NOTE: Each problem has a point value. Your goal is to maximize the

number of points you score. Your score is the sum of your three

highest-scoring submissions. Increase your score by correctly solving

tasks; maximize your score by correctly solving tasks with the

higher point values.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

GOLD PROBLEMS

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Four problems numbered 1 through 4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Problem 1: Trick or Treat on the Farm [Jacob Steinhardt, 2008]

Every year in Wisconsin the cows celebrate the USA autumn holiday

of Halloween by dressing up in costumes and collecting candy that

Farmer John leaves in the N (1 <= N <= 100,000) stalls conveniently

numbered 1..N.

Because the barn is not so large, FJ makes sure the cows extend

their fun by specifying a traversal route the cows must follow. To

implement this scheme for traveling back and forth through the barn,

FJ has posted a "next stall number" next\_i (1 <= next\_i <= N) on

stall i that tells the cows which stall to visit next; the cows

thus might travel the length of the barn many times in order to

collect their candy.

FJ mandates that cow i should start collecting candy at stall i.

A cow stops her candy collection if she arrives back at any stall

she has already visited.

Calculate the number of unique stalls each cow visits before being

forced to stop her candy collection.

POINTS: 100

PROBLEM NAME: treat

INPUT FORMAT:

\* Line 1: A single integer: N

\* Lines 2..N+1: Line i+1 contains a single integer: next\_i

SAMPLE INPUT (file treat.in):

4

1

3

2

3

INPUT DETAILS:

Four stalls.

\* Stall 1 directs the cow back to stall 1.

\* Stall 2 directs the cow to stall 3

\* Stall 3 directs the cow to stall 2

\* Stall 4 directs the cow to stall 3

OUTPUT FORMAT:

\* Lines 1..N: Line i contains a single integer that is the total

number of unique stalls visited by cow i before she returns to

a stall she has previously visited.

SAMPLE OUTPUT (file treat.out):

1

2

2

3

OUTPUT DETAILS:

Cow 1: Start at 1, next is 1. Total stalls visited: 1.

Cow 2: Start at 2, next is 3, next is 2. Total stalls visited: 2.

Cow 3: Start at 3, next is 2, next is 3. Total stalls visited: 2.

Cow 4: Start at 4, next is 3, next is 2, next is 3. Total stalls visited: 3.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Problem 2: Secret Message [David Benjamin and Jacob Steinhardt, 2008]

Bessie is leading the cows in an attempt to escape! To do this, the

cows are sending secret binary messages to each other.

Ever the clever counterspy, Farmer John has intercepted the first

b\_i (1 <= b\_i <= 10,000) bits of each of M (1 <= M <= 50,000) of

these secret binary messages.

He has compiled a list of N (1 <= N <= 50,000) partial codewords

that he thinks the cows are using. Sadly, he only knows the first

c\_j (1 <= c\_j <= 10,000) bits of codeword j.

For each codeword j, he wants to know how many of the intercepted

messages match that codeword (i.e., for codeword j, how many times

does a message and the codeword have the same initial bits). Your

job is to compute this number.

The total number of bits in the input (i.e., the sum of the b\_i and

the c\_j) will not exceed 500,000.

Memory Limit: 32MB

POINTS: 270

PROBLEM NAME: sec

INPUT FORMAT:

\* Line 1: Two integers: M and N

\* Lines 2..M+1: Line i+1 describes intercepted code i with an integer

b\_i followed by b\_i space-separated 0's and 1's

\* Lines M+2..M+N+1: Line M+j+1 describes codeword j with an integer

c\_j followed by c\_j space-separated 0's and 1's

SAMPLE INPUT (file sec.in):

4 5

3 0 1 0

1 1

3 1 0 0

3 1 1 0

1 0

1 1

2 0 1

5 0 1 0 0 1

2 1 1

INPUT DETAILS:

Four messages; five codewords.

The intercepted messages start with 010, 1, 100, and 110.

The possible codewords start with 0, 1, 01, 01001, and 11.

OUTPUT FORMAT:

\* Lines 1..N: Line j: The number of messages that the jth codeword

could match.

SAMPLE OUTPUT (file sec.out):

1

3

1

1

2

OUTPUT DETAILS:

0 matches only 010: 1 match

1 matches 1, 100, and 110: 3 matches

01 matches only 010: 1 match

01001 matches 010: 1 match

11 matches 1 and 110: 2 matches

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Problem 3: Winning Checkers [Rob Kolstad, 2008]

The cows have taken up the game of checkers with a vengeance.

Unfortunately, despite their unlimited enjoyment of playing, they

are terrible at the endgame and want your help.

Given an NxN (4 <= N <= 500) checkboard, determine the optimal set

of moves (i.e., smallest number of moves) to end the game on the

next move. Checkers move only on the '+' squares and capture by

jumping 'over' an opponent's piece in the traditional way. The piece

is removed as soon as it is jumped. See the example below where

N=8:

- + - + - + - + The K's mark Bessie's kings; the o's represent the

+ - + - + - + - opponent's checkers. Bessie always moves next. The

- + - K - + - + Kings jump opponent's checkers successively in any

+ - + - + - + - diagonal direction (and removes pieces when jumped).

- o - o - + - +

+ - K - + - + - For this board, the best solution requires the lower

- o - + - + - + left King to jump successively across all three of the

+ - K - + - K - opponents' checkers, thus ending the game (moving K

marked as >K<):

Original After move 1 After move 2 After move 3

- + - + - + - + - + - + - + - + - + - + - + - + - + - + - + - +

+ - + - + - + - + - + - + - + - + - + - + - + - + - + - + - + -

- + - K - + - + - + - K - + - + - + - K - + - + - + - K - + - +

+ - + - + - + - + - + - + - + - + ->K<- + - + - + - + - + - + -

- o - o - + - + - o - o - + - + - + - o - + - + - + - + - + - +

+ - K - + - + - >K<- K - + - + - + - K - + - + - + - K ->K<- + -

- o - + - + - + - + - + - + - + - + - + - + - + - + - + - + - +

+ ->K<- + - K - + - + - + - K - + - K - + - K - + - K - + - K -

The moves traversed these squares:

1 2 3 4 5 6 7 8 R C

1 - + - + - + - + start: 8 3

2 + - + - + - + - move: 6 1

3 - + - K - + - + move: 4 3

4 + - \* - + - + - move: 6 5

5 - o - o - + - +

6 \* - K - \* - + -

7 - o - + - + - +

8 + - K - + - K -

Write a program to determine the game-ending sequence for an NxN

input board if it exists. There is at least a king and at least one

opponent piece on the board. The king can jump a piece on every move

of the optimal solution.

POINTS: 330

PROBLEM NAME: winchk

INPUT FORMAT:

\* Line 1: A single integer: N

\* Lines 2..N+1: Line i+1 contains N characters (each one of: '-', '+',

'K', or 'o') that represent row i of a proper checkboard. Line

2 always begins with '-'.

SAMPLE INPUT (file winchk.in):

8

-+-+-+-+

+-+-+-+-

-+-K-+-+

+-+-+-+-

-o-o-+-+

+-K-+-+-

-o-+-+-+

+-K-+-K-

OUTPUT FORMAT:

\* Lines 1..?: If there is no winning sequence of jump, output

"impossible" on a line by itself. If such a sequence exists,

each line contains two space-separated integers that represent

successive locations of a king whose moves will win the game.

Any such sequence is acceptable.

SAMPLE OUTPUT (file winchk.out):

8 3

6 1

4 3

6 5

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Problem 4: Largest Fence [Zoran Dzunic, 2008]

Farmer John has purchased N (5 <= N <= 250) fence posts in order

to build a very nice-looking fence. Everyone knows the best fences

are convex polygons where fence posts form vertices of a polygon.

The pasture is represented as a rectilinear grid; fencepost i is

at integer coordinates (x\_i, y\_i) (1 <= x\_i <= 1,000; 1 <= y\_i <=

1000).

Given the locations of N fence posts (which, intriguingly, feature

no set of three points which are collinear), what is the largest

number of fence posts FJ can use to create a fence that is convex?

For test cases worth 45% of the points for this problem, N <= 65.

Time limit: 1.2 seconds

POINTS: 400

PROBLEM NAME: fence

INPUT FORMAT:

\* Line 1: A single integer: N

\* Lines 2..N+1: Line i+1 describes fence post i's location with two

space-separated integers: x\_i and y\_i

SAMPLE INPUT (file fence.in):

6

5 5

2 3

3 2

1 5

5 1

1 1

INPUT DETAILS:

A square with two points inside.

OUTPUT FORMAT:

\* Line 1: A single integer, the maximum possible number of fence posts

that form a convex polygon.

SAMPLE OUTPUT (file fence.out):

5

OUTPUT DETAILS:

The largest convex polygon is the pentagon (2,3), (3,2), (5,1), (5,5), (1,5).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*