# International Olympiad in Informatics
# National Olympiad in Informatics
# Hong Kong Delegation Team Formation Test 2012

## Task Overview

| Task | Type | CPU time limit | Memory limit | Score |
|---|---|---|---|---|
| Erdös Number | Batch | 2 seconds | 256MB | 100 |
| CD8+ count | Batch | 1 second | 256MB | 100 |
| Telepathy Card Game | Batch | 2 seconds | 256MB | 100 |
| Debug! | Interactive | 5 seconds | 256MB | 100 |

**Compile Flags:**
```
Pascal:  -So -XS -v0 -O2
C/C++:  -lm -w -O2 -static -static-libgcc
```

**Notice:**
C++ programmers should be aware that using C++ streams (cin / cout) may lead to I/O bottlenecks and substantially slower performance.

C/C++ programmers should use `"%I64d"` for 64-bit integers I/O.

# Erdös Number

## Problem

Paul Erdös was a Hungarian mathematician. Since he has published a lot of papers, his friend created the idea of 'Erdös Number', a humorous tribute to his enormous output.

To be assigned an Erdös number, an author must co-write a research paper with an author with a finite Erdös number. Paul Erdös has an Erdös number of 0. Anybody else's Erdös number is $k+1$ where $k$ is the lowest Erdös number of any co-author. Note that a research paper can be co-written by a group of authors.

Your task is, find the Erdös number of all authors.

## Input

The first line contains 2 integer $N$ and $M$, the number of authors (excluding Erdös) and the number of research papers.
The authors are labelled from 0 to $N$, where Erdös is labelled 0.
The next $M$ lines describe the papers. Each line starts with an integer $a_i$, the number of authors of that paper. $a_i$ integers follow, describing the authors. The integers are pairwise distinct.

## Output

The output consist of a single line with $N$ integers. The $i^{th}$ is the Erdös Number of author $i$. If the number is infinite, output $-1$.

## Sample Tests

| Input | Output |
|---|---|
| 3 2<br>2 0 1<br>3 1 2 3 | 1 2 2 |

| Input | Output |
|---|---|
| 8 5<br>3 2 0 4<br>2 6 3<br>2 1 5<br>4 2 4 7 8<br>3 6 7 8 | -1 1 4 1 -1 3 2 2 |

## Constrains

In test cases worth 50%,
$1 \le N, M \le 100$.

In all test cases,
$1 \le N, M \le 50000$,
$2 \le a_i \le N+1$,
$a_1 + a_2 + ... + a_m \le 1000000$.

# CD8+ count

## Problem

Now it is the year of 2500 AD. The 'supervirus' outbreak has recently caused a great disaster in the city. Fortunately, a marvellous physician and scientist, Dr. Jones, has invented a new drug against 'supervirus'. Nearly all infected citizens are saved.

One day, a patient is found to have similar symptoms of 'supervirus' infection, but he cannot be cured by this drug. Later, Dr. Jones has discovered that this 'supervirus' has mutated into a stronger virus called 'ultravirus'. He can do nothing for the patient but quarantining and stabilizing him.

In order to invent a new treatment against 'ultravirus' as soon as possible, he starts to monitor the patient's body condition. Two major parameters in the blood are regularly checked at different hours: increase in viral load and increase in CD8+ count. For example,

| Hours | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Increase in viral load | 1 | 2 | 1 | 3 | 3 | 1 |
| Increase in CD8+ count | 2 | 1 | 2 | 2 | 2 | 2 |

All the above figures are in arbitrary units.

While recording these figures, Dr. Jones observes the patient's actual condition as well. He has found that the patient's condition is better at the time intervals that the sum of increases in viral load is equal to those in CD8+ count. For instance, from 1st to 2nd hours, the sum of increase in viral load $= 2 + 1 = 3$, while the sum of increase in CD8+ count $= 1+2 = 3$. Dr. Jones defines such kind of time intervals as 'stable'.

It sounds a little bit reasonable, because CD8+ count means the estimated number of cytotoxic T lymphocyte (a.k.a CD8+ T-cell), an important white blood cell for killing the virus-infected cells in the body.

On the next day, Dr. Jones establishes an amazing finding. At a certain time interval $I$, the more the 'stable' sub-intervals, the higher the overall stability of the patient at $I$. He defines the number of 'stable' sub-intervals as the 'degree of stability' at the time interval $I$. For example, the 'degree of stability' from 2nd to 6th hours is 4, since there are 4 'stable' sub-intervals, namely $[2, 3], [3, 4], [5, 6]$ and $[3, 6]$.

Dr. Jones is very sure that monitoring the 'degree of stability' would be the breakthrough point for the invention of new drugs against 'ultravirus'. Given the increases in viral load and CD8+ count in $N$ hours, help Dr. Jones calculate the 'degree of stability' at different time intervals.

## Input

The first line is an integer $N$. Each of the following 2 lines contains a list of $N$ integers, representing the increases in viral load and CD8+ count.
The next line is an integer $Q$, meaning the number of time intervals Dr. Jones wants you to calculate the 'degree of stability'.
In each the last $Q$ lines, there is a pair of integers $L$ and $R$, which means the time interval from $L^{th}$ to $R^{th}$ hours.

## Output

Output $Q$ lines. In each line, output the 'degree of stability' of the corresponding time interval.

## Sample Tests

| Input | Output |
|---|---|
| 6<br>1 2 1 3 3 1<br>2 1 2 2 2 2<br>2<br>1 2<br>2 6 | 1<br>4 |

| Input | Output |
|---|---|
| 10<br>0 1 2 0 1 2 0 1 2 0<br>2 1 0 2 1 0 2 1 0 2<br>5<br>1 10<br>2 7<br>5 5<br>3 5<br>8 9 | 27<br>11<br>1<br>3<br>1 |

## Constraints

Let $M$ be the value of Increase in viral load and CD8+ count.

In test cases worth 20%,
$1 \le L \le R \le N \le 30$
$1 \le Q \le 500$
$0 \le M \le 100$

In test cases worth 40%,
$1 \le L \le R \le N \le 100$
$1 \le Q \le 5,000$
$0 \le M \le 200$

In test cases worth 60%,
$1 \le L \le R \le N \le 200$
$1 \le Q \le 50,000$
$0 \le M \le 300$

In all test cases,
$1 \le L \le R \le N \le 2,000$
$1 \le Q \le 500,000$
$0 \le M \le 500$

# Telepathy Card Game

## Problem

You and your friend are playing a card game called 'Half Kilogram Or Increase' (HKOI). In this game, you two are pirates, searching for treasures and pouring the gold you found in a shared gold pool. At the beginning, the gold pool is empty (it weighs 0 kg). The cards in this game represent events which can change the total weight of the gold pool. Both players have the same number of cards initially, and take turn to play a card from their hands until both players run out of cards. There are two types of cards:

**1. Increase cards**
You discovered a treasure trove! An increase card has a positive integer on it. When you play an increase card with the integer $X$, the total weight of the gold pool is added by $X$ kg. There may be two or more increase cards with the same integer on them.

**2. Half cards**
The infamous monster 'Hideous Kraken Of Islands' appears and tries to eat half of your gold. When you play a half card, if the total weight of the gold pool is even, it is divided by 2. Otherwise the total weight remains unchanged (because the monster did not complete primary school and can only calculate with integers). Each player has exactly one half card.

Your friend takes the first turn. As a greedy pirate, your goal is to maximize the total weight of the gold pool at the end of the game. However, your friend is new to this game and he might not be able to play the cards wisely. Fortunately, you know some mind reading tricks. By reading your friend's mind, you know which cards are in his hand, and the exact sequence of cards he is going to play. He will play the cards in that sequence no matter which cards you play. Find out the largest possible total weight if you play your cards in the correct sequence.

## Input

The first line contains an integer $N$, the number of cards in each player's hand at the beginning of the game. The second line is the list of cards in your friend's hand, in the sequence he is going to play them. A positive integer represents an increase card. A number '-2' represents a half card.
The third line is the list of cards in your hand, with the half card listed the first.

## Output

Output a line containing an integer, the maximum total weight of the gold pool at the end of the game.

## Sample Tests

| Input | Output |
|---|---|
| 4<br>36 6 -2 1<br>-2 2 14 18 | 48 |

| Input | Output |
|---|---|
| 3<br>8 4 -2<br>-2 12 16 | 26 |

## Explanation

In the first sample, the optimal sequence to play the cards is

| Turn | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| Card played | 36 | 2 | 6 | 14 | -2 | -2 | 1 | 18 |
| Total weight | 36 | 38 | 44 | 58 | 29 | 29 | 30 | 48 |

Note that odd turns are played by your friend and even turns are played by you.

In the second example, the optimal sequence to play the card is

| Turn | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Card played | 8 | -2 | 4 | 12 | -2 | 16 |
| Total weight | 8 | 4 | 8 | 20 | 10 | 26 |

## Constrains

In test cases worth 30%,
$1 \leq N \leq 10$

In test cases worth 50%,
$1 \leq N \leq 20$

In test cases worth 90%,
$1 \leq N \leq 100$

In all test cases,
$1 \leq N \leq 1000$
$1 \leq X \leq 10,000$

# Debug!

## Problem

`WARNING: BUG DETECTED`
The alert message appeared in your screen! A bug is founded in the system, and it is going to eat up everything!

Don't panic. As a programmer, your have been trained to debug. The world is made of $200 \times 200$ grid. We call the grid at the $r^{th}$ row, $c^{th}$ column as $(r, c)$. Rows and Columns are labelled from 1 to 200.

The bug occupies exactly one grid. At each second, the bug will move to an adjacent grid (Up/Down/Left/Right). Moreover, you know some special properties of how the bug moves:

- The bug will move along a simple cycle. When we say simple, it means that it will not visit the same grid more than once in a single cycle.
- The cycle length is at least 2000. You may or may not know the length of the cycle.

At each second, you will search a grid to see if the bug is hiding there. If the bug is there at that moment, then you successfully catch the bug! You may assume the bug move instantly at the beginning of each second and you search the grid after the bug moved.

If the bug is not there, you may still get some information: by measuring how smelly the grid is (bug is always smelly), you will know how long ago the bug visit that grid last time. That is, you may gain information such as 'The bug visit (5, 8) 12 seconds before'. You may assume the bug has complete at least one cycle when you start searching.

## Interactive

You are to implement a procedure `findbug(L)`. `L` is a positive integer, the length of the cycle. If `L` is $-1$, then the cycle length is not known.

Your implementation should call the procedure `search(x, y)` which is implemented by the judge. The $i^{th}$ call you made means you want to search `(x, y)` at the $i^{th}$ second. `search(x, y)` returns the following integer values:

$t$ $(t > 0)$ : the bug visit this grid $t$ seconds before
0 : you catch the bug!
$-1$ : the bug never visit this grid

Your procedure should terminate when (and only when) `search(x, y)` returns 0.

## Scoring

| Score | Cycle length | Time used (seconds) |
|---|---|---|
| 15 | Unknown | $\leq 2800$ |
| 25 | Unknown | $\leq 3200$ |
| 10 | Unknown | $\leq 10000$ |
| 5 | Unknown | $\leq 40000$ |
| 15 | Known | $\leq 3200$ |
| 15 | Known | $\leq 10000$ |
| 15 | Known | $\leq 40000$ |

Note that your program will be tested by a group of test cases and we take count of the worst performance. A single failure in a single case may lead to great loss of marks.

## Implementation Details

To help you understand the implementation, example solutions are provided with names `debug.pas`, `debug.c`, `debug.cpp`. You should **only** modify these files.

To test your implementation, run `testpas.exe`, `testc.exe`, `testcpp.exe`. The file `sample.txt` describe the cycle of the bug. The first line is the cycle length of the bug. The second line contains a single integer. If the value is 1, then the cycle length will be known. If it is 0, then the cycle length is unknown. The remaining lines describe the cycle of the bug. When you make the first search, the bug is located at the coordinate at the third line.

You are remind that the real judge program will be very different from the judge program we provided.