Gracias Sponsors!

Organizador

Diamond

Gold

Contest 5 - Inicial

A

Food for Animals

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

In the pet store on sale there are:

- a packs of dog food;
- b packs of cat food;
- c packs of universal food (such food is suitable for both dogs and cats).

Polycarp has x dogs and y cats. Is it possible that he will be able to buy food for all his animals in the store? Each of his dogs and each of his cats should receive one pack of suitable food for it.

Input

The first line of input contains an integer t ($1 \le t \le 10^4$) — the number of test cases in the input.

Then t lines are given, each containing a description of one test case. Each description consists of five integers a, b, c, x and y ($0 \le a, b, c, x, y \le 10^8$).

Output

For each test case in a separate line, output:

- YES, if suitable food can be bought for each of x dogs and for each of y cats;
- `NO` else.

You can output YES and `NO` in any case (for example, strings `yEs`, `yes`, `Yes` and YES will be recognized as a positive response).

Example

input

```
7
1 1 4 2 3
0 0 0 0 0
5 5 0 4 6
1 1 1 1 1
50000000 50000000 100000000 100000000 100000000
0 0 0 100000000 100000000
1 3 2 2 5
```

output

```
YES
YES
NO
YES
YES
NO
NO
```

B

Gennady and a Card Game

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Gennady owns a small hotel in the countryside where he lives a peaceful life. He loves to take long walks, watch sunsets and play cards with tourists staying in his hotel. His favorite game is called "Mau-Mau".

To play Mau-Mau, you need a pack of 52 cards. Each card has a suit (Diamonds — D, Clubs — C, Spades — S, or Hearts — H), and a rank (2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K, or A).

At the start of the game, there is one card on the table and you have five cards in your hand. You can play a card from your hand if and only if it has the same rank or the same suit as the card on the table.

In order to check if you'd be a good playing partner, Gennady has prepared a task for you. Given the card on the table and five cards in your hand, check if you can play at least one card.

Input

The first line of the input contains one string which describes the card on the table. The second line contains five strings which describe the cards in your hand.

Each string is two characters long. The first character denotes the rank and belongs to the set $\{2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K, A\}$. The second character denotes the suit and belongs to the set $\{D, C, S, H\}$.

All the cards in the input are different.

Output

If it is possible to play a card from your hand, print one word "YES". Otherwise, print "NO".

You can print each letter in any case (upper or lower).

Examples

input

```
AS
2H 4C TH JH AD
```

output

```
YES
```

input

```
2H
3D 4C AC KD AS
```

output

```
NO
```

input

```
4D
AS AC AD AH 5H
```

output

```
YES
```

Note

In the first example, there is an Ace of Spades (`AS`) on the table. You can play an Ace of Diamonds (`AD`) because both of them are Aces.

In the second example, you cannot play any card.

In the third example, you can play an Ace of Diamonds (`AD`) because it has the same suit as a Four of Diamonds (`4D`), which lies on the table.

C

Rock Is Push

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are at the top left cell $(1, 1)$ of an $n \times m$ labyrinth. Your goal is to get to the bottom right cell (n, m). You can only move right or down, one cell per step. Moving right from a cell (x, y) takes you to the cell $(x, y + 1)$, while moving down takes you to the cell $(x + 1, y)$.

Some cells of the labyrinth contain rocks. When you move to a cell with rock, the rock is pushed to the next cell in the direction you're moving. If the next cell contains a rock, it gets pushed further, and so on.

The labyrinth is surrounded by impenetrable walls, thus any move that would put you or any rock outside of the labyrinth is illegal.

Count the number of different legal paths you can take from the start to the goal modulo $10^9 + 7$. Two paths are considered different if there is at least one cell that is visited in one path, but not visited in the other.

Input

The first line contains two integers n, m — dimensions of the labyrinth ($1 \leq n, m \leq 2000$).

Next n lines describe the labyrinth. Each of these lines contains m characters. The j-th character of the i-th of these lines is equal to "R" if the cell (i, j) contains a rock, or "." if the cell (i, j) is empty.

It is guaranteed that the starting cell $(1, 1)$ is empty.

Output

Print a single integer — the number of different legal paths from $(1, 1)$ to (n, m) modulo $10^9 + 7$.

Examples

input Copy

```
1 1
.
```

output Copy

```
1
```

input Copy

```
2 3
...
..R
```

output Copy

```
0
```

input Copy

```
4 4
...R
.RR.
.RR.
R...
```

output

Copy

```
4
```

Note

In the first sample case we can't (and don't have to) move, hence the only path consists of a single cell $(1, 1)$.

In the second sample case the goal is blocked and is unreachable.

Illustrations for the third sample case can be found here: https://assets.codeforces.com/rounds/1225/index.html

D Dungeon

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are playing a new computer game in which you have to fight monsters. In a dungeon you are trying to clear, you met three monsters; the first of them has a health points, the second has b health points, and the third has c.

To kill the monsters, you can use a cannon that, when fired, deals 1 damage to the selected monster. Every 7-th (i. e. shots with numbers 7, 14, 21 etc.) cannon shot is *enhanced* and deals 1 damage to **all** monsters, not just one of them. If some monster's current amount of health points is 0, it can't be targeted by a regular shot and does not receive damage from an *enhanced* shot.

You want to pass the dungeon beautifully, i. e., kill all the monsters with the same *enhanced* shot (i. e. after some *enhanced* shot, the health points of each of the monsters should become equal to 0 **for the first time**). Each shot must hit a monster, i. e. each shot deals damage to at least one monster.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

Each test case consists of a single line that contains three integers a, b and c ($1 \le a, b, c \le 10^8$) — the number of health points each monster has.

Output

For each test case, print YES if you can kill all the monsters with the same *enhanced* shot. Otherwise, print `NO`. You may print each letter in any case (for example, `YES`, `Yes`, `yes`, `yEs` will all be recognized as positive answer).

Example

input

```
3
3 2 4
1 1 1
10 1 7
```

output

```
YES
NO
NO
```

Note

In the first test case, you can do as follows: 1-th shot to the first monster, 2-th shot to the second monster, 3-th shot to the third monster, 4-th shot to the first monster, 5-th shot to the third monster, 6-th shot to the third monster, and 7-th *enhanced* shot will kill all the monsters.

In the second test case, you can't kill monsters with the same *enhanced* shot, because the total number of health points of monsters is 3, and you will kill them in the first 3 shots.

E

0-1 MST

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Ujan has a lot of useless stuff in his drawers, a considerable part of which are his math notebooks: it is time to sort them out. This time he found an old dusty graph theory notebook with a description of a graph.

It is an undirected weighted graph on n vertices. It is a complete graph: each pair of vertices is connected by an edge. The weight of each edge is either 0 or 1; exactly m edges have weight 1, and all others have weight 0.

Since Ujan doesn't really want to organize his notes, he decided to find the weight of the minimum spanning tree of the graph. (The weight of a spanning tree is the sum of all its edges.) Can you find the answer for Ujan so he stops procrastinating?

Input

The first line of the input contains two integers n and m ($1 \leq n \leq 10^5$, $0 \leq m \leq \min(\frac{n(n-1)}{2}, 10^5)$), the number of vertices and the number of edges of weight 1 in the graph.

The i-th of the next m lines contains two integers a_i and b_i ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$), the endpoints of the i-th edge of weight 1.

It is guaranteed that no edge appears twice in the input.

Output

Output a single integer, the weight of the minimum spanning tree of the graph.

Examples

input

```
6 11
1 3
1 4
1 5
1 6
2 3
2 4
2 5
2 6
3 4
3 5
3 6
```

output

```
2
```

input

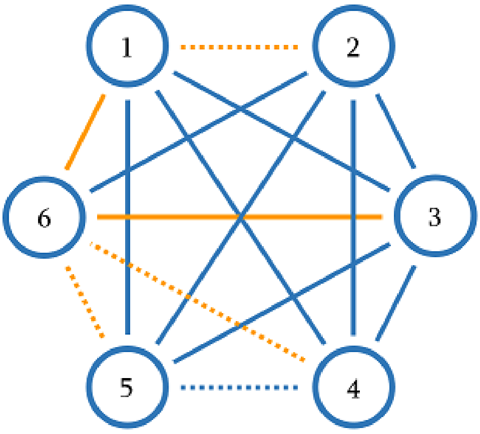
```
3 0
```

output

```
0
```

Note

The graph from the first sample is shown below. Dashed edges have weight 0, other edges have weight 1. One of the minimum spanning trees is highlighted in orange and has total weight 2.



In the second sample, all edges have weight 0 so any spanning tree has total weight 0.

F

Multi-Subject Competition

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A multi-subject competition is coming! The competition has m different subjects participants can choose from. That's why Alex (the coach) should form a competition delegation among his students.

He has n candidates. For the i-th person he knows subject s_i the candidate specializes in and r_i — a skill level in his specialization (this level can be negative!).

The rules of the competition require each delegation to choose some subset of subjects they will participate in. The only restriction is that the **number of students from the team** participating in each of the **chosen** subjects should be the **same**.

Alex decided that each candidate would participate only in the subject he specializes in. Now Alex wonders whom he has to choose to maximize the total sum of skill levels of all delegates, or just skip the competition this year if every valid non-empty delegation has negative sum.

(Of course, Alex doesn't have any spare money so each delegate he chooses must participate in the competition).

Input

The first line contains two integers n and m ($1 \le n \le 10^5$, $1 \le m \le 10^5$) — the number of candidates and the number of subjects.

The next n lines contains two integers per line: s_i and r_i ($1 \le s_i \le m$, $-10^4 \le r_i \le 10^4$) — the subject of specialization and the skill level of the i-th candidate.

Output

Print the single integer — the maximum total sum of skills of delegates who form a valid delegation (according to rules above) or 0 if every valid non-empty delegation has negative sum.

Examples

input

```
6 3
2 6
3 6
2 5
3 5
1 9
3 1
```

output

```
22
```

input

```
5 3
2 6
3 6
2 5
3 5
1 11
```

output

```
23
```

input

```
5 2
1 -1
1 -5
2 -1
2 -1
1 -10
```

output

```
0
```

Note

In the first example it's optimal to choose candidates 1, 2, 3, 4, so two of them specialize in the 2-nd subject and other two in the 3-rd. The total sum is $6 + 6 + 5 + 5 = 22$.

In the second example it's optimal to choose candidates 1, 2 and 5. One person in each subject and the total sum is $6 + 6 + 11 = 23$.

In the third example it's impossible to obtain a non-negative sum.

G. Nirvana

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Kurt reaches nirvana when he finds the product of all the digits of some positive integer. Greater value of the product makes the nirvana deeper.

Help Kurt find the maximum possible product of digits among all integers from 1 to n.

Input

The only input line contains the integer n ($1 \le n \le 2 \cdot 10^9$).

Output

Print the maximum product of digits among all integers from 1 to n.

Examples

input

```
390
```

output

```
216
```

input

```
7
```

output

```
7
```

input

```
1000000000
```

output

```
387420489
```

Note

In the first example the maximum product is achieved for 389 (the product of digits is $3 \cdot 8 \cdot 9 = 216$).

In the second example the maximum product is achieved for 7 (the product of digits is 7).

In the third example the maximum product is achieved for 999999999 (the product of digits is $9^9 = 387420489$).

H

AB Balance

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a string s of length n consisting of characters a and/or b.

Let $\mathrm{AB}(s)$ be the number of occurrences of string ab in s as a **substring**. Analogically, $\mathrm{BA}(s)$ is the number of occurrences of ba in s as a **substring**.

In one step, you can choose any index i and replace s_i with character a or b.

What is the minimum number of steps you need to make to achieve $\mathrm{AB}(s) = \mathrm{BA}(s)$?

Reminder:

The number of occurrences of string d in s as substring is the number of indices i ($1 \le i \le |s| - |d| + 1$) such that substring $s_i s_{i+1} \dots s_{i+|d|-1}$ is equal to d. For example, $\mathrm{AB}($aabbbabaa$) = 2$ since there are two indices i: $i = 2$ where a<u>ab</u>bbabaa and $i = 6$ where aabbb<u>ab</u>aa.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$). Description of the test cases follows.

The first and only line of each test case contains a single string s ($1 \le |s| \le 100$, where $|s|$ is the length of the string s), consisting only of characters a and/or b.

Output

For each test case, print the resulting string s with $\mathrm{AB}(s) = \mathrm{BA}(s)$ you'll get making the minimum number of steps.

If there are multiple answers, print any of them.

Example

input

```
4
b
aabbbabaa
abbb
abbaab
```

output

```
b
aabbbabaa
bbbb
abbaaa
```

Note

In the first test case, both $\mathrm{AB}(s) = 0$ and $\mathrm{BA}(s) = 0$ (there are no occurrences of ab (ba) in b), so can leave s untouched.

In the second test case, $\mathrm{AB}(s) = 2$ and $\mathrm{BA}(s) = 2$, so you can leave s untouched.

In the third test case, $\mathrm{AB}(s) = 1$ and $\mathrm{BA}(s) = 0$. For example, we can change s_1 to b and make both values zero.

In the fourth test case, $\mathrm{AB}(s) = 2$ and $\mathrm{BA}(s) = 1$. For example, we can change s_6 to a and make both values equal to 1.

I

Balanced Playlist

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Your favorite music streaming platform has formed a *perfectly balanced* playlist exclusively for you. The playlist consists of n tracks numbered from 1 to n. The playlist is automatic and cyclic: whenever track i finishes playing, track $i+1$ starts playing automatically; after track n goes track 1.

For each track i, you have estimated its *coolness* a_i. The higher a_i is, the cooler track i is.

Every morning, you choose a track. The playlist then starts playing from this track in its usual cyclic fashion. At any moment, you remember the maximum coolness x of already played tracks. Once you hear that a track with coolness **strictly** less than $\frac{x}{2}$ (no rounding) starts playing, you turn off the music immediately to keep yourself in a good mood.

For each track i, find out how many tracks you will listen to before turning off the music if you start your morning with track i, or determine that you will never turn the music off. Note that if you listen to the same track several times, every time must be counted.

Input

The first line contains a single integer n ($2 \le n \le 10^5$), denoting the number of tracks in the playlist.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), denoting coolnesses of the tracks.

Output

Output n integers $c_1, c_2, \ldots, c_n$, where c_i is either the number of tracks you will listen to if you start listening from track i or -1 if you will be listening to music indefinitely.

Examples

input

```
4
11 5 2 7
```

output

```
1 1 3 2
```

input

```
4
3 2 5 3
```

output

```
5 4 3 6
```

input

```
3
4 3 6
```

output

```
-1 -1 -1
```

Note

In the first example, here is what will happen if you start with...

- track 1: listen to track 1, stop as $a_2 < \frac{a_1}{2}$.
- track 2: listen to track 2, stop as $a_3 < \frac{a_2}{2}$.
- track 3: listen to track 3, listen to track 4, listen to track 1, stop as $a_2 < \frac{\max(a_3, a_4, a_1)}{2}$.
- track 4: listen to track 4, listen to track 1, stop as $a_2 < \frac{\max(a_4, a_1)}{2}$.

In the second example, if you start with track 4, you will listen to track 4, listen to track 1, listen to track 2, listen to track 3, listen to track 4 again, listen to track 1 again, and stop as $a_2 < \frac{max(a_4, a_1, a_2, a_3, a_4, a_1)}{2}$. Note that both track 1 and track 4 are counted twice towards the result.

J

WOW Factor

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Recall that string a is a subsequence of a string b if a can be obtained from b by deletion of several (possibly zero or all) characters. For example, for the string a="wowwo", the following strings are subsequences: "wowwo", "wowo", "oo", "wow", "", and others, but the following are not subsequences: "owoo", "owwwo", "ooo".

The *wow factor* of a string is the number of its subsequences equal to the word "wow". Bob wants to write a string that has a large *wow factor*. However, the "w" key on his keyboard is broken, so he types two "v"s instead.

Little did he realise that he may have introduced more "w"s than he thought. Consider for instance the string "ww". Bob would type it as "vvvv", but this string actually contains three occurrences of "w":

- "**vv**vv"
- "v**vv**v"
- "vv**vv**"

For example, the *wow factor* of the word "vvvovvv" equals to four because there are four *wow*s:

- "**vv**v**ovv**v"
- "**vv**v**o**v**vv**"
- "v**vvovv**v"
- "v**vvo**v**vv**"

Note that the subsequence "**v**v**vovv**v" does not count towards the *wow factor*, as the "v"s have to be consecutive.

For a given string s, compute and output its *wow factor*. Note that it is **not** guaranteed that it is possible to get s from another string replacing "w" with "vv". For example, s can be equal to "vov".

Input

The input contains a single non-empty string s, consisting only of characters "v" and "o". The length of s is at most 10^6.

Output

Output a single integer, the *wow factor* of s.

Examples

input Copy

```
vvvovvv
```

output Copy

```
4
```

input Copy

```
vvovooovovvovoovoovvvvovovvvov
```

output

Copy

```
100
```

Note

The first example is explained in the legend.

K Tree Diameter

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is a weighted tree with n nodes and $n-1$ edges. The nodes are conveniently labeled from 1 to n. The weights are positive integers at most 100. Define the distance between two nodes to be the sum of edges on the unique path between the nodes. You would like to find the diameter of the tree. Diameter is the maximum distance between a pair of nodes.

Unfortunately, the tree isn't given to you, but you can ask some questions about it. In one question, you can specify two nonempty disjoint sets of nodes p and q, and the judge will return the maximum distance between a node in p and a node in q. In the words, maximum distance between x and y, where $x \in p$ and $y \in q$. After asking not more than 9 questions, you must report the maximum distance between any pair of nodes.

Interaction

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1\,000$). Description of the test cases follows.

The first line of each test case contains an integer n ($2 \le n \le 100$) — the number of nodes in the tree.

To ask a question, print "$k_1\ k_2\ a_1\ a_2\ \ldots\ a_{k_1}\ b_1\ b_2\ \ldots\ b_{k_2}$" ($k_1, k_2 \ge 1$ and $k_1 + k_2 \le n$). These two sets must be nonempty and disjoint. The judge will respond with a single integer $\max_{p,q} dist(a_p, b_q)$. If you ever get a result of -1 (because you printed an invalid query), exit immediately to avoid getting other verdicts.

After printing a query do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

When you are ready to answer, print "$-1\ d$", where d is the maximum shortest distance over all pairs of nodes.

You can only ask at most 9 questions per test case.

Hack Format

To hack, use the following format. Note that you can only hack with one test case.

The first line should contain a single integer t ($t = 1$).

The second line should contain a single integer n ($2 \le n \le 100$).

Each of the next $n-1$ lines should contain three integers a_i, b_i, c_i ($1 \le a_i, b_i \le n$, $1 \le c_i \le 100$). This denotes an undirected edge between nodes a_i and b_i with weight c_i. These edges must form a tree.

Example

input

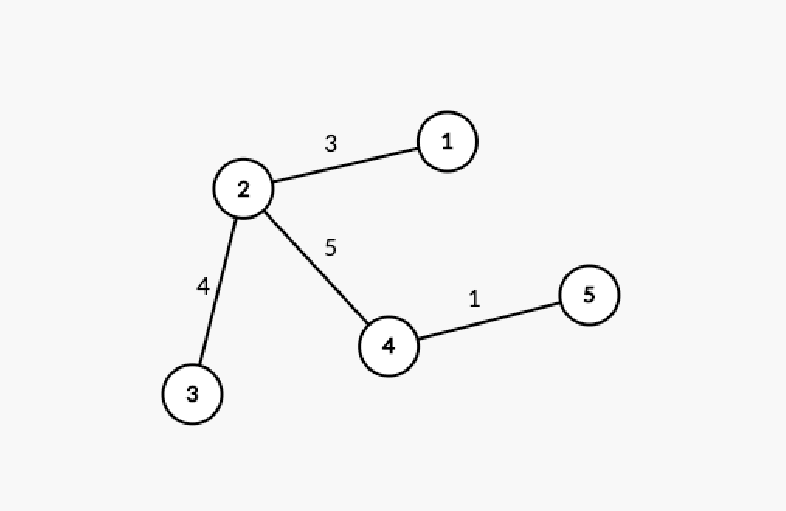
```
2
5
9
6
10
9
10
2
99
```

output

```
1 4 1 2 3 4 5
1 4 2 3 4 5 1
1 4 3 4 5 1 2
1 4 4 5 1 2 3
1 4 5 1 2 3 4
-1 10
1 1 1 2
-1 99
```

Note

In the first example, the first tree looks as follows:



In the first question, we have $p = 1$, and $q = 2, 3, 4, 5$. The maximum distance between a node in p and a node in q is 9 (the distance between nodes 1 and 5).

The second tree is a tree with two nodes with an edge with weight 99 between them.

L

Balanced Rating Changes

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Another Codeforces Round has just finished! It has gathered n participants, and according to the results, the expected rating change of participant i is a_i. These rating changes are *perfectly balanced* — their sum is equal to 0.

Unfortunately, due to minor technical glitches, the round is declared *semi-rated*. It means that all rating changes must be divided by two.

There are two conditions though:

- For each participant i, their modified rating change b_i must be integer, and as close to $\frac{a_i}{2}$ as possible. It means that either $b_i = \lfloor \frac{a_i}{2} \rfloor$ or $b_i = \lceil \frac{a_i}{2} \rceil$. In particular, if a_i is even, $b_i = \frac{a_i}{2}$. Here $\lfloor x \rfloor$ denotes rounding down to the largest integer not greater than x, and $\lceil x \rceil$ denotes rounding up to the smallest integer not smaller than x.
- The modified rating changes must be perfectly balanced — their sum must be equal to 0.

Can you help with that?

Input

The first line contains a single integer n ($2 \le n \le 13\,845$), denoting the number of participants.

Each of the next n lines contains a single integer a_i ($-336 \le a_i \le 1164$), denoting the rating change of the i-th participant.

The sum of all a_i is equal to 0.

Output

Output n integers b_i, each denoting the modified rating change of the i-th participant in order of input.

For any i, it must be true that either $b_i = \lfloor \frac{a_i}{2} \rfloor$ or $b_i = \lceil \frac{a_i}{2} \rceil$. The sum of all b_i must be equal to 0.

If there are multiple solutions, print any. We can show that a solution exists for any valid input.

Examples

input Copy

```
3
10
-5
-5
```

output Copy

```
5
-2
-3
```

input Copy

```
7
-7
-29
0
```

```
3
24
-29
38
```

output

Copy

```
-3
-15
0
2
12
-15
19
```

Note

In the first example, $b_1 = 5$, $b_2 = -3$ and $b_3 = -2$ is another correct solution.

In the second example there are 6 possible solutions, one of them is shown in the example output.

M

Squares and Cubes

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarp likes squares and cubes of positive integers. Here is the beginning of the sequence of numbers he likes: 1, 4, 8, 9, ...

For a given number n, count the number of integers from 1 to n that Polycarp likes. In other words, find the number of such x that x is a square of a positive integer number or a cube of a positive integer number (or both a square and a cube simultaneously).

Input

The first line contains an integer t ($1 \le t \le 20$) — the number of test cases.

Then t lines contain the test cases, one per line. Each of the lines contains one integer n ($1 \le n \le 10^9$).

Output

For each test case, print the answer you are looking for — the number of integers from 1 to n that Polycarp likes.

Example

input Copy

```
6
10
1
25
1000000000
999999999
500000000
```

output Copy

```
4
1
6
32591
32590
23125
```