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

GOLD PROBLEMS

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

Three problems numbered 1 through 3

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

Problem 1: Earthquake Damage [Hal Burch, 2004]

Wisconsin has had an earthquake that has struck Farmer John's farm!

The earthquake has damaged some of the pastures so that they are

unpassable. Remarkably, none of the cowpaths was damaged.

As usual, the farm is modeled as a set of P (1 <= P <= 30,000)

pastures conveniently numbered 1..P which are connected by a set

of C (1 <= C <= 100,000) non-directional cowpaths conveniently

numbered 1..C. Cowpath i connects pastures a\_i and b\_i (1 <= a\_i

<= P; 1 <= b\_i <= P). Cowpaths might connect a\_i to itself or perhaps

might connect two pastures more than once. The barn is located in

pasture 1.

A total of N (1 <= N <= P) cows (in different pastures) sequentially

contact Farmer John via moobile phone with an integer message

report\_j (2 <= report\_j <= P) that indicates that pasture report\_j

is undamaged but that the calling cow is unable to return to the

barn from pasture report\_j because she could not find a path that

does not go through damaged pastures.

After all the cows report in, determine the minimum number of

pastures (including ones that are uncrossable) from which it is not

possible to return to the barn.

Note: Feedback on some of the test data will be provided on the

first 50 submissions

PROBLEM NAME: damage

INPUT FORMAT:

\* Line 1: Three space-separated integers: P, C, and N

\* Lines 2..C+1: Line i+1 describes cowpath i with two integers: a\_i

and b\_i

\* Lines C+2..C+N+1: Line C+1+j contains a single integer: report\_j

SAMPLE INPUT (file damage.in):

4 3 1

1 2

2 3

3 4

3

OUTPUT FORMAT:

\* Line 1: A single integer that is the minimum count of pastures from

which a cow can not return to the barn (including the damaged

pastures themselves)

SAMPLE OUTPUT (file damage.out):

3

OUTPUT DETAILS:

Pasture 2 is damaged, resulting in cows in pastures 2, 3, 4 not being able to

return to the barn.

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

Problem 2: The Baric Bovine [Jeffrey Wang, 2007]

Following up on a journal article about increasing milk production,

Bessie has become the Baric Bovine by studying atmospheric pressure

in order to curry favor with Farmer John.

She takes N (1 <= N <= 100) measurements conveniently named M\_1

through M\_N during the day (1 <= M\_i <= 1,000,000); these measurements

are numbered in the order in which Bessie observed them.

In order to characterize the day's atmospheric pressure readings,

she is interested in finding a subset of the measurements (given

by the K (1 <= K <= N) indices s\_j where 1 <= s\_1 < s\_2 < ... < s\_K

<= N) that accurately reflects the entire set, i.e., limits the

error as described below.

In any subset of measurements, an error is incurred for each

measurement (1) before the first member of the subset, (2) between

two consecutive measurements in the subset, and (3) after the last

member of the subset. The total error value for a given set of s\_j

values is the sum of each of the individual errors.

Specifically, for all measurements whose index i is not one of the

s\_j values:

\* if i is less than s\_1, then the sample error is:

2 \* | M\_i - M\_(s\_1) |

\* if i is between s\_j and s\_(j+1), then the sample error is

| 2 \* M\_i - Sum(s\_j, s\_(j+1)) |

where Sum(x, y) = M\_x + M\_y;

\* if i is greater than s\_K, then the sample error is

2 \* | M\_i - M\_(s\_K) |

Given a maximum error value E (1 <= E <= 1,000,000), determine the

size of the smallest subset of measurements that produces an error

of at most E.

PROBLEM NAME: baric

INPUT FORMAT:

\* Line 1: Two space-separated integers: N and E

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

SAMPLE INPUT (file baric.in):

4 20

10

3

20

40

INPUT DETAILS:

Bessie takes four measurements; the maximum error is 20. The

measurements are, in sequence: 10, 3, 20, and 40.

OUTPUT FORMAT:

\* Line 1: Two space-separated integers: the size of the smallest

subset of measurements that produces an error of at most E and

the least possible error for the subset of that size.

SAMPLE OUTPUT (file baric.out):

2 17

OUTPUT DETAILS:

Choosing the second and fourth measurements is the best option,

giving an error of 17. The first term's error is 2\*|10-3| = 14; the

third term's error is |2\*20 - (3+40)| = 3.

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

Problem 3: Safe Travel [Long Fan, 2008]

Gremlins have infested the farm. These nasty, ugly fairy-like

creatures thwart the cows as each one walks from the barn (conveniently

located at pasture\_1) to the other fields, with cow\_i traveling to

from pasture\_1 to pasture\_i. Each gremlin is personalized and knows

the quickest path that cow\_i normally takes to pasture\_i. Gremlin\_i

waits for cow\_i in the middle of the final cowpath of the quickest

route to pasture\_i, hoping to harass cow\_i.

Each of the cows, of course, wishes not to be harassed and thus

chooses an at least slightly different route from pasture\_1 (the

barn) to pasture\_i.

Compute the best time to traverse each of these new not-quite-quickest

routes that enable each cow\_i that avoid gremlin\_i who is located

on the final cowpath of the quickest route from pasture\_1 to

pasture\_i.

As usual, the M (2 <= M <= 200,000) cowpaths conveniently numbered

1..M are bidirectional and enable travel to all N (3 <= N <= 100,000)

pastures conveniently numbered 1..N. Cowpath i connects pastures

a\_i (1 <= a\_i <= N) and b\_i (1 <= b\_i <= N) and requires t\_i (1 <=

t\_i <= 1,000) time to traverse. No two cowpaths connect the same

two pastures, and no path connects a pasture to itself (a\_i != b\_i).

Best of all, the shortest path regularly taken by cow\_i from pasture\_1

to pasture\_i is unique in all the test data supplied to your program.

By way of example, consider these pastures, cowpaths, and [times]:

1--[2]--2-------+

| | |

[2] [1] [3]

| | |

+-------3--[4]--4

TRAVEL BEST ROUTE BEST TIME LAST PATH

p\_1 to p\_2 1->2 2 1->2

p\_1 to p\_3 1->3 2 1->3

p\_1 to p\_4 1->2->4 5 2->4

When gremlins are present:

TRAVEL BEST ROUTE BEST TIME AVOID

p\_1 to p\_2 1->3->2 3 1->2

p\_1 to p\_3 1->2->3 3 1->3

p\_1 to p\_4 1->3->4 6 2->4

For 20% of the test data, N <= 200.

For 50% of the test data, N <= 3000.

TIME LIMIT: 3 Seconds

MEMORY LIMIT: 64 MB

PROBLEM NAME: travel

INPUT FORMAT:

\* Line 1: Two space-separated integers: N and M

\* Lines 2..M+1: Three space-separated integers: a\_i, b\_i, and t\_i

SAMPLE INPUT (file travel.in):

4 5

1 2 2

1 3 2

3 4 4

3 2 1

2 4 3

INPUT DETAILS:

As in the text example.

OUTPUT FORMAT:

\* Lines 1..N-1: Line i contains the smallest time required to travel

from pasture\_1 to pasture\_i+1 while avoiding the final cowpath

of the shortest path from pasture\_1 to pasture\_i+1. If no such

path exists from pasture\_1 to pasture\_i+1, output -1 alone on

the line.

SAMPLE OUTPUT (file travel.out):

3

3

6

OUTPUT DETAILS:

As in the text example.

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