제 18회
한국정보올림피아드

문 제

종이 자르기

아래 <그림 1>과 같이 직사각형 모양의 종이가 있다. 이 종이는 가로 방향과 세로 방향으로 1 cm마다 점선이 그어져 있다. 가로 점선은 위에서 아래로 1번부터 차례로 번호가 붙어 있고, 세로 점선은 왼쪽에서 오른쪽으로 번호가 붙어 있다.

<그림 1>

점선을 따라 이 종이를 칼로 자르려고 한다. 가로 점선을 따라 자르는 경우는 종이의 왼쪽 끝에서 오른쪽 끝까지, 세로 점선인 경우는 위쪽 끝에서 아래쪽 끝까지 한번에 자른다. 예를 들어, <그림 1>의 가로 길이 10 cm이고 세로 길이 8 cm인 종이를 3번 가로 점선, 4번 세로 점선, 그리고 2번 가로 점선을 따라 자르면 <그림 2>와 같이 여러 개의 종이 조각으로 나뉘게 된다. 이 때 가장 큰 종이 조각의 넓이는 30 cm^2이다.

입력으로 종이의 가로와 세로 길이, 그리고 잘라야할 점선들이 주어질 때, 가장 큰 종이 조각의 넓이가 몇 cm^2인지를 구하는 프로그램을 작성하시오.

실행파일의 이름은 AA.EXE로 하고, 프로그램의 실행시간은 5초를 넘을 수 없다. 부분점수는 없다.

입력 형식

입력 파일명은 INPUT.TXT로 한다. 입력 파일의 첫째 줄에는 종이의 가로와 세로의 길이가 차례로 자연수로 주어진다. 가로와 세로의 길이는 최대 100 cm이다. 둘째 줄에는 칼로 잘라야하는 점선의 개수가 주어진다. 입력 파일의 셋째 줄부터 마지막 줄까지 한 줄에 점선이 하나씩 아래와 같은 방법으로 입력된다. 가로로 자르는 점선은 0과 점선 번호가 차례로 주어지고, 세로로 자르는 점선은 1과 점선 번호가 주어진다. 입력되는 두 숫자 사이에는 빈칸이 하나씩 있다.

<그림 2>

출력 형식

출력 파일명은 OUTPUT.TXT로 한다. 첫째 줄에 가장 큰 종이 조각의 넓이를 출력한다. 단, 넓이의 단위는 출력하지 않는다.

입력과 출력의 예

입력(INPUT.TXT)

```
10 8
3
0 3
1 4
0 2
```

출력(OUTPUT.TXT)

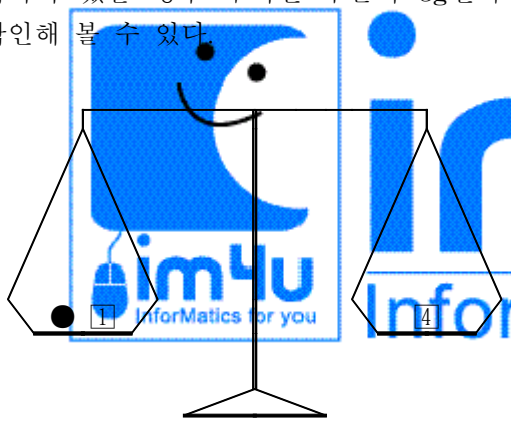
```
30
```

양팔 저울

양팔 저울과 몇 개의 추가 주어졌을 때, 이를 이용하여 입력으로 주어진 구슬의 무게를 확인할 수 있는지를 결정하려고 한다.

무게가 각각 1g과 4g인 두 개의 추가 있을 경우, 주어진 구슬과 1g 추 하나를 양팔 저울의 양쪽에 각각 올려놓아 수평을 이루면 구슬의 무게는 1g이다. 또 다른 구슬이 4g인지를 확인하려면 1g 추 대신 4g 추를 올려놓으면 된다.

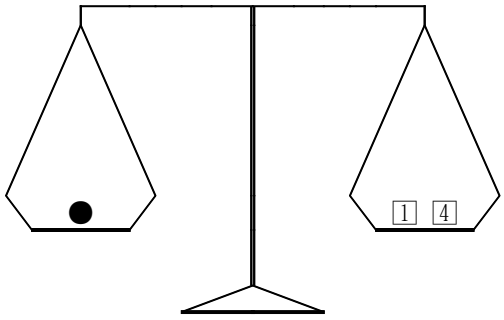
구슬이 3g인 경우 아래 <그림 1>과 같이 구슬과 추를 올려놓으면 양팔 저울이 수평을 이루게 된다. 따라서 각각 1g과 4g인 추가 하나씩 있을 경우 주어진 구슬이 3g인지도 확인해 볼 수 있다.



<그림 3> 구슬이 3g인지 확인하는 방법
(1은 1g인 추, 4는 4g인 추, ●은 무게를 확인할 구슬)

<그림 2>와 같은 방법을 사용하면 구슬이 5g인지도 확인할 수 있다. 구슬이 2g이면 주어진 추를 가지고는 확인할 수 없다.

추들의 무게와 확인할 구슬들의 무게가 입력되었을 때, 주어진 추만을 사용하여 구슬의 무게를 확인할 수 있는지를 결정하는 프로그램을 작성하시오.



<그림 4> 구슬이 5g인지 확인하는 방법

실행파일의 이름은 BB.EXE로 하고, 프로그램의 실행시간은 5초를 넘을 수 없다. 부분점수는 없다.

입력 형식

입력 파일명은 INPUT.TXT로 한다. 입력 파일의 첫째 줄에는 추의 개수가 자연수로 주어진다. 추의 개수는 30 이하이다. 둘째 줄에는 추의 무게들이 자연수로 가벼운 것부터 차례로 주어진다. 같은 무게의 추가 여러 개 있을 수도 있다. 추의 무게는 500g이하이며, 입력되는 무게들 사이에는 빈칸이 하나씩 있다. 세 번째 줄에는 무게를 확인하고자 하는 구슬들의 개수가 주어진다. 확인할 구슬의 개수는 7이하이다. 네 번째 줄에는 확인하고자 하는 구슬들의 무게가 자연수로 주어지며, 입력되는 무게들 사이에는 빈칸이 하나씩 있다.

출력 형식

출력 파일명은 OUTPUT.TXT로 한다. 주어진 각 구슬의 무게에 대하여 확인이 가능하면 Y, 아니면 N을 차례로 출력한다. 출력 파일은 한 개의 줄로 이루어지며, 각 구슬에 대한 답 사이에는 빈칸을 하나씩 둔다.

입력과 출력의 예(1)

입력(INPUT.TXT)

```
2
1 4
2
3 2
```

출력(OUTPUT.TXT)

```
Y N
```

입력과 출력의 예(2)

입력(INPUT.TXT)

```
4
2 3 3 3
3
1 4 10
```

출력(OUTPUT.TXT)

```
Y Y N
```


모범답안

1. AA

```
{$A+,B-,D+,E+,F-,G-,I+,L+,N-,O-,P-,Q+,R+,S+,T-,V+,X+,Y+}
{$M 16384,0,655360}

Program Paper;

const
      inpf = 'input.txt';
      outf = 'output.txt';
      maximum = 1000;

var
   w,h,m : integer;
   c : array [0..1,0..maximum] of integer;

procedure input_data;

var
   f : text;
   m,i,a,b: integer;

begin
      assign(f,inpf);
      reset(f);
      readln(f,w,h);
      readln(f,m);

      c[1,0] := 2 ; c[1,1] := 0 ; c[1,2] := w ;
      c[0,0] := 2 ; c[0,1] := 0 ; c[0,2] := h ;
      for i:= 1 to m do
      begin
            readln(f,a,b);
            inc(c[a,0]);
            c[a,c[a,0]] := b ;
      end;
      close(f)
end;

procedure swap(var a,b : integer);
var
      c           : integer;

begin
      c := a ; a := b ; b := c ;
end;

procedure Quick_sort(a : integer ; l,h : integer);
var
      last, i : integer;

begin
      if (l<h) then
      begin
            last := l ;
            for i := l+1 to h do
            begin
                  if c[a,l]>c[a,i] then
                  begin
                        inc(last);
                        swap(c[a,last],c[a,i]);
                  end;
            end;

            swap(c[a,last],c[a,l]);
            Quick_sort(a,l,last-1);
            Quick_sort(a,last+1,h);
      end;
end;

function find_longest_period(a : integer) : longint;
var
      i,max       : integer;
```

```
begin
        max := 0 ;
        for i := 1 to c[a,0]-1 do
            if (c[a,i+1]-c[a,i])>max then
                max := c[a,i+1]-c[a,i];
        find_longest_period := max;
end;

function process : longint;
begin
        Quick_sort(1,1,c[1,0]);
        Quick_sort(0,1,c[0,0]);
        process := find_longest_period(1)*find_longest_period(0);
end;

procedure write_result(s : longint);

var
   f : text;

begin
    assign(f,outf);
    rewrite(f);
    writeln(f,s);
    close(f)
end;

begin
    input_data;
    write_result(process);
end.
```

2. BB

```
program test1;

const
        filename1 = 'input.txt';
        filename2 = 'output.txt';
        maximum   = 100 ;

var
        num,solution      : array[0..maximum] of longint;
        subsum            : array[1..maximum] of longint;
        gcd               : array[1..maximum] of longint;
        n,s               : longint;
        count             : longint;
        ox                : longint;

procedure sol(level, sum : longint);
var
        i       : longint;

begin
        if (sum=s) then
        begin
            ox := 1 ;
            inc(count);
        end;

        if ( (ox=0) and
             ((level<=n) and
            (s>=sum-subsum[level]) and
            (s<=sum+subsum[level]) ) and
            (((s-sum) mod gcd[level])=0) ) then
        begin
            if( (not (num[level]=num[level-1])) or
                (solution[level-1]=1) ) then
            begin
                solution[level] := 1 ; sol(level+1,sum+num[level]);
            end;
            solution[level] := 2 ; sol(level+1,sum);
            if( (not (num[level]=num[level-1])) or
                (solution[level-1]=3) ) then
            begin
                solution[level] := 3 ; sol(level+1,sum-num[level]);
            end;
        end;
end;

procedure swaping(var a,b : longint);
var
        temp    : longint;

begin
        temp := a ; a := b ; b := temp ;
end;

function gcd_(a,b : longint) : longint;
var
        r       : longint;
begin
        while(true) do
        begin
            if(a<b) then swaping(a,b);
            r := a mod b ;
            if (r=0) then
            begin
                gcd_ := b ;
                break;
            end;
            a := b ; b := r ;
        end;
end;

procedure base;
var
        i       : longint;

begin
        subsum[n+1] := 0 ;
        for i := n downto 1 do
        begin
            subsum[i] := subsum[i+1]+num[i];
        end;
        gcd[n+3] := 1 ;
```

```
        gcd[n+2] := 1 ;
        gcd[n+1] := 1 ; gcd[n] := num[n] ;
        for i := n-1 downto 1 do
        begin
            gcd[i] := gcd_(gcd[i+1],num[i]);
        end;
end;

procedure process2;
begin
        count := 0 ;

        sol(1,0);
end;
procedure input_data;
var
        infile,outfile  : text;
        i,m             : longint;

begin
        assign(infile,filename1);
        reset(infile);
        readln(infile,n);
        for i := n downto 1 do
        begin
            read(infile,num[i]);
        end;
        num[n+1] := -1;
        num[0] := -1;
        readln(infile);

        base;
        assign(outfile,filename2);
        rewrite(outfile);


        readln(infile,m);
        for i := 1 to m do
        begin
            read(infile,s);
            ox := 0 ;
            process2;
            if(ox=0) then write(outfile,'N ')
                     else write(outfile,'Y ');
        end;
        writeln(outfile);

        close(outfile);
        readln(infile);

        close(infile);
end;


begin
        input_data;
end.
```

im4u
InforMatics for you

색종이 만들기

아래 <그림 1>과 같이 여러 개의 정사각형칸들로 이루어진 정사각형 모양의 종이가 주어져 있고, 각 정사각형칸들은 하얀색으로 칠해져 있거나 파란색으로 칠해져 있다. 주어진 종이를 일정한 규칙에 따라 잘라서 다양한 크기를 가진 정사각형 모양의 하얀색 또한 파란색 색종이를 만들려고 한다.

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

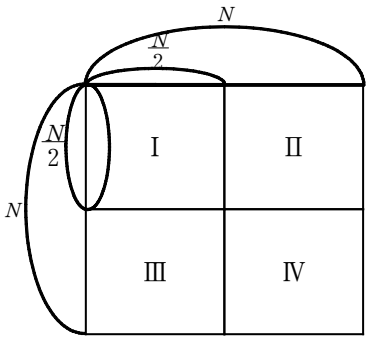
□ 하얀색
■ 파란색

<그림 1> 8×8 종이

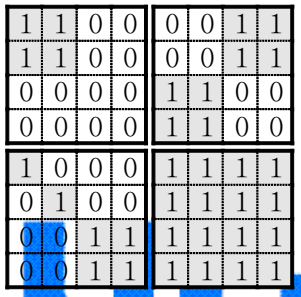
전체 종이의 크기가 $N \times N$ ($N = 2^k$, k는 1 이상 7 이하의 자연수) 이라면 종이를 자르는 규칙은 다음과 같다.

전체 종이가 모두 같은 색으로 칠해져 있지 않으면 가로와 세로로 중간 부분을 잘라서 <그림 2>의 Ⅰ, Ⅱ, Ⅲ, Ⅳ와 같이 똑같은 크기의 네 개의 $\frac{N}{2} \times \frac{N}{2}$ 색종이로 나눈다. 나누어진 종이 Ⅰ, Ⅱ, Ⅲ, Ⅳ 각각에 대해서도 앞에서와 마찬가지로 모두 같은 색으로 칠해져 있지 않으면 같은 방법으로 똑같은 크기의 네 개의 색종이로 나눈다. 이와 같은 과정을 잘라진 종이가 모두 하얀색 또는 모두 파란색으로 칠해져 있거나, 하나의 정사각형 칸이 되어 더 이상 자를 수 없을 때까지 반복한다.

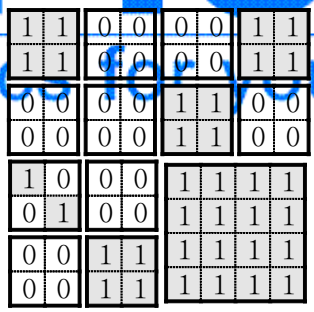
위와 같은 규칙에 따라 잘랐을 때 <그림 3>은 <그림 1>의 종이를 처음 나눈 후의 상태를, <그림 4>는 두 번째 나눈 후의 상태를, <그림 5>는 최종적으로 만들어진 다양한 크기의 9장의 하얀색 색종이와 7장의 파란색 색종이를 보여주고 있다.



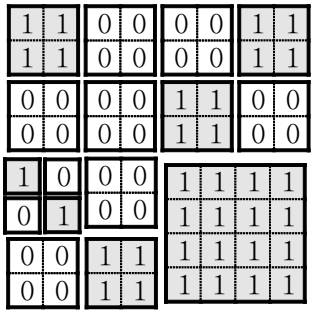
<그림 2>



<그림 3> 처음 나눈 후의 상태



<그림 4> 두 번째 나눈 후의 상태



<그림 5> 최종적으로 나누어진 색종이들

입력으로 주어진 종이의 한 변의 길이 N과 각 정사각형칸의 색 (하얀색 또는 파란색)이 주어질 때 잘라진 하얀색 색종이와 파란색 색종이의 개수를 구하는 프로그램을 작성하시오.

실행 파일의 이름은 COLOR.EXE로 하고, 프로그램의 실행시간은 5초를 초과할 수 없다. 부분점수는 없다.

입력형식

입력 파일명은 INPUT.TXT로 한다. 입력 파일의 첫째 줄에는 전체 종이의 한 변의 길이 N이 주어져 있다. N은 2, 4, 8, 16, 32, 64, 128 중 하나이다. 색종이의 각 가로줄의 정사각형칸들의 색이 윗줄부터 차례로 입력 파일의 둘째 줄부터 마지막 줄까지 주어진다. 하얀색으로 칠해진 칸은 0, 파란색으로 칠해진 칸은 1로 주어지며 각 숫자 사이에는 빈칸이 하나씩 있다.

출력형식

출력 파일명은 OUTPUT.TXT로 한다. 첫째 줄에는 잘라진 하얀색 색종이의 개수를 출력하고 둘째 줄에는 파란색 색종이의 개수를 출력한다.

입력과 출력의 예

입력(INPUT.TXT)

```
8
1 1 0 0 0 0 1 1
1 1 0 0 0 0 1 1
0 0 0 0 1 1 0 0
0 0 0 0 1 1 0 0
1 0 0 0 1 1 1 1
0 1 0 0 1 1 1 1
0 0 1 1 1 1 1 1
0 0 1 1 1 1 1 1
```

출력(OUTPUT.TXT)

```
9
7
```

줄 세우기

KOI 어린이집에는 N명의 아이들이 있다. 오늘은 소풍을 가는 날이다. 선생님은 1번부터 N번까지 번호가 적혀있는 번호표를 아이들의 가슴에 붙여주었다. 선생님은 아이들을 효과적으로 보호하기 위해 목적지까지 번호순서대로 일렬로 서서 걸어가도록 하였다. 이동 도중에 보니 아이들의 번호순서가 바뀌었다. 그래서 선생님은 다시 번호 순서대로 줄을 세우기 위해서 아이들의 위치를 옮기려고 한다. 그리고 아이들이 혼란스러워하지 않도록 하기 위해 위치를 옮기는 아이들의 수를 최소로 하려고 한다.

예를 들어, 7명의 아이들이 다음과 같은 순서대로 줄을 서 있다고 하자.

3 7 5 2 6 1 4

아이들을 순서대로 줄을 세우기 위해, 먼저 4번 아이를 7번 아이의 뒤로 옮겨보자. 그러면 다음과 같은 순서가 된다.

3 7 4 5 2 6 1

이제, 7번 아이를 맨 뒤로 옮긴다.

3 4 5 2 6 1 7

다음, 1번 아이를 맨 앞으로 옮긴다.

1 3 4 5 2 6 7

마지막으로 2번 아이를 1번 아이의 뒤로 옮기면 번호 순서대로 배치된다.

1 2 3 4 5 6 7

위의 방법으로는 모두 4명의 아이를 옮겨 번호 순서대로 줄을 세운다. 위의 예에서 3명의 아이만을 옮겨서는 순서대로 배치할 수가 없다. 따라서, 4명을 옮기는 것이 가장 적은 수의 아이를 옮기는 것이다.

N명의 아이들이 임의의 순서로 줄을 서 있을 때, 번호 순서대로 배치하기 위해 옮겨지는 아이의 최소 수를 구하는 프로그램을 작성하시오.

실행파일의 이름은 LINE.EXE로 하고, 프로그램의 실행시간은 5초를 초과할 수 없다. 부분 점수는 없다.

입력 형식

입력 파일명은 INPUT.TXT로 한다. 입력 파일의 첫째 줄에는 아이들의 수 N이 주어진다. 둘째 줄부터는 1부터 N까지의 숫자가 한 줄에 하나씩 주어진다. N은 2이상 200이하의 정수이다.

출력 형식

출력 파일명은 OUTPUT.TXT로 한다. 출력 파일의 첫째 줄에는 번호 순서대로 줄을 세우는데 옮겨지는 아이들의 최소 수를 출력한다.

입력과 출력의 예

입력(INPUT.TXT)

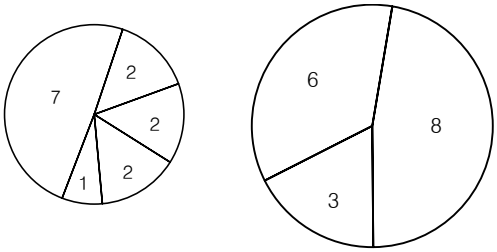
```
7
3
7
5
2
6
1
4
```

출력(OUTPUT.TXT)

```
4
```

피자 판매

고객이 두 종류의 피자 A와 B를 취급하는 피자 가게에서 피자를 주문하고자 한다. <그림 1>과 같이 각 종류의 피자는 다양한 크기의 여러 개의 피자조각으로 나누어져 있다. 각 조각에 쓰여진 숫자는 피자조각의 크기를 나타낸다.



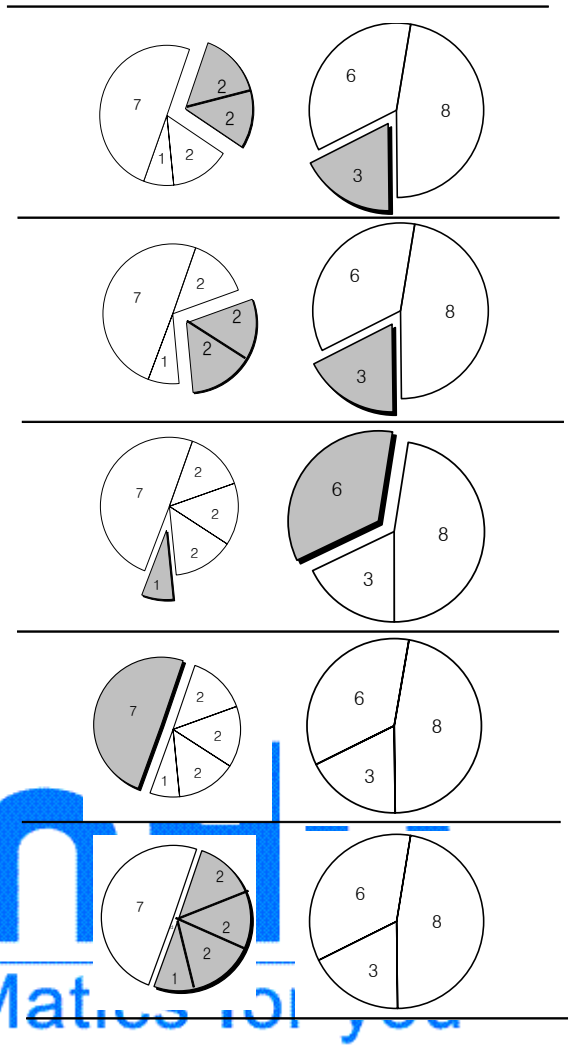
피자 A 피자 B

<그림 1>

고객이 원하는 피자의 크기를 이야기하면, 피자 가게에서는 한 종류의 피자를 2 조각 이상 판매할 때는 반드시 연속된 조각들을 잘라서 판매한다. 이때 판매한 피자조각의 크기 합이 주문한 크기가 되어야 한다. 판매한 피자조각은 모두 A종류이거나, 모두 B종류이거나, 또는 A와 B 종류가 혼합될 수 있다. 예를 들어서, <그림 1>과 같이 잘라진 피자가 있을 때, 손님이 전체 크기가 7 인 피자를 주문하면, 피자 가게에서는 <그림 2>와 같이 5 가지 방법으로 피자를 판매할 수 있다.

피자가게에서 손님이 원하는 크기의 피자를 판매하는 모든 방법의 가지 수를 계산하는 프로그램을 작성하시오.

실행파일의 이름은 PIZZA.EXE로 하고, 프로그램의 실행시간은 10초를 초과할 수 없다. 부분점수는 없다.



<그림 2>

입력 형식

입력 파일명은 INPUT.TXT로 한다. 입력 파일의 첫 번째 줄에는 손님이 구매하고자 하는 피자크기를 나타내는 2,000,000 이하의 자연수가 주어진다. 두 번째 줄에는 A, B 피자의 피자조각의 개수를 나타내는 정수 m, n 이 차례로 주어진다 ($3 \le m, n \le 1000$). 세 번째 줄부터 차례로 m 개의 줄에는 피자 A의 미리 잘라진 피자조각의 크기를 나타내는 정수가 주어진다. 그 다음 n 개의 줄에는 차례로 피자 B의 미리 잘라진 피자조각의 크기를 나타내는 정수가 주어진다. 각 종류의 피자조각의 크기는 시계방향으로 차례로 주어지며, 각 피자 조각의 크기는 1000 이하의 자연수이

다.

출력 형식

출력 파일명은 OUTPUT.TXT로 한다. 출력 파일의 첫째 줄에는 피자를 판매하는 방법의 가지 수를 나타내는 정수를 출력한다. 피자를 판매하는 방법이 없는 경우에는 숫자 0을 출력한다.

입력과 출력의 예

입력(INPUT.TXT)

```
7
5 3
2
2
1
7
2
6
8
3
```

출력(OUTPUT.TXT)

```
5
```

모범답안

1. COLOR

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

#define MAX 128

int a[MAX][MAX];

int sumblue=0,sumwhite=0;

void ansfinder(int x, int y, int step)
{
  int blue=0,white=0,i,j;
  for (i=x;i<x+step;i++)
    for (j=y;j<y+step;j++)
      if (a[i][j]==1) blue++; else white++;
  if (blue==0) { sumwhite++; return; }
  if (white==0) {sumblue++; return; }
  ansfinder(x, y, step/2);
  ansfinder(x, y+step/2, step/2);
  ansfinder(x+step/2, y, step/2);
  ansfinder(x+step/2, y+step/2, step/2);
}

void main()
{
  int n,i,j;

  FILE *in = fopen("input.txt","r");
  fscanf(in,"%d",&n);
  for (i=0;i<n;i++)
    for (j=0;j<n;j++)
      fscanf(in,"%d",&a[i][j]);
  fclose(in);

  ansfinder(0,0,n);

  FILE *out = fopen("output.txt","w");
  fprintf(out,"%d\n",sumwhite);
  fprintf(out,"%d\n",sumblue);
  fclose(out);
}
```

2. LINE

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

const int MAX = 210; // spare -_-

int n;
int lineup[MAX];

void get_input()
{
  int i;
  FILE *in = fopen("input.txt","r");
  fscanf(in,"%d",&n);
  for (i=0;i<n;i++)
    fscanf(in,"%d",&lineup[i]);
  fclose(in);
}

int find_MIS()
{
  int i,j,max=0;
  int partmax[MAX];
  for (i=0;i<n;i++) {
```

```
      partmax[i] = 1;
      for (j=0;j<i;j++)
        if (lineup[j]<lineup[i])
            if (partmax[j]+1>partmax[i])
              partmax[i] = partmax[j]+1;
      if (partmax[i]>max)
        max = partmax[i];
    }
    return max;
}

void print_out(int ans)
{
    FILE *out = fopen("output.txt","w");
    fprintf(out,"%d\n",ans);
    fclose(out);
}

void main()
{
    get_input();
    int incr = find_MIS();
    print_out(n-incr);
}
```

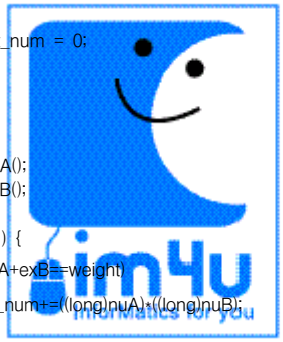
3. PIZZA

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

#define INF 99999999

// The O(N^2 log N) algorithm.

const int MAX = 1000;
int n,m;
long weight;
long pa[MAX],pb[MAX];

void get_input();
long go_go_main();
void print_out(long sum);

void main()
{
    long sum=0;

    get_input();

    sum = go_go_main();

    print_out(sum);
}

void get_input()
{
    int i;
    FILE *in = fopen("input.txt","r");
    fscanf(in,"%ld",&weight);
    fscanf(in,"%d %d",&n,&m);
    for (i=0;i<n;i++)
      fscanf(in,"%ld",&pa[i]);
    for (i=0;i<m;i++)
      fscanf(in,"%ld",&pb[i]);
    fclose(in);
}

long sumA[MAX]; int placeA[MAX],heapA[MAX];
long sumB[MAX]; int placeB[MAX],heapB[MAX];

void init_A()
{
    int i,j,temp;
    sumA[0] = 0;
    for (i=1;i<n;i++)
      sumA[i] = pa[i];
    placeA[0] = 0;
    for (i=1;i<n;i++)
      placeA[i] = 1;
    for (i=0;i<n;i++)
      heapA[i] = i;
    for (i=0;i<n;i++)
      for (j=i+1;j<n;j++)
        if (sumA[heapA[i]]>sumA[heapA[j]]) {
            temp = heapA[i];
            heapA[i] = heapA[j];
            heapA[j] = temp;
        }
}

void init_B()
{
    int i,j,temp;
    sumB[0] = 0;
    for (i=0;i<m;i++)
      sumB[0] += pb[i];
    for (i=1;i<m;i++)
      sumB[i] = sumB[0] - pb[i];
    placeB[0] = m;
    for (i=1;i<m;i++)
      placeB[i] = m-1;
    for (i=0;i<m;i++)
      heapB[i] = i;
    for (i=0;i<m;i++)
      for (j=i+1;j<m;j++)
        if (sumB[heapB[i]]<sumB[heapB[j]]) {
            temp = heapB[i];
            heapB[i] = heapB[j];
            heapB[j] = temp;
        }
}

long out_and_in_A()
{
    long retnum;
```

```
  int hp = heapA[0];
  retnum = sumA[hp];
  if (((placeA[hp]==n-1) && (hp!=0))
      || ((placeA[hp]==n) && (hp==0)))
    sumA[hp] = +INF;
  else
    sumA[hp] += pa[(hp+placeA[hp])%n];
  placeA[hp]++;
  heapA[0] = heapA[n-1];

  int pm;
  int temp,chal;

  pm = 0;
  while (1) {
    // no challenger
    if (pm*2+1>=n-1) break;

    if (pm*2+1==n-2)
      chal = pm*2+1; // 1 challenger
    else if (sumA[heapA[pm*2+1]]>sumA[heapA[pm*2+2]])
      chal = pm*2+2; // 2 challenger and the latter defeats
    else chal = pm*2+1; // 2 challenger and the former defeats

    if (sumA[heapA[pm]]>sumA[heapA[chal]]) {
      temp = heapA[pm];
      heapA[pm] = heapA[chal];
      heapA[chal] = temp;
    }
    else
      break;

    pm = chal;
  }

  heapA[n-1] = hp;
  pm = n-1;
  while (1) {
    // no challenger
    if (pm==0) break;

    // 1 challenger
    chal = (pm-1)/2;

    if (sumA[heapA[pm]]<sumA[heapA[chal]]) {
      temp = heapA[pm];
      heapA[pm] = heapA[chal];
      heapA[chal] = temp;
    }
    else
      break;

    pm = chal;
  }
  return retnum;
}

long out_and_in_B()
{
  long retnum;
```

```
  int hp = heapB[0];
  retnum = sumB[hp];
  if (((placeB[hp]==1) && (hp!=0))
      || ((placeB[hp]==0) && (hp==0)))
    sumB[hp] = -INF;
  else
    sumB[hp] -= pb[(hp+placeB[hp])%m];
  placeB[hp]--;
  heapB[0] = heapB[m-1];

  int pm;
  int temp,chal;

  pm = 0;
  while (1) {
    // no challenger
    if (pm*2+1>=m-1) break;

    if (pm*2+1==m-2)
      chal = pm*2+1; // 1 challenger
    else if (sumB[heapB[pm*2+1]]<sumB[heapB[pm*2+2]])
      chal = pm*2+2; // 2 challenger and the latter defeats
    else chal = pm*2+1; // 2 challenger and the former defeats

    if (sumB[heapB[pm]]<sumB[heapB[chal]]) {
      temp = heapB[pm];
      heapB[pm] = heapB[chal];
      heapB[chal] = temp;
    }
    else
      break;

    pm = chal;
  }

  heapB[m-1] = hp;
  pm = m-1;
  while (1) {
    // no challenger
    if (pm==0) break;

    // 1 challenger
    chal = (pm-1)/2;

    if (sumB[heapB[pm]]>sumB[heapB[chal]]) {
      temp = heapB[pm];
      heapB[pm] = heapB[chal];
      heapB[chal] = temp;
    }
    else
      break;

    pm = chal;
  }
  return retnum;
}

long exA, exB;
int nuA, nuB;
```

```
void extract_A()
{
  nuA = 0;
  exA = sumA[heapA[0]];
  if (exA==+INF) return;
  while (sumA[heapA[0]]==exA) {
    out_and_in_A();
    nuA++;
  }
}

void extract_B()
{
  nuB = 0;
  exB = sumB[heapB[0]];
  if (exB==-INF) return;
  while (sumB[heapB[0]]==exB) {
    out_and_in_B();
    nuB++;
  }
}

long go_go_main()
{
  long ret_num = 0;

  init_A();
  init_B();

  extract_A();
  extract_B();

  while (1) {
    if (exA+exB==weight)
      ret_num+=((long)nuA)*((long)nuB);

    if (exA+exB>weight)
      extract_B();
    else extract_A();
    if (exA==+INF || exB==-INF)
      return ret_num;
  }
}

void print_out(long sum)
{
  FILE *out = fopen("output.txt","w");
  fprintf(out,"%ld\n",sum);
  fclose(out);

}
```



im4u
InforMatics for you

두 배열의 합

한 배열 $A[1..n]$에 대하여 부배열은 $1 \le i \le j \le n$인 $A[i..j]$를 말한다. 부배열 $A[i..j]$의 합은 $A[i]+\cdots+A[j]$이다. 원소가 양의 정수인 두 개의 배열 $A[1..n]$와 $B[1..m]$가 주어져 있다. A의 부배열의 합에 B의 부배열의 합을 더해서 t가 되는 모든 부배열 쌍의 개수를 구하는 프로그램을 작성하시오.

예를 들어, A = (1, 3, 1, 2), B = (1, 3, 2)이고 t=5인 경우, 합이 5가 되는 A와 B의 부배열 쌍의 개수는 아래와 같이 7가지이다.

$$
\begin{aligned}
&A[1]+B[1]+B[2]\\
&=A[1]+A[2]+B[1]\\
&=A[2]+B[3]\\
&=A[2]+A[3]+B[1]\\
&=A[3]+B[1]+B[2]\\
&=A[3]+A[4]+B[3]\\
&=A[4]+B[2]\\
&=5
\end{aligned}
$$

실행 파일의 이름을 SUM.EXE로 하고, 프로그램의 실행시간은 10초를 초과할 수 없다. 부분점수는 없다.

입력 형식

입력 파일의 이름은 INPUT.TXT이다. 첫째 줄에는 t($\le$10000)가 주어진다. 둘째 줄에는 배열 A의 크기 n($\le$1000)이 주어지고, 셋째 줄에는 A의 원소들이 $A[1]$부터 차례대로 주어진다. 넷째 줄에는 배열 B의 크기 m($\le$1000)이 주어지고, 마지막 줄에는 B의 원소들이 $B[1]$부터 차례대로 주어진다. 원소의 값은 1000을 넘지 않는 양의 정수이며 숫자 사이에는 빈칸이 하나 있다.

출력 형식

출력 파일의 이름은 OUTPUT.TXT이다. 첫째 줄에 합이 t가 되는 A와 B의 부배열 쌍의 개수를 출력한다. 합이 t가 되는 부배열 쌍이 없는 경우에는 숫자 0을 출력한다.

입력과 출력의 예

입력(INPUT.TXT)

```
5
4
1 3 1 2
3
1 3 2
```

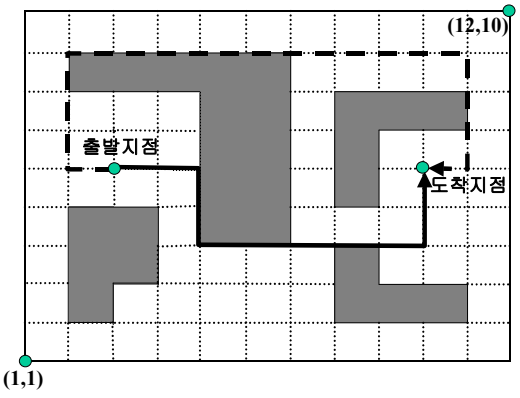
출력(OUTPUT.TXT)

```
7
```

로봇

이차원 평면상에 변들이 좌표축에 평행한 L-모양 다각형의 장애물들이 있다. 이들 장애물들은 서로 겹치지 않고 또한 변이나 꼭지점에서 만나지 않는다. 로봇이 어느 한 지점에서 출발하여 정해진 도착 지점으로 이들 장애물들을 피해서 이동하고자 한다. 이 로봇은 항상 좌표축과 평행하게 이동하며 장애물의 변을 따라서 이동할 수도 있다. 로봇이 출발지점에서 도착지점까지 이동하는 여러 경로들 중에서 꺾이는 횟수가 최소가 되는 경로를 찾고자 한다. 단, 로봇의 크기를 무시하여 로봇을 이차원 상의 점으로 표현할 수 있다고 가정한다. 또한, 로봇의 출발지점과 도착지점은 항상 장애물의 내부나 경계선에 놓여있지 않다.

아래 그림의 예를 통해서 로봇이 도착지점에 도달하기 위한 여러 경로가 있음을 알 수 있고, 또한 이 경우에서는 꺾이는 수가 3이 최소임을 알 수 있다.

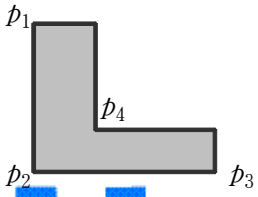


L-모양 다각형의 장애물들, 출발지점, 도착지점이 입력으로 주어져 있을 때 로봇이 출발지점에서 도착지점으로 이동하는 경로 중 꺾이는 횟수가 최소인 것을 찾는 프로그램을 작성하시오.

실행 파일의 이름은 ROBOT.EXE로 하고 실행 시간은 5초를 초과할 수 없다. 부분점수는 없다.

입력 형식

입력 파일의 이름은 INPUT.TXT이다. 첫째 줄에는 로봇의 출발지점 좌표가 주어지고 둘째 줄에는 로봇의 도착지점 좌표가 주어진다. 셋째 줄에는 장애물의 개수(≤50)가 주어지고, 그 다음 줄부터 각 줄에는 하나의 장애물의 위치를 나타내는 네 점의 좌표 p_1, p_2, p_3, p_4가 차례로 주어지고 이 네 점의 좌표의 위치는 다음의 그림과 같다. 단, 모든 좌표는 x-좌표, y-좌표가 한 칸씩 띄어서 순서대로 주어지며, 좌표의 각 값은 자연수(≤100)이다.



출력 형식

출력 파일의 이름은 OUTPUT.TXT이다. 로봇의 경로들 중에서 꺾이는 횟수가 최소가 되는 경로의 꺾이는 횟수를 출력한다.

입력과 출력의 예

입력(INPUT.TXT)

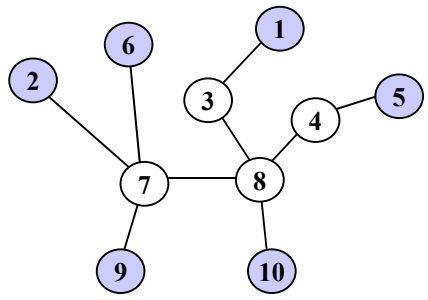
```
3 6
10 6
4
8 4 8 2 11 2 9 3
4 5 2 5 2 2 3 3
7 4 7 9 2 9 5 8
11 8 8 8 8 5 9 7
```

출력(OUTPUT.TXT)

```
3
```

버스 노선

<그림 1>은 어떤 도시의 도로망을 나타내고 있다.



<그림 1>

이 도시의 지점은 숫자를 포함하는 동그라미로 표시되어 있고, 두 지점 사이에 있는 도로는 두 지점을 잇는 선으로 표시되어 있다. 이 도로망의 특성은 다음과 같다.

특성 1. 임의의 한 지점에서 도로와 지점을 거쳐 모든 다른 지점으로 갈 수 있다.

특성 2. 한 지점에서 출발하여 어떤 도로를 두 번 이상 거치지 않고는 출발지점으로 되돌아올 수 없다.

특성 3. 한 지점과 접한 도로의 개수는 10이하이다.

위의 세 가지 특성을 만족하는 도로망을 **트리 도로망**이라 한다. 이제 이 도시에서 몇 개의 버스 노선을 신설하려고 한다. 각각의 버스노선은 한 종점에서 반대편 종점까지 가는 도로와 지점으로 이루어진다. 종점이 될 수 있는 지점은 도로망에서 단말지점(자신과 연결된 다른 지점이 하나 뿐인 곳)이어야 한다. 예를 들어, <그림 1>에서 버스 종점이 될 수 있는 지점은 색칠된 1, 2, 5, 6, 9, 10번 지점이다.

<그림 1>과 같은 경우에 두 개씩의 단말지점으로 짝을 지어 세 개의 서로 다른 버스노선을 만들 수 있다. 단, 버스 노선을 설정하기 위해서는 다음 조건들을 만족해야 한다.

조건 1. 도시의 모든 지점은 반드시 하나의 노선에 포함되어야 하고, 두 개 이상의 버스 노선에 포함될 수도 있다. 예를 들어, 1-3-8-10 노선과 9-7-8-4-5 노선과 같이 한 교차지점을 공유하는 것은 허용된다.

조건 2. 도시의 모든 도로는 하나의 버스노선에는 포함되어야 한다. 그러나 한 도로는 두 개의 버스노선에 포함될 수는 없다. 예로 <그림 1>에서 6-7-8-10이 버스노선인 경우, 9-7-8-4-5는 도로 (7, 8)을 공유하게되므로 버스노선이 될 수 없다.

조건 3. 버스노선의 종점은 단말지점이어야 한다. 그리고 버스노선은 지점과 도로를 한 번씩만 지나야 한다. 예로 <그림 1>에서 2-7-8-3-1은 버스노선으로 가능하지만 2-7-9-7-8-10은 도로 (7, 9)와 지점 7을 두 번 방문하기 때문에 버스노선이 될 수 없다.

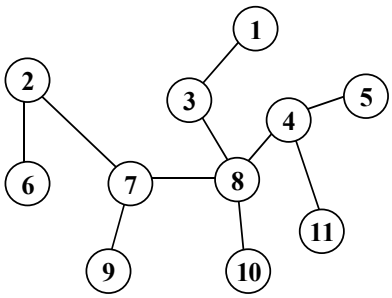
조건 4. 노선이 지나치게 길면 안되므로 위의 세 가지 조건을 만족하는 버스노선 중에서 가장 긴 노선의 길이가 최대한 짧게 설계되어야 한다. 단, 모든 도로의 길이는 1로 가정한다.

<그림 1>에서 제한조건을 만족하는 버스노선은 다음과 같다.

2-7-8-10
9-7-6
1-3-8-4-5

이 경우 가장 긴 노선의 길이가 4이다.

다음 <그림 2>와 같은 경우라면 위의 조건들을 만족시키는 버스노선은 존재하지 않는다. 왜냐하면 6-2-7-9가 버스노선이 되면 도로 (7, 8)은 다른 노선에 포함될 수 없고, 6번에서 시작하여 9번이 아닌 다른 곳이 종점이 된다면 9번의 다른 쪽 종점을 정할 수 없기 때문이다.



<그림 2>

트리도로망이 주어졌을 때, 위의 조건들을 만족하는 버스노선을 찾는 프로그램을 작성하시오.

실행 파일의 이름은 BUS.EXE로 하고, 프로그램의 실행시간은 10초를 초과할 수 없다. 부분점수는 없다.

입력 형식

입력 파일의 이름은 INPUT.TXT이다. 입력 파일의 첫째 줄에는 도로망에 있는 지점 개수 n ($2 \le n \le 500$)이 주어진다. 둘째 줄부터 $n-1$ 개의 도로에 대한 정보가 한 줄에 하나씩 주어진다. 만일 지점 i와 지점 j사이에 도로가 있다면 "i j"로 한 줄에 표시한다. 또는 "j i"라고 표시될 수도 있다. 숫자 사이에는 빈칸이 하나 있다. 지점은 1부터 n까지의 서로 다른 숫자로 표시된다.

출력 형식

출력 파일의 이름은 OUTPUT.TXT이다. 출력 파일의 첫째 줄에는 제일 긴 노선의 길이를 출력한다. 둘째 줄에는 버스노선의 개수 m을 출력한다. 다음 m줄의 각 줄에는 위의 조건을 만족하는 버스노선을 한 줄에 하나씩 출력한다. 한 버스노선은 한 종점 번호에서부터 다른 종점까지 거치는 지점 번호들을 차례로 출력한다. 지점 번호 사이에는 빈칸을 하나 둔다. 답이 여러 개인 경우는 그 중에 하나만 출력한다. 만일 위 네 가지 조건을 만족하는 답이 존재하진 않을 경우에는 첫째 줄에 숫자 0을 출력한다.

입력과 출력의 예

입력(INPUT.TXT)

```
10
1 3
3 8
8 4
4 5
8 10
7 8
6 7
2 7
9 7
```

출력(OUTPUT.TXT)

```
4
3
2 7 8 10
9 7 6
1 3 8 4 5
```

모범답안

1. SUM

```
Program Problem_M_1;

Const
    MAX_N = 1000;
    INPUT_FILE = 'input.txt';
    OUTPUT_FILE = 'output.txt';
```

```
Var
    T : Integer;
    n , m : Integer;
    A , B : Array[1..MAX_N] of Integer;
    i , j : Integer;
    PreSum , Result : LongInt;
    ouf : Text;

Procedure Input_data;

Var
    i : Integer;
    inf : Text;

Begin

    Assign(inf, INPUT_FILE);
    Reset(inf);

    Readln(inf, T);
    Readln(inf, n);

    for i:=1 to n do
        Read(inf, A[i]);
    Readln(inf);

    Readln(inf, m);
    for i:=1 to m do
        Read(inf, B[i]);
    Readln(inf);

    Close(inf);

End;
```

2. ROBOT

```
Program ROBOT;

Const
      in_file='input.txt';
      out_file='output.txt';
      size=52;

Type
     p_type=record
     x,y:integer;
     end;
     rrr=array [0..105] of shortint;


Var
    fi,fo:text;
    dap,maxx,minx,maxy,miny,e1,e2,st,ed,edd,c,xc,yc,n:integer;
    hx,hy,hx1,hy1:integer;
    p:array [1..size,1..4] of p_type;
    que:array [1..12000] of p_type;
    pp1,pp2:Array [1..size*3] of p_type;
    bd:array [0..105] of ^rrr;
    tys:Array [1..size] of integer;
    a,b:p_type;



Procedure Read_data;
var
    i,j:integer;

    Procedure insertx(a:integer);
    var
        i:integer;
        eq:boolean;
    begin
          if a>maxx then maxx:=a;
          if a<minx then minx:=a;
    end;

    Procedure inserty(a:integer);
    var
        eq:boolean;
        i:integer;
    begin
          if a>maxy then maxy:=a;
          if a<miny then miny:=a;
    end;

begin
    for i:=0 to 105 do
    begin
        new(bd[i]);
        for j:=0 to 105 do
            bd[i]^[j]:=0;
    end;
    minx:=30000;miny:=30000;
    maxx:=-30000;maxy:=-30000;


    assign(fi,in_file);
    reset(fi);
    read(fi,a.x,a.y,b.x,b.y);
    read(fi,n);
    for i:=1 to n do
    begin
          read(fi,p[i,1].x,p[i,1].y,p[i,2].x,p[i,2].y,
                  p[i,3].x,p[i,3].y,p[i,4].x,p[i,4].y);
          for j:=1 to 4 do
          begin
                insertx(p[i,j].x);
                inserty(p[i,j].y);
          end;
    end;

    insertx(a.x);
    insertx(b.x);
    inserty(a.y);
    inserty(b.y);
    close(fi);
```

```
end;

Function min(a,b:integer):integer;
begin
    if a>b then min:=b else min:=a;
end;

Function max(a,b:integer):integer;
begin
    if a>b then max:=a else max:=b;
end;

Procedure Swap(var a,b:integer);
var
  t:integer;
begin

    t:=a;
    a:=b;
    b:=t;
end;

Procedure Pr;
var
  i,j:integer;
begin
    for i:=yc downto 1 do
    begin
        for j:=1 to xc do
        begin
            if (i=b.y) and (j=b.x) then
                write('*',bd[i]^[j]:3) else
            write(bd[i]^[j]:3);
        end;
        writeln;
    end;
    readln;
end;

Procedure Solve_mlp;
var
  fx,fy,i,j,ii,k:integer;
  x1,x2,y1,y2:integer;
  chk:boolean;

  Procedure make(a,b:p_type);
  var
    i:integer;
  begin
      if a.x=b.x then
      begin
          y1:=min(a.y,b.y);
          y2:=max(a.y,b.y);
          for i:=y1 to y2 do
          begin
              e1:=i;e2:=a.x;
              bd[i]^[a.x]:=-1;
          end;
      end else
      begin
          x1:=min(a.x,b.x);
          x2:=max(a.x,b.x);
          for i:=x1 to x2 do
          begin
              bd[a.y]^[i]:=-1;
          end;
      end;
  end;

  Function Intro(xx,yy:real):boolean;
  var
    i:integer;
  begin
      intro:=true;
      for i:=1 to n do
      begin
          case tys[i] of
          1:
          begin
              x1:=min(p[i,2].x,p[i,3].x);
              x2:=max(p[i,2].x,p[i,3].x);
              y1:=min(p[i,2].y,p[i,4].y);
              y2:=max(p[i,2].y,p[i,4].y);

              if (x1<xx) and (x2>xx) and
                  (y1<yy) and (y2>yy) then exit;

              x1:=min(p[i,2].x,p[i,4].x);
              x2:=max(p[i,2].x,p[i,4].x);
              y1:=min(p[i,2].y,p[i,1].y);
              y2:=max(p[i,2].y,p[i,1].y);

              if (x1<xx) and (x2>xx) and
                  (y1<yy) and (y2>yy) then exit;
          end;
          2:
          begin
              x1:=min(p[i,1].x,p[i,2].x);
              x2:=max(p[i,1].x,p[i,2].x);
              y1:=min(p[i,1].y,p[i,4].y);
              y2:=max(p[i,1].y,p[i,4].y);

              if (x1<xx) and (x2>xx) and
                  (y1<yy) and (y2>yy) then exit;

              x1:=min(p[i,2].x,p[i,4].x);
              x2:=max(p[i,2].x,p[i,4].x);
              y1:=min(p[i,2].y,p[i,3].y);
              y2:=max(p[i,2].y,p[i,3].y);

              if (x1<xx) and (x2>xx) and
                  (y1<yy) and (y2>yy) then exit;
          end;
          end;{Case}
      end;
      intro:=false;
  end;


begin

    for i:=1 to n do
    begin
```

```
        make(p[i,1],p[i,2]);
        make(p[i,2],p[i,3]);
        if p[i,1].x=p[i,2].x then
        begin
            tys[i]:=1;
            pp1[i].x:=p[i,4].x;
            pp1[i].y:=p[i,1].y;
            pp2[i].x:=p[i,3].x;
            pp2[i].y:=p[i,4].y;
            make(p[i,1],pp1[i]);
            make(pp1[i],p[i,4]);
            make(p[i,4],pp2[i]);
            make(pp2[i],p[i,3]);
        end else
        begin
            tys[i]:=2;
            pp1[i].x:=p[i,1].x;
            pp1[i].y:=p[i,4].y;
            pp2[i].x:=p[i,4].x;
            pp2[i].y:=p[i,3].y;
            make(p[i,1],pp1[i]);
            make(pp1[i],p[i,4]);

            make(p[i,4],pp2[i]);
            make(pp2[i],p[i,3]);
        end;
    end;

    if bd[a.y]^[a.x]=0 then bd[a.y]^[a.x]:=1
                       else bd[a.y]^[a.x]:=-1;

    que[1].x:=a.x;
    que[1].y:=a.y;
    c:=1;
    st:=1;ed:=1;edd:=1;

    {BFS}
    repeat

        ed:=edd;
        for i:=st to ed do
        begin
            hx1:=que[i].x;
            hy1:=que[i].y;

            {right}
            hx:=hx1:hy:=hy1;
            hx:=hx;
            repeat
                if hx>=maxx then break;
                if bd[hy]^[hx]<0 then chk:=true else
chk:=false;

                inc(hx);
                if chk then
                  if intro(hx-0.5,hy) then break;
                if (abs(bd[hy]^[hx])<>0) and
(bd[hy]^[hx]<>-1) then continue;

                inc(edd);
                que[edd].x:=hx;
                que[edd].y:=hy;
                if bd[hy]^[hx]=0 then
bd[hy]^[hx]:=abs(bd[hy1]^[hx1])+1 else

bd[hy]^[hx]:=(abs(bd[hy1]^[hx1])+1)*-1;
            until i=0;

            {left}
            hx:=hx1;hy:=hy1;
            repeat
                if hx<=minx then break;
                if bd[hy]^[hx]<0 then chk:=true else
chk:=false;

                dec(hx);
                if chk then
                  if intro(hx+0.5,hy) then break;
                if (abs(bd[hy]^[hx])<>0) and
(bd[hy]^[hx]<>-1) then continue;

                inc(edd);
                que[edd].x:=hx;
                que[edd].y:=hy;
                if bd[hy]^[hx]=0 then
bd[hy]^[hx]:=abs(bd[hy1]^[hx1])+1 else

bd[hy]^[hx]:=(abs(bd[hy1]^[hx1])+1)*-1;
            until i=0;

            {up}
            hx:=hx1:hy:=hy1;
            repeat
                if hy>=maxy then break;
                if bd[hy]^[hx]<0 then chk:=true else
chk:=false;

                inc(hy);
                if chk then
                  if intro(hx,hy-0.5) then break;
                if (abs(bd[hy]^[hx])<>0) and
(bd[hy]^[hx]<>-1) then continue;

                inc(edd);
                que[edd].x:=hx;
                que[edd].y:=hy;
                if bd[hy]^[hx]=0 then
bd[hy]^[hx]:=abs(bd[hy1]^[hx1])+1 else

bd[hy]^[hx]:=(abs(bd[hy1]^[hx1])+1)*-1;
            until i=0;

            {down}
            hx:=hx1;hy:=hy1;
            repeat
                if hy<=miny then break;
                if bd[hy]^[hx]<0 then chk:=true else
```

```
chk:=false;
                    dec(hy);
                    if chk then
                     if intro(hx,hy+0.5) then break;
                    if (abs(bd[hy]^[hx])<>0) and
(bd[hy]^[hx]<>-1) then continue;


                    inc(edd);
                    que[edd].x:=hx;
                    que[edd].y:=hy;
                    if bd[hy]^[hx]=0 then
bd[hy]^[hx]:=abs(bd[hy1]^[hx1])+1 else

bd[hy]^[hx]:=(abs(bd[hy1]^[hx1])+1)*-1;
                until i=0;
                st:=st;
             end;  {for}
             st:=ed+1;
            {pr;}

      until (st>edd) or (bd[b.y]^[b.x]<>0);

      dap:=(abs(bd[b.y]^[b.x])-2);
      assign(fo,out_file);
      rewrite(fo);
      writeln(fo,dap);
      close(fo);

end;


begin
      Read_Data;
      Solve_MLP;
end.
```

3. BUS

```
{$A+,B-,D-,E+,F-,G-,I+,L-,N-,O+,P-,Q-,R-,S-,T-,V+,X+,Y+}
{$M 65500,0,655360}

{ Written by An, Hyung-Chan }

Program Bus;

const
      MAX = 501;
      DMX = 11;
      INFI = 10000;
      inpf = 'input.txt';
      outf = 'output.txt';

var
    adl : array [1..MAX,1..DMX] of integer;
    adn : array [1..MAX] of integer;
    n,sn : integer;
    sol : array [1..MAX,1..2] of integer;


procedure rd;

var
    f : text;
    i,a,b : integer;

begin
      assign(f,inpf);
      reset(f);
      readln(f,n);
      for i:=1 to n-1 do
      begin
            readln(f,a,b);
            inc(adn[a]);
            inc(adn[b]);
            adl[a,adn[a]]:=b;
            adl[b,adn[b]]:=a
      end;
      close(f)
end;

function possible(lim : integer) : boolean;

var
    dummy1,dummy2 : integer;
    p : boolean;

    function search(v,p : integer;var up,ul : integer) : boolean;

    var
        r,l : array [1..DMX] of integer;
        k : integer;
        i,j,a,b,t,m,bsn : integer;
        flag : boolean;
    begin
          m:=0;
          k:=MAX+1;
          for i:=1 to adn[v] do
              if adl[v,i]<>p then
                        begin
                            inc(m);
                            if not search(adl[v,i],v,r[m],l[m])
then

          begin

              search:=false;

              exit

          end
                        end
                        else k:=i;
          for i:=1 to m-1 do
              for j:=i+1 to m do
                  if r[i]>r[j] then
                              begin
                                  t:=r[i];
                                  r[i]:=r[j];
                                  r[j]:=t;
                                  t:=l[i];
                                  l[i]:=l[j];
                                  l[j]:=t
```

```
                    end;
            if p=-1 then
                begin
                    if m=0 then
                        begin
                            search:=true;
                            exit
                        end;
                    if m mod 2=1 then
                        begin
                            if m>1 then
                                begin

search:=false;
                                        exit
                                    end;
                                search:=(r[1]+1<=lim);
                                inc(sn);
                                sol[sn,1]:=v;
                                sol[sn,2]:=l[1]
                            end
                            else
                            begin
                                for i:=1 to m div 2 do
                                    i                   f
r[i]+r[m+1-i]+2>lim then

    begin
        search:=false;
        exit
    end;
                                for i:=1 to m div 2 do
                                begin
                                    inc(sn);
                                    sol[sn,1]:=l[i];
sol[sn,2]:=l[m+1-i]
                                end;
                                search:=true
                            end
                end
                else
                begin
                    if m=0 then
                        begin
                            up:=0;
                            ul:=v;
                            search:=true;
                            exit
                        end;
                    if m mod 2=0 then
                        begin
                            search:=false;
                            exit
                        end;
                    for i:=1 to m do
                    begin
                        flag:=true;
                        bsn:=sn;
                        for j:=1 to m div 2 do
                        begin
                            a:=j;
                            b:=m-j;
                            if a>=i then inc(a);
                            if b>=i then inc(b);
                            inc(sn);
                            sol[sn,1]:=l[a];
                            sol[sn,2]:=l[b];
                            if r[a]+r[b]+2>lim then
                                begin
flag:=false;

break
                                end
                        end;
                        if flag then
                            begin
                                up:=r[i]+1;
                                ul:=l[i];
                                search:=true;
                                exit
                            end
                            else sn:=bsn
                    end;
                    search:=false
                end
    end;

begin
    sn:=0;
    possible:=search(1,-1,dummy1,dummy2)
end;

function pr : integer;

var
    s,e,m,lastm : integer;
    lastp : boolean;

begin
    lastm:=-1;
    s:=1;
    e:=n;
    while s<e do
    begin
        m:=(s+e) div 2;
        lastm:=m;
        lastp:=possible(m);
        if lastp then e:=m else s:=m+1
    end;
    m:=(s+e) div 2;
    if m<>lastm then lastp:=possible(m);
    if lastp then pr:=m else pr:=0
end;

procedure wr(sv : integer);

var
    f : text;
    i : integer;

    procedure printpath(s,e : integer);

    var
        path : array [1..MAX] of integer;
        pn : integer;
        flag : boolean;
```

```
        procedure traverse(v,p : integer);

        var
            i : integer;

        begin
                inc(pn);
                path[pn]:=v;
                if v=e then
                        begin
                                for i:=1 to pn-1 do
                                        write(f,path[i],' ');
                                writeln(f,path[pn]);
                                flag:=true;
                                exit
                        end;
                for i:=1 to adn[v] do
                        if adl[v,i]<>p then
                                        begin
                                                traverse(adl[v,i],v);
                                                if flag then exit
                                        end;
                dec(pn)
        end;

    begin
            pn:=0;
            flag:=false;
            traverse(s,-1)
    end;

begin
        assign(f,outf);
        rewrite(f);
        writeln(f,sv);
        if sv>0 then
                begin
                        writeln(f,sn);
                        for i:=1 to sn do printpath(sol[i,1],sol[i,2])
                end;
        close(f)
end;

begin
        rd;
        wr(pr)

end.
```