Good Versus Evil

Middle Earth is about to go to war. The forces of good will have many battles with the forces of evil. Different races will certainly be involved. Each race has a certain 'worth' when battling against others. On the side of good we have the following races, with their associated worth:

Hobbits - 1
Men - 2
Elves - 3
Dwarves - 3
Eagles - 4
Wizards - 10

On the side of evil we have:

Orcs - 1
Men - 2
Wargs - 2
Goblins - 2
Uruk Hai - 3
Trolls - 5
Wizards - 11

Although weather, location, supplies and valor play a part in any battle, if you add up the worth of the side of good and compare it with the worth of the side of evil, the side with the larger worth will tend to win.

Thus, given the count of each of the races on the side of good, followed by the count of each of the races on the side of evil, determine which side wins.

Input

The first line of input will contain an integer greater than 0 signifying the number of battles to process. Information for each battle will consist of two lines of data as follows.

First, there will be a line containing the count of each race on the side of good. Each entry will be separated by a single space. The values will be ordered as follows: Hobbits, Men, Elves, Dwarves, Eagles, Wizards.

The next line will contain the count of each race on the side of evil in the following order: Orcs, Men, Wargs, Goblins, Uruk Hai, Trolls, Wizards.

All values are non-negative integers. The resulting sum of the worth for each side will not exceed the limit of a 32-bit integer.

Output

For each battle, print "`Battle`" followed by a single space, followed by the battle number starting at 1, followed by a "`:`", followed by a single space. Then print "`Good triumphs over Evil`" if good wins. Print "`Evil eradicates all trace of Good`" if evil wins. If there is a tie, then print "`No victor on this battle field`".

Sample Input	Sample Output
3 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 10 0 1 1 1 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0	Battle 1: Evil eradicates all trace of Good Battle 2: Good triumphs over Evil Battle 3: No victor on this battle field

Magic Multiple

The Elvish races of Middle Earth believed that certain numbers were more significant than others. When using a particular quantity n of metal to forge a particular sword, they believed that sword would be most powerful if the thickness k were chosen according to the following rule:

Given a nonnegative integer n, what is the smallest k such that the decimal representations of the integers in the sequence:

$$n, \quad 2n, \quad 3n, \quad 4n, \quad 5n, \quad \ldots, \quad kn$$

contain all ten digits (0 through 9) at least once?

Lord Elrond of Rivendell has commissioned you with the task to develop an algorithm to find the optimal thickness (k) for any given quantity of metal (n).

Input

Input will consist of a single integer n per line. The end of input will be signaled by end of file. The input integer will be between 1 and 200,000,000, inclusive.

Output

The output will consist of a single integer per line, indicating the value of k needed such that every digit from 0 through 9 is seen at least once.

Sample Input	Sample Output
1 10 123456789 3141592	10 9 3 5

Rings and Runes

Description

Frodo has entered the mines of Moria and encountered a series of gates. Each gate has written upon it an ancient riddle describing the state of a set of special rings which control that particular gate. By examining the riddle, Frodo can determine whether or not the gate can be opened, or if it is simply a death trap.

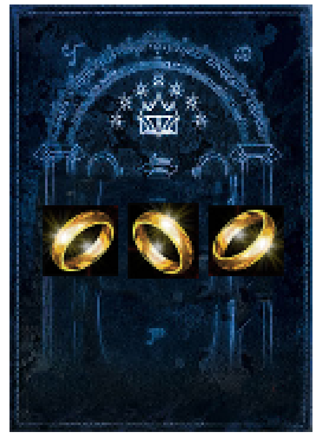


Figure 1: A gate in the Mines of Moria that is controlled by an ancient Rune

Riddles consist of multiple **runes**. A valid **rune** contains exactly **3 statements** about 3 different **rings**. Each **statement** in a rune is either **true** or **false**, depending on the **state** (spinning or not spinning) of a specific ring in the set of rings controlling the gate. The riddle for a particular gate does not have to use all of the possible rings contained in the gate's controlling set.

To open the gates, the hobbits must read the riddle, then, decide which of the rings to spin, and which of the rings to leave alone. Once they have the correct rings spinning, they say an incantation, and if the entire riddle for the gate is satisfied the gate will open. For the gate to open, every rune in the riddle must have **at least** one statement in it that is true.

For example, consider a specific rune: `1 -2 3 0`. This rune will be true if (ring 1 is spinning) **OR** (ring 2 is NOT spinning) **OR** (ring 3 is spinning). (NOTE: The `0` indicates the end of a rune, and at least one of the statements in that rune must be true for that specific rune to be true.) If a ring number in a rune is negative (e.g., `-2`), it means that ring 2 must NOT be spinning for that particular statement in the rune to be true. If the ring number is positive, (e.g., `3`) it means that ring 3 MUST be spinning for that statement in the rune to be true. A specific ring may only appear one time in a specific rune, however, a ring may be used multiple times in the entire riddle, just not in the same rune!

Input

The input will consist of the following:

- The first line of input contains a single integer g (where $1 \leq g \leq 30$), which denotes the number of gates with riddles to be decoded.
- The first line for each gate contains two integers, *rings* (where $3 \leq rings \leq 22$) and *runes* (where $1 \leq runes \leq 100$), separated by a space. *rings* is the number of rings in the controlling set; each ring is numbered from 1 to *rings*, and riddles are not required to use every possible ring. *runes* is the number of runes that must be satisfied by the set of *rings*.
- The next *runes* lines describe individual runes that specify the relationships between the rings for that gate. Each rune is represented by a single line containing four numbers: r_1, r_2,

r_3, and 0, where each of these numbers are separated by a space. The first three numbers are 32-bit integers.

Output

Each gate contains exactly one riddle (consisting of multiple runes), and your algorithm should output exactly one line for each gate. If one or more runes in a riddle contain errors, output only the highest priority error for that riddle. The priority of errors is described below:

- If ANY rune in a riddle contains a statement about a null ring, e.g., 0 or -0, this is the most severe error in a riddle, and the whole riddle is invalid. Output "`INVALID: NULL RING`" as the highest priority error.
- If ANY rune in a riddle contains a statement r or $-r$ where ($r < -rings$) or ($r > rings$) then this is the SECOND most severe error in a riddle, and so output "`INVALID: RING MISSING`". NOTE: Do NOT output this message if the riddle contained a NULL ring!
- If ANY individual rune refers to the same ring more than once (e.g. `-2 2 3 0` OR `3 1 1 0`), this is the THIRD most severe error, so output "`INVALID: RUNE CONTAINS A REPEATED RING`". Again, don't output this message if a higher priority error occurred somewhere in the riddle.
- Riddles may contain repeated runes. Treat all of these repeated runes as a single rune; since they are identical, if one is true all of the repeated runes will be true.
- If there is a configuration of spinning / still rings that satisfies all the runes in the riddle, output "`RUNES SATISFIED!`"
- If there is no possible configuration of spinning / still rings that satisfies all the runes, output "`RUNES UNSATISFIABLE! TRY ANOTHER GATE!`"

Sample Input	Sample Output

<pre>5
3 5
1 2 3 0
1 -2 3 0
1 3 -2 0
-3 -1 2 0
1 2 3 0
3 8
3 1 2 0
3 -1 2 0
3 1 -2 0
3 -1 -2 0
2 1 -3 0
-2 1 -3 0
-1 2 -3 0
-1 -2 -3 0
3 2
-1 1 3 0
0 1 3 0
3 2
-1 1 3 0
7 1 3 0
3 2
-1 1 3 0
2 1 3 0</pre>
	<pre>RUNES SATISFIED!
	RUNES UNSATISFIABLE! TRY ANOTHER GATE!
	INVALID: NULL RING
	INVALID: RING MISSING
	INVALID: RUNE CONTAINS A REPEATED RING</pre>

Saruman's Level Up

Saruman's army of orcs and other dark minions continuously mine and harvest lumber out of the land surrounding his mighty tower for N continuous days. On day number i, Saruman either chooses to spend resources on mining coal and harvesting more lumber, or on raising the level (i.e., height) of his tower. He levels up his tower by one unit only on days where the binary representation of i contains a total number of 1's that is an exact multiple of 3. Assume that the initial level of his tower on day 0 is zero.

For example, Saruman will level up his tower on day 7 (binary 111), next on day 11 (binary 1011) and then day 13, day 14, day 19, and so on.

Saruman would like to forecast the level of his tower after N days. Can you write a program to help?

Input

The input file will contain multiple input test cases, each on a single line. Each test case consists of a positive integer $N < 10^{16}$, as described above. The input ends on end of file.

Output

For each test case, output one line: "`Day N: Level = L`", where `N` is the input N, and `L` is the number of levels after N days.

Sample Input	Sample Output
2 19 64	Day 2: Level = 0 Day 19: Level = 5 Day 64: Level = 21

Seating Chart

Bilbo's birthday is coming up, and Frodo and Sam are in charge of all the party planning! They have invited all the hobbits of Middle Earth to the party, and everyone will be sitting in a single row at an extremely long dining table.

However, due to poor communication, Frodo and Sam have each independently put together a seating chart for all the hobbits at the dining table. Help Frodo and Sam find out how similar their seating charts are by counting the total number of distinct pairs of hobbits who appear in different orders in the two charts.

Input

The input file will contain multiple test cases. Each test case begins with a single line containing an integer N ($1 \leq N \leq 100{,}000$) indicating the number of hobbits. The next two lines represent Frodo's and Sam's seating charts, respectively. Each seating chart is specified as a single line of N unique alphabetical strings; the set of strings in each line are guaranteed to be identical. The end-of-input is denoted by a line containing the number 0.

Output

For each input test case, output a single integer denoting, out of the N choose 2 distinct pairs of hobbits, how many pairs appear in different orders in Frodo's and Sam's seating arrangements.

Sample Input	Sample Output

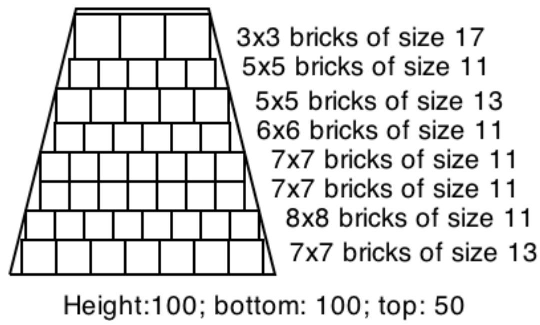
`3`
`Frodo Sam Bilbo`
`Sam Frodo Bilbo`
`5`
`A B C D E`
`B A D E C`
`0`
	`1`
	`3`

Temple Build

The Dwarves of Middle Earth are renowned for their delving and smithy ability, but they are also master builders. During the time of the dragons, the dwarves found that above ground the buildings that were most resistant to attack were truncated square pyramids (a square pyramid that does not go all the way up to a point, but instead has a flat square on top).

The dwarves knew what the ideal building shape should be based on the height they wanted and the size of the square base at the top and bottom. They typically had three different sizes of cubic bricks with which to work. Their goal was to maximize the volume of such a building based on the following rules:

The building is constructed of layers; each layer is a single square of bricks of a single size. No part of any brick may extend out from the ideal shape, either to the sides or at the top. The resulting structure will have jagged sides and may be shorter than the ideal shape, but it must fit completely within the ideal design. The picture at the right is a vertical cross section of one such tower.



There is no limit on how many bricks of each type can be used.

Input

Each line of input will contain six entries, each separated by a single space. The entries represent the ideal temple height, the size of the square base at the bottom, the size of the square base at the top (all three as non-negative integers less than or equal to one million), then three sizes of cubic bricks (all three as non-negative integers less than or equal to ten thousand). Input is terminated upon reaching end of file.

Output

For each line of input, output the maximum possible volume based on the given rules, one output per line.

Sample Input	Sample Output
`500000 800000 300000 6931 11315 5000`	`160293750000000000`

Tile Cut

When Frodo, Sam, Merry, and Pippin are at the Green Dragon Inn drinking ale, they like to play a little game with parchment and pen to decide who buys the next round. The game works as follows:

Given an $m \times n$ rectangular tile with each square marked with one of the incantations `W`, `I`, and `N`, find the maximal number of triominoes that can be cut from this tile such that the triomino has `W` and `N` on the ends and `I` in the middle (that is, it spells `WIN` in some order). Of course the only possible triominoes are the one with three squares in a straight line and the two ell-shaped ones. The Hobbit that is able to find the maximum number wins and chooses who buys the next round. Your job is to find the maximal number.

Side note: Sam and Pippin tend to buy the most rounds of ale when they play this game, so they are lobbying to change the game to Rock, Parchment, Sword (RPS)!

Input

Each input file will contain multiple test cases. Each test case consists of an $m \times n$ rectangular grid (where $1 \leq m, n \leq 30$) containing only the letters `W`, `I`, and `N`. Test cases will be separated by a blank line. Input will be terminated by end-of-file.

Output

For each input test case, print a line containing a single integer indicating the maximum total number of tiles that can be formed.

Sample Input	Sample Output
WIIW NNNN IINN WWWI NINWN INIWI WWWIW NNNNN IWINN	5 5

Tongues

Gandalf's writings have long been available for study, but no one has yet figured out what language they are written in. Recently, due to programming work by a hacker known only by the code name ROT13, it has been discovered that Gandalf used nothing but a simple letter substitution scheme, and further, that it is its own inverse—the same operation scrambles the message as unscrambles it.

This operation is performed by replacing vowels in the sequence

(a i y e o u)

with the vowel three advanced, cyclicly, while preserving case (i.e., lower or upper). Similarly, consonants are replaced from the sequence

(b k x z n h d c w g p v j q t s r l m f)

by advancing ten letters. So for instance the phrase

One ring to rule them all.

translates to

Ita dotf ni dyca nsaw ecc.

The fascinating thing about this transformation is that the resulting language yields pronounceable words.

For this problem, you will write code to translate Gandalf's manuscripts into plain text.

Input

The input file will contain multiple test cases. Each test case consists of a single line containing up to 100 characters, representing some text written by Gandalf. All characters will be plain ASCII, in the range space (32) to tilde (126), plus a newline terminating each line. The end of the input is denoted by the end-of-file.

Output

For each input test case, print its translation into plaintext. The output should contain exactly the same number of lines and characters as the input.

Sample Input	Sample Output
`Ita dotf ni dyca nsaw ecc.`	`One ring to rule them all.`