line(Point P1,Point P2){

A = P1.y-P2.y;

B = P2.x-P1.x;

C = ((-A)\*P1.x)-(B\*P1.y);

}

bool intersectingPoint(const Line& l,double& X,double& Y){

double det = A\*l.B-l.A\*B;

if(fabs(det)<EPS){

return false;

}

X = double(B\*l.C-C\*l.B)/double(det);

Y = double(C\*l.A-A\*l.C)/double(det);

return true;

}

};

int main(){

double auxx,auxy,radius,ang;

Point p1,p2,p3,c;

int sides;

int cases = 0;

while(scanf("%d",&sides) && sides && scanf("%lf%lf%lf%lf%lf%lf",&p1.x,&p1.y,&p2.x,&p2.y,&p3.x,&p3.y)==6){

Line l1(p1,p2);

Line l2(p2,p3);

auxx = (p1.x-p2.x)/2+p2.x;

auxy = (p1.y-p2.y)/2+p2.y;

Line pp1(auxx,auxy,auxx+l1.A,auxy+l1.B);

auxx = (p3.x-p2.x)/2+p2.x;

auxy = (p3.y-p2.y)/2+p2.y;

Line pp2(auxx,auxy,auxx+l2.A,auxy+l2.B);

pp1.intersectingPoint(pp2,c.x,c.y);

radius = sqrt((c.x-p1.x)\*(c.x-p1.x)+(c.y-p1.y)\*(c.y-p1.y));

double sumAng = 2\*PI/sides;

Vector2d vec(p1.x-c.x,p1.y-c.y);

double xmax, xmin, ymax, ymin;

xmax = xmin = p1.x;

ymax = ymin = p1.y;

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

vec = vec.rotate(sumAng);

double xact = vec.x+c.x;

double yact = vec.y+c.y;

xmin = min(xmin, xact);

ymin = min(ymin, yact);

xmax = max(xmax, xact);

ymax = max(ymax, yact);

}

double area = (ymax-ymin) \* (xmax-xmin);

printf("Polygon %d: %.3lf\n",++cases,area);

}

return 0;

}

# **FORWARDING EMAILS (Graph)**

import java.util.\*;

import java.io.\*;

import java.lang.\*;

public class Main

{

public static int[] graph = new int[50001];

public static int[] sum = new int[50001];

public static boolean[] visited = new boolean[50001];

public static int dfs(int u)

{

visited[u] = true;

int total = 0;

if (graph[u] != -1 && !visited[graph[u]])

{

total += 1 + dfs(graph[u]);

}

visited[u] = false;

return sum[u] = total;

}

public static void main(String[] args) throws IOException

{

//long t1 = System.nanoTime();

BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

StringTokenizer tokenizer = null;

int T = Integer.valueOf(reader.readLine());

for(int i = 1; i <= T; i++)

{

int N = Integer.valueOf(reader.readLine());

for(int j = 0 ; j < N; j++)

{

tokenizer = new StringTokenizer(reader.readLine());

int u = Integer.valueOf(tokenizer.nextToken());

int v = Integer.valueOf(tokenizer.nextToken());

graph[u] = v;

visited[u] = false;

sum[u] = -1;

}

int bestLen = 0, answer = 0;

for(int z = 1; z <= N; z++)

{

if (sum[z] == -1) dfs(z);

if (sum[z] > bestLen)

{

bestLen = sum[z];

answer = z;

}

}

System.out.println("Case " + i + ": " + answer);

}

}

}

# **BEE (Math)**

#include <cstdio>

using namespace std;

typedef long long llong;

const int MAXN = 102;

llong F[MAXN], M[MAXN];

int main(int argc, char \*argv[]) {

int N;

F[0] = 1; F[1] = 1; F[2] = 2;

M[0] = 0; M[1] = 1; M[2] = 2;

for(int i = 3; i < MAXN; ++i) {

F[i] = F[i-1] + F[i-2];

M[i] = M[i-1] + M[i-2] + 1LL;

}

while(1 == scanf("%d",&N)) {

if(N == -1) break;

printf("%lld %lld\n", M[N], M[N] + F[N]);

}

return 0;

}

# **COLA (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int N;

while (sc.hasNext()) {

N = sc.nextInt();

System.out.println(GetNum(N) + N);

}

}

static int GetNum(int empty) {

if (empty == 2) {

return 1;

} else if (empty < 2) {

return 0;

}

return GetNum(empty / 3 + empty % 3) + empty / 3;

}

}

# **RELATIONAL OPERATOR (Math )**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

long t, n1, n2;

t = sc.nextLong();

while (t != 0) {

n1 = sc.nextLong();

n2 = sc.nextLong();

if (n1 > n2) {

System.out.println(">");

} else if (n1 < n2) {

System.out.println("<");

} else {

System.out.println("=");

}

t--;

}

}

}

# **TERNARY (Math)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

while (sc.hasNext()) {

int n = sc.nextInt();

if (n < 0) {

break;

}

System.out.println(Integer.toString(n, 3));

}

}

}

# **BRICK WALL PATTERNS (Math)**

import java.util.Scanner;

public class Main {

static public long[] vector = new long[53];

public static void main(String[] args) {

int num;

Scanner sc = new Scanner(System.in);

llenar\_vec();

num = sc.nextInt();

while (num != 0) {

System.out.println(vector[num]);

num = sc.nextInt();

}

}

public static void llenar\_vec() {

for (int i = 0; i < vector.length - 1; i++) {

if (i == 0) {

vector[0] = 1;

} else if (i == 1) {

vector[1] = 1;

} else {

vector[i] = vector[i - 1] + vector[i - 2];

}

}

}

}

# **BOILED EGGS (Math)**

import java.util.Arrays;

import java.util.Scanner;

public class Main {

static public int[] vector;

static int t, n, p=0, q, timp = 1, suma = 0,res=0;

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

t = sc.nextInt();

while (t > 0) {

n = sc.nextInt();

p = sc.nextInt();

vector = new int[n];

q = sc.nextInt();

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

vector[i] = sc.nextInt();

}

Arrays.sort(vector);

if (n < p) {

p = n;

}

revisar(p);

System.out.println("Case "+timp+": "+res);

p=0;

t--;

timp++;

}

}

public static void revisar(int p) {

res=p;

suma = 0;

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

suma += vector[i];

}

if (suma > q) {

p--;

revisar(p);

}

}

}

# **COMMANDO WAR (Sorting)**

import java.util.Arrays;

import java.util.Comparator;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int k=0;

while(sc.hasNext()){

k++;

int n =sc.nextInt();

if(n==0){

break;

}

Bj bj[] = new Bj[n];

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

bj[i]=new Bj();

bj[i].b = sc.nextInt();

bj[i].j = sc.nextInt();

}

sort(bj);

int s=0;

int ans =0;

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

s = s+bj[i].b;

ans = Math.max(ans, s+bj[i].j); }

System.out.println("Case "+k+": "+ans);

}

}

private static void sort(Bj[] bj) {

Arrays.sort(bj, new Comparator<Bj>() {

@Override

public int compare(Bj o1, Bj o2) {

if(o1.j<o2.j){

return 1;

}

return -1;

}

});

}

}

class Bj{

public int b;

public int j;

}

# **THE BLOCKS PROBLEM (String-Stack)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.Stack;

import java.util.StringTokenizer;

public class Main {

Stack<Integer>[] blocks;

int[] pos;

public static void main(String[] args) throws IOException {

Main p = new Main();

p.go();

p.print();

}

private void print() {

for(int i=0; i<blocks.length; i++){

System.out.println(blockToString(i));

}

}

private String blockToString(int i) {

String ans = "";

while(!blocks[i].isEmpty())

ans = " "+ blocks[i].pop() + ans;

ans = i+ ":" + ans;

return ans;

}

@SuppressWarnings("unchecked")

public void go() throws NumberFormatException, IOException {

BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

int N = Integer.parseInt(in.readLine());

blocks = new Stack[N];

pos = new int[N];

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

blocks[i] = new Stack<Integer>();

blocks[i].push(i);

pos[i] = i;

}

String l = "";

while (!(l = in.readLine()).equals("quit")) {

StringTokenizer tokens = new StringTokenizer(l);

//Separa la linea cada vez que encuentra un espacio

String action = tokens.nextToken();

int a = Integer.parseInt(tokens.nextToken());

String type = tokens.nextToken();

int b = Integer.parseInt(tokens.nextToken());

if(a==b || pos[a] ==pos[b]) continue;

if (action.equals("move")) {

if (type.equals("onto"))

moveOnto(a, b);

else if (type.equals("over"))

moveOver(a, b);

} else if (action.equals("pile")) {

if (type.equals("onto"))

pileOnto(a, b);

else if (type.equals("over"))

pileOver(a, b);

}

}

}

public void moveOnto(int a, int b) {

clearAbove(b);

moveOver(a, b);

}

public void moveOver(int a, int b) {

clearAbove(a);

blocks[pos[b]].push(blocks[pos[a]].pop());

pos[a] = pos[b];

}

public void pileOnto(int a, int b) {

clearAbove(b);

pileOver(a, b);

}

public void pileOver(int a, int b) {

Stack<Integer> aPile = new Stack<Integer>();

while (blocks[pos[a]].peek() != a)

aPile.push(blocks[pos[a]].pop());

aPile.push(blocks[pos[a]].pop());

while (!aPile.isEmpty()) {

int top = aPile.pop();

blocks[pos[b]].push(top);

pos[top] = pos[b];

}

}

public void clearAbove(int x) {

while (blocks[pos[x]].peek() != x)

returnHome(blocks[pos[x]].pop());

}

private void returnHome(Integer x) {

while (!blocks[x].isEmpty()) {

returnHome(blocks[x].pop());

}

blocks[x].push(x);

pos[x] = x;

}

}

# **ECOLOGICAL BIN PACKING (Math-graph)**

import java.util.\*;

import java.io.\*;

public class Main {

public static void main(String[] args) throws Exception {

Scanner in = new Scanner(System.in);

PrintWriter out = new PrintWriter(System.out, true);

// types of bottles: brown, green, clear

char[] bottleType = {'B', 'G', 'C'};

// all 6 possible permutations of bottle type-to-bin mappings

int[][] permutations = {

{0, 1, 2},

{0, 2, 1},

{1, 0, 2},

{1, 2, 0},

{2, 0, 1},

{2, 1, 0}

};

// bottles[i][j] are the number of type j bottles in bin i

int[][] bottles = new int[3][3];

// while there is a test case left

while (in.hasNextInt()) {

// the minimum bottle move-count so far,

// initially -1 to indicate that we haven't found anything yet

int minCount = -1;

// the string representation of the permutation with the minimum bottle move-count so far

String minPerm = "";

// read in the bottle counts

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

bottles[i][j] = in.nextInt();

}

}

// loop through each permutation

for (int i = 0; i < permutations.length; i++) {

// count how many bottles need to be moved

int curCount = 0;

// j is the current bin, and permutations[i][j] is the bottle-type that we want in that bin

for (int j = 0; j < 3; j++) {

// go through all each bottle-type k

for (int k = 0; k < 3; k++) {

// if the current bottle type is not supposed to be in the current bin

if (permutations[i][j] != k) {

// then we need to move them, and we're counting how many we need to move

curCount += bottles[j][k];

}

}

}

// the string representing the current permutation, example: "GCB"

String curPerm = "" + bottleType[permutations[i][0]] + bottleType[permutations[i][1]] + bottleType[permutations[i][2]];

// if this is the first permutation we try, or if the current bottle move-count is less than the minimum bottle move-count,

// or the current bottle move-count is equal to the minimum bottle move-count and the permutation string is less than the

// permutation string of the minimum bottle move-count

if (minCount == -1 || curCount < minCount || (curCount == minCount && curPerm.compareTo(minPerm) < 0)) {

// then this is the best permutation so far

minCount = curCount;

minPerm = curPerm;

}

}// print the minimum bottle move-count and the permutation string

out.println(minPerm + " " + minCount);

}

}

}

# **TEX QUOTES (String)**

import java.util.ArrayList;

import java.util.Scanner;

public class Main {

static boolean opened = false;

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

ArrayList<String> sol = new ArrayList<String>();

while (in.hasNext()) {

sol.add(convert(in.nextLine()));

}

for (String s : sol)

System.out.println(s);

}

private static String convert(String nextLine) {

StringBuffer buf = new StringBuffer();

for (char c : nextLine.toCharArray()) {

if (c == '\"') {

if (opened)

buf.append("''");

else

buf.append("``");

opened = !opened;

}

else

buf.append(c);

}

return buf.toString();

}

}

# [**MACHINED S**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=355)**URFACES (Math)**

import java.util.Arrays;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

int vector[];

Scanner sc = new Scanner(System.in);

int p = 0, y = 0;

int primer = sc.nextInt();

String cada = "";

vector = new int[primer];

while (primer != 0) {

sc.nextLine();

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

cada = sc.nextLine();

for (int j = 0; j < cada.length(); j++) {

if (cada.charAt(j) == ' ') {

p++;

}

}

vector[i] = p;

p = 0;

}

Arrays.sort(vector);

for (int i = 0; i < vector.length; i++) {

y = y + Math.abs(vector[0] - vector[i]);

}

System.out.println(y);

y = 0;

primer = sc.nextInt();

vector = new int[primer];

}

}

}

# **ROTATING SENTENCES (String)**

import java.util.\*;

import java.util.Scanner;

public class Main {

public static Scanner sc = new Scanner(System.in);

public static String linea = new String();

public static char[][] cas = new char[101][101];

public static int contadorLineas = 0, espacios = 0;

public static char[] ey;

public static Integer[] can = new Integer[101];

public static List<Integer> list = new ArrayList<Integer>();

public static void main(String[] args) {

while (sc.hasNextLine()) {

linea = sc.nextLine();

if (linea.isEmpty() == false) {

cas[contadorLineas] = linea.toCharArray();

list.add(linea.length());

contadorLineas++;

espacios++;

} else {

break;

}

}

Collections.sort(list);

Collections.reverse(list);

//for (Integer x : list) {

// System.out.println(list.get(x));

//}

for (int i = 0; i < list.get(0); i++) {

for (int j = espacios-1; j >= 0; j--) {

try {

System.out.print(cas[j][i]);

} catch (ArrayIndexOutOfBoundsException e) {

System.out.print(" ");

}

}

System.out.println("");

}

}

}

# **ARTIFICIAL INTELLIGENCE? (String)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

char problemas[];

String problema, numeros1 = "", numeros2 = "", numeros3 = "";

int casos, c = 1;

boolean codigo1 = false, codigo2 = false, codigo3 = false;

double a = 0.0, b = 0.0, d = 0.0;

casos = sc.nextInt();

sc.nextLine();

while (c != casos + 1) {

problema = sc.nextLine();

problemas = new char[problema.length()];

problemas = problema.toCharArray();

for (int i = 0; i < problemas.length; i++) {

if (problemas[i] == 'U' && problemas[i + 1] == '=') {

for (int j = i + 2; j < problemas.length; j++) {

if (problemas[j] == 'V') {

break;

} else {

numeros1 = numeros1 + Character.toString(problemas[j]);

}

}

if (numeros1.charAt(numeros1.length() - 1) == 'm') {

numeros1 = numeros1.replace('m', ' ');

a = Double.parseDouble(numeros1);

a = a \* 0.001;

} else if (numeros1.charAt(numeros1.length() - 1) == 'k') {

numeros1 = numeros1.replace('k', ' ');

a = Double.parseDouble(numeros1);

a = a \* 1000;

} else if (numeros1.charAt(numeros1.length() - 1) == 'M') {

numeros1 = numeros1.replace('M', ' ');

a = Double.parseDouble(numeros1);

a = a \* 1000000;

} else {

a = Double.parseDouble(numeros1);

}

codigo1 = true;

} else if (problemas[i] == 'I' && problemas[i + 1] == '=') {

for (int j = i + 2; j < problemas.length; j++) {

if (problemas[j] == 'A') {

break;

} else {

numeros2 = numeros2 + Character.toString(problemas[j]);

}

}

if (numeros2.charAt(numeros2.length() - 1) == 'm') {

numeros2 = numeros2.replace('m', ' ');

b = Double.parseDouble(numeros2);

b = b \* 0.001;

} else if (numeros2.charAt(numeros2.length() - 1) == 'k') {

numeros2 = numeros2.replace('k', ' ');

b = Double.parseDouble(numeros2);

b = b \* 1000;

} else if (numeros2.charAt(numeros2.length() - 1) == 'M') {

numeros2 = numeros2.replace('M', ' ');

b = Double.parseDouble(numeros2);

b = b \* 1000000;

} else {

b = Double.parseDouble(numeros2);

}

codigo2 = true;

} else if (problemas[i] == 'P' && problemas[i + 1] == '=') {

for (int j = i + 2; j < problemas.length; j++) {

if (problemas[j] == 'W') {

break;

} else {

numeros3 = numeros3 + Character.toString(problemas[j]);

}

}

if (numeros3.charAt(numeros3.length() - 1) == 'm') {

numeros3 = numeros3.replace('m', ' ');

d = Double.parseDouble(numeros3);

d = d \* 0.001;

} else if (numeros3.charAt(numeros3.length() - 1) == 'k') {

numeros3 = numeros3.replace('k', ' ');

d = Double.parseDouble(numeros3);

d = d \* 1000;

} else if (numeros3.charAt(numeros3.length() - 1) == 'M') {

numeros3 = numeros3.replace('M', ' ');

d = Double.parseDouble(numeros3);

d = d \* 1000000;

} else {

d = Double.parseDouble(numeros3);

}

codigo3 = true;

}

}

if (codigo1 == true && codigo2 == true) {

System.out.println("Problem #" + c);

System.out.printf("P=%.2f", a \* b);

System.out.println("W");

} else if (codigo2 == true && codigo3 == true) {

System.out.println("Problem #" + c);

System.out.printf("U=%.2f", d / b);

System.out.println("V");

} else if (codigo1 == true && codigo3 == true) {

System.out.println("Problem #" + c);

System.out.printf("I=%.2f", d / a);

System.out.println("A");

}

System.out.println("");

codigo1 = false;

codigo2 = false;

codigo3 = false;

a = 0.0;

b = 0.0;

d = 0.0;

numeros1 = "";

numeros2 = "";

numeros3 = "";

c++;

}

}

}

# [**THE DECODER**](https://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=399) **JAVA (String)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

String entrada, n;

int x = 0;

while (sc.hasNext()) {

entrada = sc.nextLine();

for (int i = 0; i < entrada.length(); i++) {

x = entrada.charAt(i) - 7;

char ch = (char) (x);

if (entrada.charAt(i) == 10) {

System.out.print(10);

} else {

System.out.print(ch);

}

}

}

}

//solucion pro:

/\*import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

public class Main {

public static void main(String[] args) throws IOException {

DataInputStream in = new DataInputStream(System.in);

DataOutputStream output = new DataOutputStream(System.out) ;

int C;

while ((C =in.read()) !=-1)

output.write((Character.isSpace((char)C) ? C : (C-7)));

}

}\*/

}

# **LINEAL CELULAR AUTOMATA (String)**

import java.io.IOException;

import java.util.Scanner;

public class Main {

public static void main(String[] args) throws IOException {

Scanner sc = new Scanner(System.in);

final char[] den = {' ', '.', 'x', 'W'};

int[] DNA = new int[10];

int[] current, next;

int tests = sc.nextInt();

int i, j, k;

for (int test = 0; test < tests; test++) {

for (i = 0; i < 10; i++) {

DNA[i] = sc.nextInt();

}

current = new int[40];

current[19] = 1;

for (i = 0; i < 50; i++) {

next = new int[40];

for (j = 0; j < 40; j++) {

System.out.print(den[current[j]]);

}

System.out.println();

for (j = 1; j < 39; j++) {

k = current[j] + current[j - 1] + current[j + 1];

next[j] = DNA[k];

}

next[0] = DNA[current[0] + current[1]];

next[39] = DNA[current[38] + current[39]];

current = next.clone();

}

if (test < tests - 1) {

System.out.println();

}

}

}

}

# **FACTORS AND FACTORIALS (Math/PrintFormat)**

import java.io.IOException;

import java.math.BigInteger;

import java.util.Scanner;

public class Main {

static BigInteger[] factoriales = new BigInteger[101];

static String[] res = new String[101];

static int contador, primo = 2, quince = 0;

public static void main(String[] args) throws IOException {

Scanner sc = new Scanner(System.in);

int entrada;

fact();

entrada = sc.nextInt();

while (entrada != 0) {

int tamano = String.valueOf(entrada).length();

if (tamano == 1) {

System.out.println(" " + entrada + "! =" + res[entrada]);

} else if (tamano == 2) {

System.out.println(" " + entrada + "! =" + res[entrada]);

} else {

System.out.println("" + entrada + "! =" + res[entrada]);

}

entrada = sc.nextInt();

}

// System.out.println("");

}

public static void fact() {

factoriales[1] = BigInteger.ONE;

BigInteger numero;

for (int i = 2; i < 101; i++) {

factoriales[i] = factoriales[i - 1].multiply(BigInteger.valueOf(i));

//System.out.println(factoriales[i]);

numero = factoriales[i];

String aux = "";

primo = 2;

contador = 0;

quince = 0;

while (!numero.equals(BigInteger.ONE)) {

if (numero.mod(BigInteger.valueOf(primo)).equals(BigInteger.ZERO)) {

numero = numero.divide(BigInteger.valueOf(primo));

contador++;

} else {

int tamano = String.valueOf(contador).length();

if (quince == 15) {

aux += "\n ";

}

if (tamano == 1) {

aux += " " + contador;

quince++;

} else {

aux += " " + contador;

quince++;

}

contador = 0;

primo = BigInteger.valueOf(primo).nextProbablePrime().intValue();

}

}

int tamano = String.valueOf(contador).length();

if (tamano == 1) {

res[i] = aux + " " + contador;

} else {

res[i] = aux + " " + contador;

}

}

for (int i = 53; i < 59; i++) {

String cambio = res[i];

String cambio2 = cambio.substring(0, cambio.length() - 3) + "\n 1";

// System.out.println(cambio2);

res[i] = cambio2;

}

}

}

# **SAFE SALUTATIONS (Math)**

ANSI C

#include <stdio.h>

int main(){

//Sumatoria Cn=((c\*((2\*n)+1))/(n+2))\*Cn sub -1

int n,i=0;

int sol[]={1,1,2,5,14,42,132,429,1430,4862,16796};

while(scanf("%d",&n)==1){

if(i==1)

printf("\n");

printf("%d\n",sol[n]);

i=1;

}

return 0;

}

# **GRAMMAR EVALUATIONS (STRING)**

import java.util.Scanner;

import javax.script.ScriptEngine;

import javax.script.ScriptEngineManager;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int tc = sc.nextInt();

while (tc-->0) {

try {

ScriptEngineManager mgr = new ScriptEngineManager();

ScriptEngine engine = mgr.getEngineByName("JavaScript");

String formula = sc.next();

System.out.println(engine.eval(formula));

} catch (Exception e) {

System.out.println("ERROR");

}

}

}

}

# **LIST OF CONQUEST (String)**

import java.util.\*;

class Main {

public static void main(String args[]) throws Exception {

Scanner s = new Scanner(System.in);

int n = s.nextInt();

HashMap hm = new HashMap(n);

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

String t = s.nextLine();

if (t.equals("")) {

continue;

}

String[] names = t.split(" ");

String country = names[0];

String woman\_name = getWomanName(names);

if (hm.containsKey(country)) {

data d = (data) hm.get(country);

if (!d.names.contains(woman\_name)) {

d.names.add(woman\_name);

d.count++;

hm.remove(country);

hm.put(country, d);

}

} else {

data d1 = new data();

d1.names.add(woman\_name);

d1.count++;

hm.put(country, d1);

}

}

String[] keys = (String[]) hm.keySet().toArray(new String[0]);

Arrays.sort(keys);

for (String key : keys) {

int count = ((data) hm.get(key)).count;

System.out.println(key + " " + count);

}

}

static String getWomanName(String[] names) {

String n = "";

for (int i = 1; i < names.length; i++) {

n = n + names[i] + " ";

}

n = n.trim();

return n;

}

}

class data {

ArrayList<String> names = new ArrayList(5);

int count = 0;

}

# **BEAT THE SPREAD! (Math)**

import java.util.Scanner;

import java.util.StringTokenizer;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

StringTokenizer st;

int n = Integer.parseInt(sc.nextLine());

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

st = new StringTokenizer(sc.nextLine());

int sum = Integer.parseInt(st.nextToken());

int diff = Integer.parseInt(st.nextToken());

if (sum < diff || (sum + diff) % 2 != 0) {

System.out.println("impossible");

} else {

int x = (sum + diff) / 2;

int y = (sum - diff) / 2;

System.out.println(Math.max(x, y) + " " + Math.min(x, y));

}

}

}

}

# **RAT ATTACK (Data Structure)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.StringTokenizer;

class Main {

public static void main(String[] abc) throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int testCaseCount = Integer.parseInt(br.readLine());

for (int testCase = 0; testCase < testCaseCount; testCase++) {

int n = Integer.parseInt(br.readLine());

int ratPop = Integer.parseInt(br.readLine());

int[][] ratCount = new int[1025][1025];

int minx = 1024;

int maxx = 0;

int miny = 1024;

int maxy = 0;

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

StringTokenizer st = new StringTokenizer(br.readLine());

int x = Integer.parseInt(st.nextToken());

int y = Integer.parseInt(st.nextToken());

int count = Integer.parseInt(st.nextToken());

for (int tempx = Math.max(0, x - n); tempx <= Math.min(ratCount.length - 1, x + n); tempx++) {

for (int tempy = Math.max(0, y - n); tempy <= Math.min(ratCount[tempx].length - 1, y + n); tempy++) {

ratCount[tempx][tempy] += count;

}

}

minx = Math.min(minx, Math.max(0, x - n));

maxx = Math.max(maxx, Math.min(ratCount.length - 1, x + n));

miny = Math.min(miny, Math.max(0, y - n));

maxy = Math.max(maxy, Math.min(ratCount.length - 1, y + n));

}

int[] bestCase = new int[3];

for (int x = minx; x <= maxx; x++) {

for (int y = miny; y <= maxy; y++) {

if (ratCount[x][y] > bestCase[2]) {

bestCase[0] = x;

bestCase[1] = y;

bestCase[2] = ratCount[x][y];

}

}

}

System.out.println(bestCase[0] + " " + bestCase[1] + " " + bestCase[2]);

}

}

}

# **MACARONS (Graph)**

#include <bits/stdc++.h>

using namespace std;

const int MAXM = 256;

const int MOD = 1e+9;

struct Matrix {

static int n;

int64\_t v[MAXM\*MAXM];

void init(int val=0) {

memset(v, 0, sizeof(v[0])\*n\*n);

if (val) {

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

v[i\*n+i] = val;

}

}

};

void square(Matrix &x) {

static int64\_t trans\_v[MAXM\*MAXM], v[MAXM\*MAXM];

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

for (int j = 0; j < Matrix::n; j++)

trans\_v[i\*Matrix::n+j] = x.v[j\*Matrix::n+i];

}

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

for (int j = 0; j < Matrix::n; j++) {

int64\_t sum = 0;

for (int k = 0; k < Matrix::n; k++) {

sum += x.v[i\*Matrix::n+k] \* trans\_v[j\*Matrix::n+k];

if (sum >= MOD)

sum %= MOD;

}

v[i\*Matrix::n+j] = sum;

}

}

memcpy(x.v, v, sizeof(v[0])\*Matrix::n\*Matrix::n);

}

int Matrix::n = 0;

int main() {

int n;

int64\_t m;

static Matrix y;

while (scanf("%d %lld", &n, &m) == 2) {

assert(n <= 8);

const int t = 1<<n;

Matrix::n = t;

y.init();

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

for (int j = 0; j < t; j++) {

int dp[16][2] = {};

dp[0][1] = 1;

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

if ((i>>k)&1) {

if ((j>>k)&1) {

dp[k+1][0] = dp[k][1];

dp[k+1][1] = dp[k][0];

} else {

dp[k+1][1] = dp[k][1];

}

} else {

if ((j>>k)&1) {

dp[k+1][0] = dp[k][1];

dp[k+1][1] = dp[k][0] + dp[k][1];

} else {

dp[k+1][1] = dp[k][1];

}

}

}

y.v[i\*t+j] = dp[n][1];

}

}

int64\_t A[256] = {}, B[256];

A[t-1] = 1;

while (m) {

if (m&1) {

memset(B, 0, sizeof(B[0])\*t);

for (int p = 0; p < t; p++) {

for (int q = 0; q < t; q++) {

B[q] += y.v[p\*t+q] \* A[p];

if (B[q] >= MOD)

B[q] %= MOD;

}

}

memcpy(A, B, sizeof(B[0])\*t);

}

square(y), m >>= 1;

}

int64\_t ret = 0;

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

ret += A[i];

ret %= MOD;

printf("%lld\n", ret);

}

return 0;

}

/\*

2

2

2

4

\*/

# **SCARECROW (String)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

class Main {

public static void main(String[]args) throws IOException {

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int testCaseCount=Integer.parseInt(br.readLine());

for (int testCase=1;testCase<=testCaseCount;testCase++) {

int N=Integer.parseInt(br.readLine());

String s=br.readLine();

int start=0;

int count=0;

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

if (i-start==1) {

start=i+2;

i++;

count++;

} else {

if (s.charAt(i)=='#') {

start=i+1;

}

}

}

if (start==N-1) {

count++;

}

System.out.println("Case "+testCase+": "+count);

}

}

}

# **GREY CODES (Math-bits)**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int tn;

scanf("%d", &tn);

while(tn--)

{

int n, k;

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

int res=k^(k>>1);

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

}

return 0;

}

# **BE EFICCIENT (Math)**

//ANSI C

#include <stdio.h>

int X[10005];

int main() {

int A, B, C, M, N;

int testcase, cases = 0;

int i;

scanf("%d", &testcase);

while(testcase--) {

scanf("%d %d %d %d %d", &A, &B, &C, &M, &N);

X[0] = A;

for(i = 1; i < N; i++)

X[i] = (X[i-1]\*B+C)%M+1;

int sum = 0, mark[10005] = {};

int ret = 0;

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

sum = (sum+X[i])%M;

ret += mark[sum];

if(sum == 0) ret++;

mark[sum]++;

}

printf("Case %d: %d\n", ++cases, ret);

}

return 0;

}

# **EIGENSECUENCE (Graph)**

#include <cstdio>

#include <cstring>

typedef long long ll;

const int N = 50;

int s, e;

ll dp[N][N];

ll solve () {

memset(dp, 0, sizeof(dp));

dp[1][s] = 1;

for (int i = 2; i < N; i++) {

for (int j = s+i-2; j < N; j++) {

if (dp[i-1][j]) {

for (int k = j+1; k < N; k++) {

if (k % (k - j))

continue;

dp[i][k] += dp[i-1][j];

}

}

}

}

ll ans = 0;

for (int i = 1; i < N; i++)

ans += dp[i][e];

return ans;

}

int main () {

while (scanf("%d%d", &s, &e) == 2 && s + e) {

printf("%d %d %lld\n", s, e, solve());

}

return 0;

}

# **ARRANGE SOME MARBLES (Graph)**

#include <cstdio>

#include <cstring>

#include <algorithm>

using namespace std;

const int maxn = 4500;

const int maxs = 5;

const int maxc = 5;

int N, PS, PC;

int num[maxc];

int f[maxn][maxs][maxc][maxs][maxc];

int dp (int state, int s, int c) {

int& ans = f[state][PS][PC][s][c];

if (ans != -1)

return ans;

if (!state) {

if (PS != s && PC != c)

return ans = 1;

return ans = 0;

}

int tmp[maxc];

int tS = state;

for (int i = N - 1; i >= 0; i--) {

if (tS >= (1<<(3\*i))) {

tmp[i] = tS/(1<<(3\*i));

tS %= (1<<(3\*i));

} else

tmp[i] = 0;

}

ans = 0;

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

if (i == c)

continue;

for (int j = 1; j <= min(3, tmp[i]); j++) {

if (j == s)

continue;

ans += dp(state - (j \* (1<<(3\*i))), j, i);

}

}

return ans;

}

void solve () {

scanf ("%d", &N);

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

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

int state = 0;

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

state += num[i] \* (1<<(3\*i));

int ans = 0;

if (state) {

for (int c = 0; c < N; c++)

for (int s = 1; s <= min(num[c], 3); s++) {

PS = s;

PC = c;

ans += dp(state - s \* (1<<(3\*c)), s, c);

}

} else

ans = 1;

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

}

int main () {

int T;

scanf ("%d", &T);

memset (f, -1, sizeof(f));

while (T--) {

solve();

}

return 0;

}

# **ZEROES III (Graph)**

#include <bits/stdc++.h>

using namespace std;

// O(n)

long long dp[1048576] = {};

long long count(int n, int p) {

long long ret = 0;

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

dp[i] = dp[i/p] + i/p;

ret += (long long) dp[i] \* (n-i+1);

}

return ret;

}

// O(log n)

long long f2(int n, int p) {

long long ret = 0, tp = p, cnt = 1;

while (n >= tp) {

for (int i = 1; n - tp\*i+1 > 0; i++) {

if (i%p == 0)

continue;

long long st = n - tp\*i + 1, ed = 1;

ret += (st + ed) \* (st-ed+1)/2 \* cnt;

}

tp \*= p;

cnt ++;

}

return ret;

}

int main() {

int n, b;

while (scanf("%d %d", &n, &b) == 2 && n) {

int mxp = 1;

long long ret = LONG\_LONG\_MAX;

for (int i = 2; i <= b; i++) {

if (b%i == 0) {

mxp = i;

int cnt = 0;

while (b%i == 0)

b /= i, cnt++;

// printf("A:%lld\nB:%lld\n", count(n, i), f2(n, i));

// ret = min(ret, count(n, i)/cnt);

ret = min(ret, f2(n, i)/cnt);

}

}

printf("%lld\n", ret);

}

return 0;

}

# **COST CUTTING (Math)**

#include<iostream>

#include <algorithm>

#include <vector>

using namespace std;

int main (){

int t,a,b,c,max,min,mid,cases=1;

cin>>t;

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

cin>>a>>b>>c;

if(a>b&&a>c){

max=a;

if(b>c){

mid=b;

min=c;

}else{

mid=c;

min=b;

}

}

if(b>a&&b>c){

max=b;

if(a>c){

mid=a;

min=c;

}else{

mid=c;

min=a;

}

}if(c>a&&c>b){

max=c;

if(a>b){

mid=a;

min=b;

}else{

mid=b;

min=a;

}

}

cout<<"Case "<<cases<<": "<<mid<<endl;

cases++;

}

}

# **MINESWEEPER (String)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main {

public static void main(String[] args) throws NumberFormatException,

IOException {

BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

String newLine = in.readLine();

String[] dimensions = newLine.split("\\s");

int m = Integer.parseInt(dimensions[0]);

int n = Integer.parseInt(dimensions[1]);

int[][] board = new int[m][n];

int counter = 1;

while (m != 0 && n != 0) {

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

newLine = in.readLine();

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

if (newLine.charAt(j) == '\*') {

board[i][j] = -1;

fillBoard(board, i, j, m, n);

}

}

}

System.out.println("Field #" + counter++ + ":");

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

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

if (board[i][j] != -1) {

System.out.print(board[i][j]);

} else {

System.out.print("\*");

}

}

System.out.print("\n");

}

newLine = in.readLine();

dimensions = newLine.split("\\s");

m = Integer.parseInt(dimensions[0]);

n = Integer.parseInt(dimensions[1]);

board = new int[m][n];

if (m != 0 && n != 0) {

System.out.print("\n");

}

}

}

public static void fillBoard(int[][] board, int i, int j, int m, int n) {

if (j < n - 1) {

if (board[i][j + 1] != -1) {

board[i][j + 1]++;

}

if (i > 0) {

if (board[i - 1][j + 1] != -1) {

board[i - 1][j + 1]++;

}

}

if (i < m - 1) {

if (board[i + 1][j + 1] != -1) {

board[i + 1][j + 1]++;

}

}

}

if (j > 0) {

if (board[i][j - 1] != -1) {

board[i][j - 1]++;

}

if (i > 0) {

if (board[i - 1][j - 1] != -1) {

board[i - 1][j - 1]++;

}

}

if (i < m - 1) {

if (board[i + 1][j - 1] != -1) {

board[i + 1][j - 1]++;

}

}

}

if (i > 0) {

if (board[i - 1][j] != -1) {

board[i - 1][j]++;

}

}

if (i < m - 1) {

if (board[i + 1][j] != -1) {

board[i + 1][j]++;

}

}

}

}

# **THE SNAIL (Math)**

#include<bits/stdc++.h>

using namespace std;

int main(){

double h,u,d,f,f2,actual=0;

int res=0;

cin>>h>>u>>d>>f;

while (h!=0){

f=f/100;

f2=u\*f;

while (true){

res++;

if(u>=0) actual+=u;

if (actual>h){

cout<<"success on day "<<res<<"\n";

break;

}

actual-=d;

if(actual<0){

cout<<"failure on day "<<res<<"\n";

break;

}

u=u-f2;

}

res=0;

actual=0;

cin>>h>>u>>d>>f;

}

}

# **HORROR DASH (Sort)**

#include<bits/stdc++.h>

using namespace std;

int main(){

int t,personas,aux,cases=1;

cin>>t;

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

cin>>personas;

vector<int> velocidades;

for (int j=0;j<personas;j++){

cin>>aux;

velocidades.push\_back(aux);

}

sort(velocidades.rbegin(),velocidades.rend());

cout<<"Case "<<cases<<": "<<velocidades[0]<<"\n";

cases++;

}

}

# **DIVISION OF NLOGONIA (Geometric)**

#include<bits/stdc++.h>

using namespace std;

int main(){

int k,n,m,x,y;

vector<string> salidas;

double numerador, denominador, pendiente;

cin>>k;

while (k!=0){

cin>>n>>m;

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

cin>>x>>y;

numerador=(m-y);

denominador=(n-x);

pendiente=numerador/denominador;

if(m==y&&n==x){

salidas.push\_back("divisa");

continue;

}if (denominador!=0){

if(pendiente==0){

salidas.push\_back("divisa");

continue;

}

if( pendiente>0){

if(m>y){

salidas.push\_back("SO");

continue;

}else{

salidas.push\_back("NE");

continue;

}

}

if(pendiente<0){

if(m>y){

salidas.push\_back("SE");

continue;

}else{

salidas.push\_back("NO");

continue;

}

}

}else {

salidas.push\_back("divisa");

continue;

}

}

for (int k=0;k<salidas.size();k++){

cout<<salidas[k]<<"\n";

}

salidas.clear();

cin>>k;

}

}

# **PESKY PALINDROMES(String)**

import java.util.HashSet;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner s = new Scanner(System.in);

String entrada, sub1, sub2 = "";

HashSet<String> hs = new HashSet<>();

while (s.hasNext()) {

entrada = s.nextLine();

for (int i = 0; i < entrada.length(); i++) {

hs.add(entrada.charAt(i) + "");

}

for (int i = 0; i < entrada.length(); i++) {

for (int j = i ; j < entrada.length(); j++) {

sub1 = entrada.substring(i, j+1);

for (int k = sub1.length(); k > 0; k--) {

sub2 += sub1.charAt(k - 1) + "";

//System.out.println(sub2);

}

if (sub1.equals(sub2)) {

hs.add(sub2);

}

sub2 = "";

}

}

System.out.println("The string \'" + entrada + "\' contains " + (hs.size()) + " palindromes.");

hs.clear();

}

}

}

# **COWS IN A ROW (Brute force)**

#include <cstdio>

int id[1005];

int get\_largest\_block(int n, int idignore) {

int maxBlockSize = 0;

int curBreed = -1;

int curSize = 0;

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

if(id[i] != idignore) {

if(curBreed == id[i]) {

curSize++;

} else {

curBreed = id[i];

curSize = 1;

}

if(curSize > maxBlockSize)

maxBlockSize = curSize;

}

}

return maxBlockSize;

}

int main() {

freopen("cowrow.in","r",stdin);

freopen("cowrow.out","w",stdout);

int n;

scanf("%d", &n);

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

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

}

int ans = 0;

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

int size = get\_largest\_block(n, id[i]);

if(size > ans)

ans = size;

}

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

}

# **REQUEST FOR PROPOSAL (Math)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.StringTokenizer;

class Main {

public static void main (String [] abc) throws IOException {

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

String s;

int testCase=1;

while ((s=br.readLine())!=null && !s.equals("0 0")) {

StringTokenizer st=new StringTokenizer(s);

int n=Integer.parseInt(st.nextToken());

int p=Integer.parseInt(st.nextToken());

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

br.readLine();

//doesn't matter as those in RFP are from here.

}

String selectedCarName="";

int selectedCarReqCount=0;

double selectedCarPrice=0.0;

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

String currCarName=br.readLine();

st=new StringTokenizer(br.readLine());

double currCarPrice=Double.parseDouble(st.nextToken());

int currCarReqCount=Integer.parseInt(st.nextToken());

for (int i2=0;i2<currCarReqCount;i2++) {

br.readLine();

//doesn't matter as those in RFP are from req list.

}

if (currCarReqCount>selectedCarReqCount || (currCarReqCount==selectedCarReqCount && selectedCarPrice>currCarPrice)) {

selectedCarName=currCarName;

selectedCarReqCount=currCarReqCount;

selectedCarPrice=currCarPrice;

}

}

if (testCase > 1) {

System.out.println();

}

System.out.println("RFP #"+(testCase++)+"\n"+selectedCarName);

}

}

}

# **EVENT PLANING (Math)**

import java.io.BufferedReader;

import java.io.InputStreamReader;

public class Main {

public static void main(String args[] ) throws Exception {

BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

String str;

while (true) {

str = in.readLine();

if(str==null || str.isEmpty()) break;

String[] temp = str.split(" ");

int no\_of\_par = Integer.parseInt(temp[0]);

int bud = Integer.parseInt(temp[1]);

int hotels =Integer.parseInt(temp[2]);

int weeks = Integer.parseInt(temp[3]);

int[] result = new int[hotels];

int count = 0;

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

{

int price\_for\_one = Integer.parseInt(in.readLine());

String[] beds = in.readLine().split(" ");

boolean valid = false;

for(int j=0;j<weeks;j++)

{

if (Integer.parseInt(beds[j]) >= no\_of\_par )

{

if(no\_of\_par \* price\_for\_one <= bud)

{

result[count] = no\_of\_par \* price\_for\_one;

count++;

break;

}

}

}

}

int res =result[0];

if (count > 0) {

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

res = Math.min(res,result[i]);

System.out.println(res);

}

else

System.out.println("stay home");

}

}

}

# **LOANSOME CAR BUYER (Math)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.StringTokenizer;

class Main {

public static void main (String [] abc) throws IOException {

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

String s;

//duration-down payment-loan amount-depreciation months

//month-num value

while ((s=br.readLine())!=null) {

StringTokenizer st=new StringTokenizer(s);

int duration=Integer.parseInt(st.nextToken());

if (duration<0) {

break;

}

double downPayment=Double.parseDouble(st.nextToken());

double loanAmount=Double.parseDouble(st.nextToken());

int dpmn=Integer.parseInt(st.nextToken());

double [] values=new double [duration+1];

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

st=new StringTokenizer(br.readLine());

int month=Integer.parseInt(st.nextToken());

double rate=1-Double.parseDouble(st.nextToken());

for (int i2=month;i2<values.length;i2++) {

values[i2]=rate;

}

}

double monthlyPayment=loanAmount/duration;

double owe=loanAmount;

double carValue=(loanAmount+downPayment)\*values[0];

int month=1;

while (owe > carValue && month<values.length) {

owe-=monthlyPayment;

carValue\*=values[month];

month++;

}

month--;

if (month==1) {

System.out.println("1 month");

} else {

System.out.println(month+" months");

}

}

}

}

# **EXPRESIONS (Tree)**

import java.util.Scanner;

import java.util.Stack;

public class Main {

public Main() {

resultados = new String[10000];

for (int i = 0; i < 10000; i++) {

resultados[i]="";

}

int casos;

Scanner s = new Scanner(System.in);

casos = s.nextInt();

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

String procedimiento = s.next();

Stack<Nodo> pila = new Stack<>();

for (int j = 0; j < procedimiento.length(); j++) {

if (procedimiento.charAt(j) > 96) {

pila.push(new Nodo(procedimiento.charAt(j)));

} else {

pila.push(new Nodo(pila.pop(), pila.pop(), procedimiento.charAt(j)));

}

}

pila.peek().asignarNivel(-1);

for (int j = maxNivel; j >= 0; j--) {

for (int k = resultados[j].length()-1; k >= 0; k--) {

System.out.print(resultados[j].charAt(k));

}

resultados[j] = "";

}

System.out.println("");

maxNivel=0;

}

}

public String resultados[];

public int maxNivel = 0;

public class Nodo {

public Nodo i, d;

public char valor;

public int nivel;

@Override

public String toString() {

return valor + "";

}

public Nodo(char valor) {

i = null;

d = null;

this.valor = valor;

}

public Nodo(Nodo d, Nodo i, char valor) {

this.i = i;

this.d = d;

this.valor = valor;

}

public void asignarNivel(int nivelPadre) {

this.nivel = nivelPadre + 1;

if (this.nivel > maxNivel) {

maxNivel = this.nivel;

}

resultados[this.nivel] += valor;

try {

this.i.asignarNivel(this.nivel);

} catch (NullPointerException e) {

}

try {

this.d.asignarNivel(this.nivel);

} catch (NullPointerException e) {

}

}

}

public static void main(String[] args) {

new Main();

}

}

# **PARENTHESES BALANCE (String-Queue)**

import java.util.LinkedList;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int t = 0;

t = sc.nextInt();

sc.nextLine();

int cont2 = 1;

String caracter;

//Cola de ()

LinkedList<String> cola = new LinkedList<>();

//Cola de []

LinkedList<String> cola2 = new LinkedList<>();

int cont = 0;

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

String Entrada = sc.nextLine();

if (Entrada.equals("")) {

//System.out.println(cont2++ + "Yes" + " " + Entrada);

System.out.println("Yes");

} else if (Entrada.contains("(]") || Entrada.contains("[)")) {

System.out.println("No");

} else {

for (int j = 0; j < Entrada.length(); j++) {

caracter = Entrada.charAt(j) + "";

if (caracter.equals("(")) {

//Meter los ( en la cola

cola.offer("(");

cont--;

} else if (caracter.equals("[")) {

//Meter los [ en la cola

cola2.offer("[");

cont--;

} else if (caracter.equals(")") && !cola.isEmpty()) {

cola.poll();

cont++;

} else if (caracter.equals("]") && !cola2.isEmpty()) {

cola2.poll();

cont++;

} else if ((caracter.equals("]") && cola2.isEmpty()) || ((caracter.equals(")") && cola.isEmpty()))) {

cont++;

}

}

if (cola.isEmpty() && cola2.isEmpty() && cont == 0) {

//System.out.println(cont2++ + "Yes" + " " + Entrada);

System.out.println("Yes");

} else {

System.out.println("No");

// System.out.println(cont2++ + "No" + " " + Entrada);

}

}

cola.clear();

cola2.clear();

cont = 0;

}

}

}

# **COLLECTING BEEPERS (Graph)**

import java.util.\*;

class Main {

final static int MAX = 11;

public static void main(String... args) {

Scanner sc = new Scanner(System.in);

for (int t = 0, T = sc.nextInt(); t < T; t++) {

int x = sc.nextInt(), y = sc.nextInt();

int[][] coords = new int[MAX][2];

coords[0][0] = sc.nextInt();

coords[0][1] = sc.nextInt();

int n = sc.nextInt();

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

coords[i][0] = sc.nextInt();

coords[i][1] = sc.nextInt();

}

int[][] G = new int[n+1][n+1];

for (int u = 0; u < n+1; u++) {

for (int v = u+1; v < n+1; v++) {

int dx = Math.abs(coords[u][0] - coords[v][0]);

int dy = Math.abs(coords[u][1] - coords[v][1]);

G[u][v] = G[v][u] = dx + dy;

}

}

boolean[] visited = new boolean[n+1];

int ans = getAns(G, 0, n, 0, 0, visited);

System.out.printf("The shortest path has length %d\n", ans);

}

}

static int getAns(int[][] G, int cnt, int n, int u, int acc, boolean[] visited) {

if (cnt == n) {

return acc + G[u][0];

}

visited[u] = true;

int ans = Integer.MAX\_VALUE;

for (int v = 0; v < n+1; v++) {

if (!visited[v]) {

int val = getAns(G, cnt+1, n, v, acc+G[u][v], visited);

ans = Math.min(val, ans);

}

}

visited[u] = false;

return ans;

}

}

# **ENCODER AND DECODER (String)**

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in);

String codigo,cambio,transformada="";

while (sc.hasNextLine()) {

codigo=sc.nextLine();

if (codigo.isEmpty()) {

System.out.println("");

}

else if (Character.isDigit(codigo.charAt(0))==false) {

for (int i = codigo.length(); i > 0; i--) {

cambio=new StringBuilder(Integer.toString((int)codigo.charAt(i-1))).reverse()+"";

System.out.print(cambio);

}

System.out.println("");

}else{

codigo=new StringBuilder(codigo).reverse()+"";

for (int i = 0; i < codigo.length(); i=i+2) {

if (codigo.charAt(i)=='1') {

String tres=codigo.charAt(i)+""+codigo.charAt(i+1)+""+codigo.charAt(i+2);

int trestres= Integer.parseInt(tres);

transformada = transformada+(String.valueOf((char)trestres));

i=i+1;

}else{

//System.out.print("ojito "+codigo.charAt(i)+""+codigo.charAt(i+1));

String dos =codigo.charAt(i)+""+codigo.charAt(i+1);

int dosdos=Integer.parseInt(dos);

transformada= transformada+(Character.toString((char)dosdos));

}

}

System.out.println(transformada);

transformada="";

codigo="";

cambio="";

}

}

}

}

# **JOKING WITH FERMAT´S LAST THEOREM (Math)**

#include <stdio.h>

#include <algorithm>

using namespace std;

int main() {

int L, R, cases = 0;

while(scanf("%d %d", &L, &R) == 2) {

int limit = min(1005, R); // (10^3)^3 = 10^9

int ret = 0;

for(long long a = L; a <= limit; a++) {

for(long long b = L; b <= limit; b++) {

long long t = (a\*a\*a+b\*b\*b);

long long c = t/10;

if(t%10 != 3) continue;

if(L <= c && c <= R)

ret++;

}

}

printf("Case %d: %d\n", ++cases, ret);

}

return 0;

}

# **N+NODN (Math)**

#include <cstdio>

#include <algorithm>

#include <vector>

using namespace std;

int nod(int x){

int ret = 1;

for(int i = 2;i <= x / i;++i){

if(x % i == 0){

int cont = 1;

while(x % i == 0){

x /= i;

++cont;

}

ret \*= cont;

}

}

if(x > 1) ret \*= 2;

return ret;

}

int ans[1000001];

int main(){

int x = 1;

while(x <= 1000000){

++ans[x];

x += nod(x);

}

for(int i = 1;i <= 1000000;++i)

ans[i] += ans[i - 1];

int T,A,B;

scanf("%d",&T);

for(int tc = 1;tc <= T;++tc){

scanf("%d %d",&A,&B);

printf("Case %d: %d\n",tc,ans[B] - ans[A - 1]);

}

return 0;

}

# **PROFESOR (Math)**

#include <cstdio>

#include <algorithm>

using namespace std;

int main(){

int N;

scanf("%d",&N);

int A[N],B[N];

for(int i = 0;i<N;++i) scanf("%d %d",&A[i],&B[i]);

int ans[6];

for(int g = 1;g<=5;++g){

ans[g] = 0;

for(int i = 0,cont = 0;i<N;++i){

if(A[i]==g || B[i]==g) ++cont;

else cont = 0;

ans[g] = max(ans[g],cont);

}

}

int best = -1,G = -1;

for(int g = 1;g<=5;++g){

if(ans[g]>best){

best = ans[g];

G = g;

}

}

printf("%d %d\n",best,G);

return 0;

}

# **PUZ (Math)**

#include <cstdio>

using namespace std;

int main(){

int A,B,V;

scanf("%d %d %d",&A,&B,&V);

int ans = 1;

V -= A;

ans += (V+A-B-1)/(A-B);

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

return 0;

}

# **ROTATING DRUMS (Combinatory)**

#include <bits/stdc++.h>

using namespace std;

const int MAXM = 131072;

#define MAXL (MAXM>>5)+1

#define GET(x) (mark[x>>5]>>(x&31)&1)

#define SET(x) (mark[x>>5] |= 1<<(x&31))

int mark[MAXL];

char s[MAXM];

int main() {

int n, m;

while (scanf("%d %d", &n, &m) == 2) {

memset(mark, 0, sizeof(mark));

int tot = 1;

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

tot \*= n;

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

s[i] = 'A';

int st = 0, ed = tot-1;

int idx = 1, pos = 0;

memset(mark, 0, sizeof(mark));

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

ed = ed\*n - (n-1)\*tot;

SET(ed);

}

SET(st);

for (; idx < tot; ) {

st = st\*n - (s[pos]-'A')\*tot;

int fill = -1;

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

if (!GET(st+i)) {

fill = i, st += i;

break;

}

}

if (fill == -1)

break;

SET(st), s[pos+m] = fill+'A', pos++;

}

s[pos+m] = '\0';

puts(s);

}

return 0;

}

# **SHUFFLING CARDS (Math)**

import java.util.Scanner;

class Main {

public static void main(String... args) {

Scanner sc = new Scanner(System.in);

while (sc.hasNextInt()) {

int N = sc.nextInt();

long R = sc.nextLong();

int[] slots = new int[N];

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

slots[i] = sc.nextInt();

}

int[] ans = getAns(slots, N, R);

StringBuilder sb = new StringBuilder();

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

sb.append(ans[i]);

if (i < N-1) {

sb.append(" ");

}

}

System.out.println(sb);

}

}

static int[] getAns(int[] slots, int N, long R) {

int[] ans = new int[N];

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

if (slots[i]-1 == i) {

ans[i] = slots[i];

} else {

long rnd = getRnd(slots, i, N);

long mod = R % rnd;

ans[i] = getSlot(slots, i, N, mod);

}

}

return ans;

}

static long getRnd(int[] slots, int i, int N) {

int curr = slots[i]-1;

long ans = 1;

while (curr != i) {

curr = slots[curr]-1;

ans += 1;

}

return ans;

}

static int getSlot(int[] slots, int i, int N, long R) {

int curr = i;

for (long j = 0; j < R; j++) {

curr = slots[curr]-1;

}

return curr+1;

}

}

# **SCOCCER CHAMPIONSHIP (String)**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <iostream>

#include <sstream>

#include <vector>

#include <map>

#include <algorithm>

#include <limits.h>

using namespace std;

struct Record {

string l, r;

int lc, rc;

Record(string l="", string r="", int lc=0, int rc=0):

l(l), r(r), lc(lc), rc(rc) {}

};

void stat(vector<Record> &R) {

struct Entry {

int pt, diff, mx, vis\_mx;

Entry() {pt = mx = vis\_mx = 0; diff = 0;}

void update(int a, int b, int isvis) {

if (a > b) pt += 3;

else if (a == b) pt++;

diff += a - b;

mx += a;

if (isvis)

vis\_mx += a;

}

bool operator<(const Entry &x) const {

if (pt != x.pt)

return pt > x.pt;

if (diff != x.diff)

return diff > x.diff;

if (mx != x.mx)

return mx > x.mx;

return vis\_mx > x.vis\_mx;

}

};

struct StrMap {

map<string, int> str2int;

vector<string> str;

int get(string &s) {

auto it = str2int.find(s);

if (it != str2int.end())

return it->second;

int &ret = str2int[s];

ret = (int) str.size();

str.push\_back(s);

return ret;

}

int size() {

return (int) str.size();

}

} smap;

vector<Entry> table(R.size()\*2, Entry());

for (Record &e : R) {

int l = smap.get(e.l);

int r = smap.get(e.r);

table[l].update(e.lc, e.rc, 0);

table[r].update(e.rc, e.lc, 1);

}

int n = smap.size();

vector<pair<Entry, string>> ret(n);

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

ret[i] = {table[i], smap.str[i]};

sort(ret.begin(), ret.end());

vector<int> idx\_rank(n);

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

idx\_rank[smap.get(ret[i].second)] = i;

int paradox = 0;

for (Record &e : R) {

if (e.lc == e.rc)

continue;

int l = smap.get(e.l);

int r = smap.get(e.r);

if (e.lc > e.rc) {

paradox += idx\_rank[l] > idx\_rank[r];

} else {

paradox += idx\_rank[l] < idx\_rank[r];

}

}

printf("The paradox occurs %d time(s).\n", paradox);

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

printf("%d. %s\n", i+1, ret[i].second.c\_str());

// printf(" %d %d %d %d\n", ret[i].first.pt, ret[i].first.diff, ret[i].first.mx, ret[i].first.vis\_mx);

}

}

int main() {

int n;

static char s[65536];

while (scanf("%d", &n) == 1) {

while (getchar() != '\n');

vector<Record> R;

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

fgets(s, 65536, stdin);

vector<string> tks;

stringstream sin(s);

string token;

while (sin >> token)

tks.push\_back(token);

int vs\_pos = -1;

for (int j = 0; j < tks.size(); j++) {

if (tks[j] == "vs.") {

vs\_pos = j;

break;

}

}

string Ls = "", Rs = "";

int lc, rc;

sscanf(tks[vs\_pos-1].c\_str(), "%d", &lc);

sscanf(tks[vs\_pos+1].c\_str(), "%d", &rc);

for (int i = 0; i < vs\_pos-1; i++) {

Ls = Ls + tks[i];

if (i != vs\_pos-2)

Ls += " ";

}

for (int i = vs\_pos+2; i < tks.size(); i++) {

Rs = Rs + tks[i];

if (i != tks.size()-1)

Rs += " ";

}

R.push\_back(Record(Ls, Rs, lc, rc));

}

stat(R);

}

return 0;

}

# **STRETAN (Math)**

#include <cstdio>

using namespace std;

int numOfDigits(int K){

int aux = 0;

for(int i = 1;i<29;++i){

aux |= (1<<i);

if(aux>=K) return i;

}

return 29;

}

int main(){

int K;

scanf("%d",&K);

int D = numOfDigits(K);

for(int i = 1;i<D;++i) K -= (1<<i);

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

if(K<=(1<<(D-i-1))) putchar('4');

else{

putchar('7');

K -= (1<<(D-i-1));

}

}

putchar('\n');

return 0;

}

# **TABOVI (Math)**

#include <cstdio>

#include <algorithm>

using namespace std;

int main(){

int N;

scanf("%d",&N);

int A[N],B[N];

for(int i = 0;i<N;++i) scanf("%d",&A[i]);

for(int i = 0;i<N;++i) scanf("%d",&B[i]);

int sum = 0,pos = 0,neg = 0;

for(int i = N-1;i>=0;--i){

int aux = B[i]-A[i]-sum;

sum += aux;

if(aux<0) neg -= aux;

else pos += aux;

}

printf("%d\n",max(pos,neg));

return 0;

}

# **TIMSKO (Math)**

#include <cstdio>

#include <algorithm>

using namespace std;

int main(){

int M,N,K;

scanf("%d %d %d",&M,&N,&K);

printf("%d\n",min((M+N-K)/3,min(M/2,N)));

return 0;

}

# **COUNTING CHAOS (Dynamic)**

//11309

import java.util.Scanner;

public class chaos {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int cases = scanner.nextInt();

scanner.nextLine();

while (cases != 0) {

String line = scanner.nextLine();

String[] strs = line.split(":");

System.out.println(chek(strs[0], strs[1]));

cases--;

}

}

private static String chek(String h, String m) {

if (h.equals("00")) {

if (Integer.parseInt(m) < 9) {

return ("00:0" + (Integer.parseInt(m) + 1));

} else {

int aux = Integer.parseInt(m);

if (aux < 55) {

return ("00:" + (((aux / 11) % 10) + 1) \* 11);

} else {

return ("01:01");

}

}

} else {

int hour = Integer.parseInt(h);

int min = Integer.parseInt(m) + 1;

if ((hour > 16 && hour < 20)) {

return "20:02";

} else {

if (hour < 10) {

while (hour != min % 10) {

if (min < 59) {

min++;

} else {

hour++;

min = 0;

if (hour == 10) {

return "10:01";

}

}

}

if (min < 10) {

return "0" + hour + ":0" + min;

} else {

return "0" + hour + ":" + min;

}

} else {

int invh = Integer.parseInt("" + h.charAt(1) + h.charAt(0));

while (min != invh) {

if (min < 59) {

min++;

} else {

hour++;

min = 0;

if (hour < 24) {

if (hour < 10) {

invh = Integer.parseInt("" + String.valueOf(hour).charAt(0) + "0");

} else {

invh = Integer.parseInt("" + String.valueOf(hour).charAt(1) + String.valueOf(hour).charAt(0));

}

} else {

return ("00:00");

}

}

}

if (hour < 10) {

if (min < 10) {

return ("0" + hour + ":0" + min);

} else {

return ("0" + hour + ":" + min);

}

} else {

if (min < 10) {

return (hour + ":0" + min);

} else {

return (hour + ":" + min);

}

}

}

}

}

}

}

# **BUBBLE MAPS (String)**

import java.util.Scanner;

public class Main {

static char[] vecinos = new char[5000];

public static void Izquierda(int i) {

if (i == 0) {

vecinos[0] = 1;

return;

}

switch (vecinos[i]) {

case 'p':

Izquierda(i-1);

vecinos[i] = 'q';

break;

case 'q':

vecinos[i] = 'p';

break;

case 'r':

vecinos[i] = 's';

break;

case 's':

Izquierda(i-1);

vecinos[i] = 'r';

break;

}

}

public static void Derecha(int i) {

if (i == 0) {

vecinos[0] = 1;

return;

}

switch (vecinos[i]) {

case 'p':

vecinos[i] = 'q';

break;

case 'q':

Derecha(i-1);

vecinos[i] = 'p';

break;

case 'r':

Derecha(i-1);

vecinos[i] = 's';

break;

case 's':

vecinos[i] = 'r';

break;

}

}

public static void Arriba(int i) {

if (i == 0) {

vecinos[0] = 1;

return;

}

switch (vecinos[i]) {

case 'p':

Arriba(i - 1);

vecinos[i] = 's';

break;

case 'q':

Arriba(i - 1);

vecinos[i] = 'r';

break;

case 'r':

vecinos[i] = 'q';

break;

case 's':

vecinos[i] = 'p';

break;

}

}

public static void Abajo(int i) {

if (i == 0) {

vecinos[0] = 1;

return;

}

switch (vecinos[i]) {

case 'p':

vecinos[i] = 's';

break;

case 'q':

vecinos[i] = 'r';

break;

case 'r':

Abajo(i - 1);

vecinos[i] = 'q';

break;

case 's':

Abajo(i - 1);

vecinos[i] = 'p';

break;

}

}

public static void main(String[] args) {

int n, i, l, li;

Scanner in = new Scanner(System.in);

String m;

m = in.nextLine();

n = Integer.parseInt(m);

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

m = in.nextLine();

vecinos = m.toCharArray();

li = m.length() - 1;

l = li;

Arriba(l);

if (vecinos[0] == 1) {

System.out.print("<none> ");

} else {

System.out.print(vecinos);

System.out.print(" ");

}

l = li;

vecinos = m.toCharArray();

Abajo(l);

if (vecinos[0] == 1) {

System.out.print("<none> ");

} else {

System.out.print(vecinos);

System.out.print(" ");

}

l = li;

vecinos = m.toCharArray();

Izquierda(l);

if (vecinos[0] == 1) {

System.out.print("<none> ");

} else {

System.out.print(vecinos);

System.out.print(" ");

}

vecinos = m.toCharArray();

l = li;

l = li;

Derecha(l);

if (vecinos[0] == 1) {

System.out.println("<none>");

} else {

System.out.println(vecinos);

}

}

}

}

# **LAZY JUMPING FROG (Graph)**

import java.util.Scanner;

public class Main2 {

public static final int MAX\_N = 10000;

public static final int MAX\_K = 100;

public static void main(String[] args) {

int[] count;

int n = 1, c, k;

int i, j, x, min;

boolean first;

Scanner in = new Scanner(System.in);

while ((n = in.nextInt()) > 0) {

c = in.nextInt();

k = in.nextInt();

count = new int[MAX\_K + 1];

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

for (j = 0; j < c; j++) {

x = in.nextInt();

count[x]++;

}

}

min = n;

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

if (count[i] < min) {

min = count[i];

}

}

first = true;

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

if (count[i] == min) {

if (!first) {

System.out.print(" ");

}

first = false;

System.out.print(i);

}

}

System.out.print("\n");

}

}

}

# **TORNADO! (Dynamic)**

import java.util.Scanner;

public class Main3 {

//ACM live archive 3468 Tornado!

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n;

n = sc.nextInt();

while (n!=0) {

boolean[] post = new boolean[n];

int firstPost = -1;

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

int aux = sc.nextInt();

if (aux == 1) {

post[i] = true;

} else {

post[i] = false;

}

if (post[i] && firstPost == -1) {

firstPost = i;

}

}

if (firstPost == -1) { //Todos son ceros

System.out.printf("%d\n", n / 2 + (n & 1));

} else {

int count = 0;

int i = firstPost;

do {

if (!post[i] && !post[(i + 1) % n]) {

post[(i + 1) % n] = true;

count++;

}

i = (i + 1) % n;

} while (i != firstPost);

System.out.printf("%d\n", count);

}

n = sc.nextInt();

}

}

}

# **IS THIS INTEGRATION? (Geometry)**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class Main4 {

//uva 10209 Is This Integration?

public static void main(String[] args) throws IOException {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

StringBuffer sb=new StringBuffer("");

String m="";

while((m=br.readLine())!=null){

double a=Double.parseDouble(m);

double areaOfRect=a\*a;

double x = areaOfRect \* (1 - Math.sqrt(3) + Math.PI / 3);

double y = areaOfRect \* (2 \* Math.sqrt(3) - 4 + Math.PI / 3);

double z = areaOfRect \* (4 - Math.sqrt(3) - 2 \* Math.PI / 3);

String temp=String.format("%.3f %.3f %.3f\n", x,y,z);

sb.append(temp);

}

System.out.print(sb);

}

}

# **THE TWO NOTE RAG (Brute-force)**

import java.util.Scanner;

public class Main5 {

//ACM live archive 4239 The Two Note Rag

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int i;

String[] res = {"", "1", "9", "89", "89", "589", "3089", "3089", "3089", "315589", "315589", "8128089", "164378089", "945628089", "1922190589", "11687815589", "109344065589", "231414378089", "1452117503089", "4503875315589", "65539031565589", "141832976878089", "1667711883128089"};

int cases = sc.nextInt();

for ( i = 1; i <= cases; i++) {

int busqueda=sc.nextInt();

System.out.println(i+" "+busqueda+" "+res[busqueda]);

}

}

}

# **RECURSIVELY PALINDROMIC PARTITIONS (Math)**

import java.util.Scanner;

//ACM live archive 4235 - Recursively Palindromic Partitions

public class Main6 {

static int T, n;

static int[] res = new int[1002];

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

T = sc.nextInt();

res[0] = res[1] = 1;

for (int i = 2; i < 1001; i++) {

res[i] = 0;

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

res[i] += res[j];

}

}

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

n = sc.nextInt();

System.out.println(k + " " + res[n]);

}

}

}

# **ALMOST SHORTEST PATH (Graph)**

#include <stdio.h>

#include <string.h>

#include <queue>

#include <vector>

using namespace std;

struct edge {

int from, to, v, SP;//shortest path

edge(int a, int b, int c, int d):

to(a), v(b), SP(c), from(d) {}

};

vector<edge> g[505];

vector<edge\*> pre[505];

int dis[505];

void spfa(int st, int ed, int save) {

int inq[505] = {}, tn;

memset(dis, 63, sizeof(dis));

queue<int> Q;

dis[st] = 0;

Q.push(st), inq[st] = 1;

while(!Q.empty()) {

tn = Q.front(), Q.pop();

inq[tn] = 0;

for(vector<edge>::iterator it = g[tn].begin();

it != g[tn].end(); it++) {

if(it->SP) continue;

if(dis[it->to] > dis[tn] + it->v) {

dis[it->to] = dis[tn] + it->v;

pre[it->to].clear();

if(inq[it->to] == 0)

Q.push(it->to), inq[it->to] = 1;

}

if(save && dis[it->to] == dis[tn] + it->v)

pre[it->to].push\_back(&(\*it));

}

}

}

void cutSP(int st, int ed) {

int inq[505] = {}, tn;

queue<int> Q;

inq[ed] = 1;

Q.push(ed);

while(!Q.empty()) {

tn = Q.front(), Q.pop();

for(vector<edge\*>::iterator it = pre[tn].begin();

it != pre[tn].end(); it++) {

(\*it)->SP = 1;

if(inq[(\*it)->from] == 0) {

Q.push((\*it)->from);

inq[(\*it)->from] = 1;

}

}

}

}

int main() {

int n, m, x, y, v;

int st, ed;

int i, j, k;

while(scanf("%d %d", &n, &m) == 2 && n) {

scanf("%d %d", &st, &ed);

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

g[i].clear();

pre[i].clear();

}

while(m--) {

scanf("%d %d %d", &x, &y, &v);

g[x].push\_back(edge(y,v,0,x));

}

spfa(st, ed, 1);

cutSP(st, ed);

spfa(st, ed, 0);

if(dis[ed] == 0x3f3f3f3f)

puts("-1");

else

printf("%d\n", dis[ed]);

}

return 0;

}

# **ALL START THREE POINT CONTEST (String)**

#include <bits/stdc++.h>

using namespace std;

string toLower(string a) {

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

a[i] = tolower(a[i]);

return a;

}

bool cmpStr(string a, string b) {

return toLower(a) < toLower(b);

}

bool cmp(pair<int, string> &a, pair<int, string> &b) {

if (a.first != b.first)

return a.first > b.first;

return cmpStr(a.second, b.second);

}

int main() {

int n, cases = 0;

char s[1024];

while (scanf("%d", &n) == 1) {

while (getchar() != '\n');

vector<pair<int, string> > A;

A.reserve(n);

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

fgets(s, 1024, stdin);

char \*tok = strtok(s, ";");

string name = tok; tok = strtok(NULL, ";");

int s = 0;

while (tok != NULL) {

int a, b, c, d, e;

sscanf(tok, "%d %d %d %d %d", &a, &b, &c, &d, &e);

s += (a+b+c+d)+e\*2;

tok = strtok(NULL, ";");

}

A.emplace\_back(make\_pair(s, name));

}

sort(A.begin(), A.end(), cmp);

printf("Case %d:\n", ++cases);

for (auto e: A)

printf("%s %d\n", e.second.c\_str(), e.first);

}

return 0;

}

/\*

3

Michael Jordan;0 1 1 0 1;0 1 1 0 1;0 1 1 0 1;0 0 0 0 1;0 0 0 0 1

Scotty Pippen;0 1 1 0 1;0 1 1 0 1;0 1 1 0 1;0 0 0 0 1;0 0 0 1 1

Charles Barkley;0 1 1 0 1;0 1 1 0 1;0 1 1 0 1;0 0 0 0 1;0 0 1 1 1

\*/

# **DIVISORS (Math)**

import java.util.Scanner;

public class Main {

//NMA-ARTEMIS 2019 Divisors 294

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n, l, u, maximodiv = 0, maximonum = 0, aux = 0;

n = sc.nextInt();

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

l = sc.nextInt();

u = sc.nextInt();

for (int j = l; j <= u; j++) {

aux = divisors(j);

if (aux > maximodiv) {

maximodiv = aux;

maximonum = j;

}

}

System.out.println("Between "+l+" and "+u+", "+maximonum+" has a maximum of "+maximodiv+" divisors.");

aux=0;

maximodiv=0;

maximonum=0;

}

}

public static int divisors(int x) {

int nDiv = 1;

for (int p = 2; p \* p <= x; ++p) {

int cnt = 0;

while (x % p == 0) {

++cnt;

x /= p;

}

nDiv \*= cnt + 1;

}

if (x > 1) {

nDiv \*= 2;

}

return nDiv;

}

}