Pacific Northwest Region Programming Contest Division 2

November 15th, 2014

Reminders

- For all problems, read the input data from standard input and write the results to standard output.
- In general, when there is more than one integer or word on an input line, they will be separated from each other by exactly one space. No input lines will have leading or trailing spaces, and tabs will never appear in any input.
- Platform is as follows:

```
Ubuntu 14.04.1 LTS x86_64
geany
java version 1.7.0_65
c/c++ gcc version 4.8.2
eclipse 4.4 with CDT 8.4
Python 2.7.6 (IDE support)
Python 3.4.0 (syntax highlighting editor support)
```

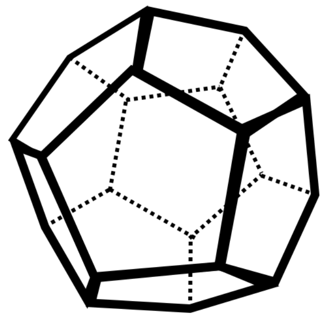
- Compiler options are as follows:

```
g++ -g -O2 -std=gnu++0x -static $*
gcc -g -O2 -std=gnu99 -static $* -lm
javac -encoding UTF-8 -sourcepath . -d . $* runjava
java -client -Xss8m -Xmx1024m $*
python $*
```

- Python may not have sufficient performance for many of the problems; use it at your discretion.

Problem M — limit 5 seconds

Polyhedra



Given a sphere, you can slice through the surface of the sphere to make different convex polyhedra. All of these convex polyhedra have the Euler characteristic which can be defined as follows:

$$x = V - E + F = 2$$

where V represents the number of vertices, E the number of edges and F the number of faces on a convex polyhedron.

Input

Input begins with a line with a single integer T, $1 \leq T \leq 100$, denoting the number of test cases. Each test case consists of a single line with two space-separated integers V and E ($4 \leq V, E \leq 100$), representing the number of vertices and edges respectively of the convex polyhedron.

Output

For each test case, print on a single line the number of faces in the defined polyhedron.

Sample Input	Sample Output
2 8 12 4 6	6 4

Problem N — limit 5 seconds

Majority

The votes are in! Mathematicians world-wide have been polled, and each has chosen their favorite number between 1 and 1000. Your goal is to tally the votes and determine what the most popular number is.

If there is a tie for the greatest number of votes, choose the smallest number with that many votes.

Input

Input will start with a single line containing the number of test cases, between 1 and 100 inclusive. For each test case, there will be a single line giving the number of votes V, $1 \leq V \leq 1000$. Following that line will be V lines, each with a single integer vote between 1 and 1000.

Output

Sample Input

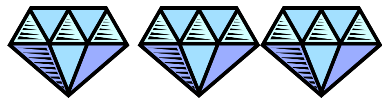
```
3
3
42
42
19
4
7
99
99
7
5
11
12
13
14
15
```

Sample Output

```
42
7
11
```

PROBLEM O — LIMIT 5 SECONDS

Diamonds



A diamond's overall worth is determined by its mass in carats as well as its overall clarity. A large diamond with many imperfections is not worth as much as a smaller, flawless diamond. The overall clarity of a diamond can be described on a scale from 0.0–10.0 adopted by the American Gem Society, where 0.0 represents a flawless diamond and 10.0 represents an imperfect diamond.

Given a sequence of N diamonds, each with weight, w_i, in carats and clarity, c_i, on the scale described above, find the longest subsequence of diamonds for which the weight and clarity are both becoming strictly more favorable to a buyer.

Example

In the following sequence of diamonds,

w_i	c_i
1.5	9.0
2.0	2.0
2.5	6.0
3.0	5.0
4.0	2.0
10.0	5.5

the longest desirable subsequence is

1.5	9.0
2.5	6.0
3.0	5.0
4.0	2.0

because the weights strictly increase while the clarities strictly decrease.

Input

Input begins with a line with a single integer T, $1 \leq T \leq 100$, indicating the number of test cases. Each test case begins with a line with a single integer N, $1 \leq N \leq 200$, indicating the number of diamonds. Next follow N lines with 2 real numbers w_i and c_i, $0.0 \leq w_i, c_i \leq 10.0$, indicating the weight in carats and the clarity of diamond i, respectively.

Output

For each test case, output a single line with the length of the longest desirable subsequence of diamonds.

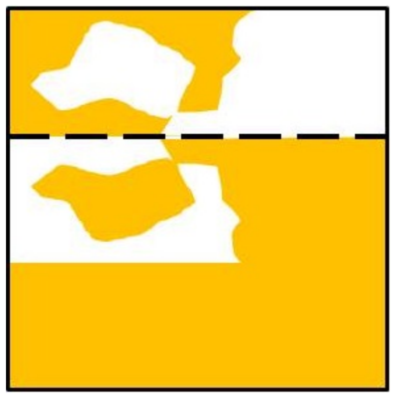
Sample Input	Sample Output

<pre>3
2
1.0 1.0
1.5 0.0
3
1.0 1.0
1.0 1.0
1.0 1.0
6
1.5 9.0
2.0 2.0
2.5 6.0
3.0 5.0
4.0 2.0
10.0 5.5</pre>
	<pre>2
	1
	4</pre>

Problem P — limit 5 seconds

Gold Leaf

Gold leaf is a very thin layer of gold with a paper backing. If the paper gets folded and then unfolded, the gold leaf will stick to itself more readily than it will stick to the paper, so there will be patches of gold and patches of exposed paper. Note that the gold leaf will always stick to itself, rather than the paper. In the following example, the paper was folded along the dashed line. Notice how the gold leaf always sticks to one side or the other, never both.



Consider a crude digital image of a sheet of gold leaf. If the area covered by a pixel is mostly gold, that will be represented by a '#'. If it's mostly exposed paper, it will be represented by a '.'. Determine where the sheet was folded. The sheet was folded exactly once, along a horizontal, vertical, or 45 degree diagonal line. If the fold is horizontal or vertical, it is always between rows/columns. If the fold is diagonal, then the fold goes through a diagonal line of cells, and the cells along the fold are always '#'.

Input

Input will start with a single line containing the number of cases, between 1 and 100, inclusive. Each test case will begin with a line with two integers, N and M $2 \leq N, M \leq 25$, where N is the number of rows, and M is the number of columns of the photograph. Each of the next N lines will contain exactly M characters, all of which will be either '#' or '.'. This represents a crudely represented digital image of the sheet of gold leaf. There is guaranteed to be at least one '.', and there is guaranteed to be a solution.

Output

For each test case, output four integers, indicating the places where the fold hits the edges of the paper. Output them in the order r1 c1 r2 c2 where (r1,c1) and (r2,c2) are row/column coordinates (r=row, c=column). The top left character is (1,1) and the bottom right is (n,m). If the fold is horizontal or diagonal, list the left side coordinates before the right. If the fold is vertical, list the top coordinates before the bottom. If the fold is horizontal, use the coordinates above the fold. If the fold is vertical, use the coordinates to the left of the fold. If the fold is diagonal, use the coordinates of the cells that the fold goes through. If more than one fold is possible, choose the one with the smallest first coordinate, then the smallest second coordinate, then third, then fourth.

Sample Input	Sample Output
3 8 10 #.#..##..# ####..#### ###.##.... ...#..#### ....##.... .#.##..##. ########## ########## 5 20 ###########.#.#.#.#. ###########...#.###. ##########..##.#..## ###########..#.#.##. ###########.###...#. 5 5 .#### ###.# ##..# #..## #####	3 1 3 10 1 15 5 15 4 1 1 4

Problem Q — limit 5 seconds

Number Game

Alice and Bob are playing a game on a line of N squares. The line is initially populated with one of each of the numbers from 1 to N. Alice and Bob take turns removing a single number from the line, subject to the restriction that a number may only be removed if it is not bordered by a higher number on either side. When the number is removed, the square that contained it is now empty. The winner is the player who removes the 1 from the line. Given an initial configuration, who will win, assuming Alice goes first and both of them play optimally?

Input

Input begins with a line with a single integer T, $1 \leq T \leq 100$, denoting the number of test cases. Each test case begins with a line with a single integer N, $1 \leq N \leq 100$, denoting the size of the line. Next is a line with the numbers from 1 to N, space separated, giving the numbers in line order from left to right.

Output

For each test case, print the name of the winning player on a single line.

<table>
<tr><th>Sample Input</th><th>Sample Output</th></tr>
<tr><td>4
4
2 1 3 4
4
1 3 2 4
3
1 3 2
6
2 5 1 6 4 3</td><td>Bob
Alice
Bob
Alice</td></tr>
</table>

PROBLEM R — LIMIT 5 SECONDS

Ramp Number

A Ramp Number is a number whose digits only rise or stay the same; they never fall.

- 123 is a ramp number.
- 101 is not a ramp number.
- 1111000001111 is not a ramp number.

Given a positive integer n, if it is a ramp number, print the number of ramp numbers less than it. If it is not a ramp number, print `-1`.

Input

Input will start with a single line giving the number of test cases. Each test case will be a single positive integer on a single line, with up to 80 digits. The result will always fit into a 64-bit long.

Output

For each test case, print `-1` if the input is not a ramp number. Print the number of ramp numbers less than the input value if the input value is a ramp number.

Sample Input	Sample Output

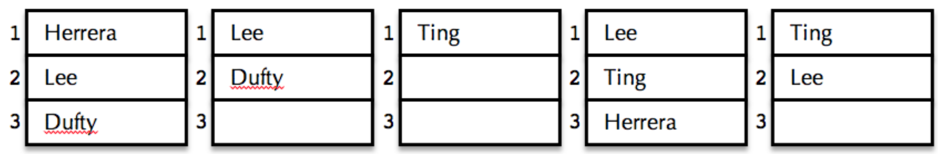
<pre>5
11
123
101
1111
99999</pre>
	<pre>10
	65
	-1
	220
	2001</pre>

Problem S — limit 60 seconds

Ranked Choice

Ranked choice voting—also known as instant runoff voting—is used in San Francisco and Oakland for mayoral elections. Rather than voting for a single candidate, those casting ballots vote for up to three candidates, ranking them 1, 2, and 3.

The first five of what might be tens or even hundreds of thousands of ballots in a real election might look like this:



Initially, only first place votes matter, and if a single candidate gets the clear majority of all first place votes, then that candidate wins. Rarely would anyone get a majority of all first place votes (there were, for instance, 16 official candidates in San Francisco's mayoral election on November 8th, 2011, and Ed Lee, who eventually won, only got 31% of the first choice votes.) In that case, the candidates with the least number of first place votes are eliminated by effectively removing them from all ballots everywhere and promoting all second and third place votes to be first and second place votes to close out any gaps.

If, for example, after an analysis of all ballots, it's determined that Phil Ting received the smallest number of first place votes, the ballots would be updated to look like this:

The first two ballots were left alone, but the next three were updated to reflect Phil Ting's elimination. The one ballot including a standalone vote for Phil Ting was removed, since it no longer contains any valid votes. The two other impacted ballots shown each show candidates Dennis Herrera and Ed Lee advance from third to second and second to first, respectively.

The process is repeated over and over again until it leaves one candidate with a majority of first choice votes. (On November 8th, 2011, this process was applied 12 times before Ed Lee prevailed with 61% of all remaining rank-one votes and was declared the winner of the San Francisco mayoral race.) If the final round sees a two-way tie between the two remaining candidates [or a three-way tie between the three remaining candidates, and so forth], then no winner is declared.

Input Format

The first line contains an integer between 1 and 100, inclusive, giving the number of test cases. Each test case leads with a single integer between 1 and 100000, which specifies how many ballots need to be processed. Each ballot is expressed as a string of capital letters, where each letter represents a single candidate. Ballot string can be of length 1, 2, 3 to denote up to three votes, and you can assume no ballot ever contains more than one vote for any particular candidate. The first character of the string reflects the voter's first choice. The second character, if present, represents the voter's second choice. The third character, if present, represents the voter's third choice.

Output Format

For each input scenario, publish a single line summarizing exactly how candidates were eliminated. If a winner can be declared immediately (i.e. a candidate has a clear majority without eliminating anyone), then simply print the capital letter for that candidate on its own line, as with:

```
A
```

If one or more elimination rounds are needed, then a summary of the elimination process should look like this:

```
EF -> D -> C -> B
```

The above line reflects the fact that three elimination rounds were needed to arrive at a winner. The above line conveys the understanding that candidates E and F were eliminated during the first round, candidate D was eliminated during a second round, and candidate C was eliminated during a third round before B was declared the winner. When multiple candidates are eliminated in the same round, they must be listed in alphabetic order. [There is exactly one space between visible characters, and there are no trailing spaces at the end of any line.]

It's possible a series of elimination rounds leaves the election in a stalemate, because all remaining candidates have an equal number of first-choice votes. In that case, all remaining candidates incidentally tie for the least number of first-choice votes and are collectively eliminated, leaving no candidates and no clear winner. In that case, the summary would look as follows:

```
DE -> A -> BCF -> no winner
```

The above conveys that two elimination rounds led to a three-way tie between candidates B, C, and F, all of whom were eliminated in a third round that left all ballots empty.

Sample Input	Sample Output
7 1 E 1 ABC 8 AB AF AC AG AH AD AE AI 6 AB AB AB CB CB CB 3 AC BC C 8 ACE CBA DBA B ABD ABD BAE CEF 5 BAC A CAB C BDE	E A A B -> AC -> no winner ABC -> no winner EF -> D -> C -> B DE -> A -> BC -> no winner

Problem T — limit 5 seconds

Runes

You are helping an archaeologist decipher some runes. He knows that this ancient society used a Base 10 system, and that they never start a number with a leading zero. He's figured out most of the digits as well as a few operators, but he needs your help to figure out the rest.

The professor will give you a simple math expression. He has converted all of the runes he knows into digits. The only operators he knows are addition (`+`), subtraction (`-`), and multiplication (`*`), so those are the only ones that will appear. Each number will be in the range from $-999,999$ to $999,999$, and will consist of only the digits '`0`'–'`9`', possibly a leading '`-`', and a few '`?`'s. The '`?`'s represent a digit rune that the professor doesn't know (never an operator, an '`=`', or a leading '`-`'). All of the '`?`'s in an expression will represent the same digit (0–9), and it won't be one of the other given digits in the expression.

Given an expression, figure out the value of the rune represented by the question mark. If more than one digit works, give the lowest one. If no digit works, well, that's bad news for the professor—it means that he's got some of his runes wrong. Output -1 in that case.

Input

The sample data will start with the number of test cases T ($1 \le T \le 100$). Each test case will consist of a single line, of the form:

[number][op][number]=[number]

Each [number] will consist of only the digits '`0`'-'`9`', with possibly a single leading minus '`-`', and possibly some '`?`'s. No number will begin with a leading '`0`' unless it is 0, no number will begin with `-0`, and no number will have more than 6 characters (digits or `?`s). The [op] will separate the first and second [number]s, and will be one of: `+`, `-` or `*`. The `=` will always be present between the second and third [number]s. There will be no spaces, tabs, or other characters. There is guaranteed to be at least one `?` in every equation.

Output

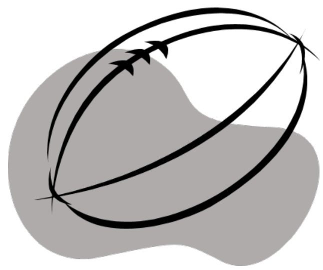
Output the lowest digit that will make the equation work when substituted for the `?`s, or output -1 if no digit will work. Output no extra spaces or blank lines.

Sample Input	Sample Output

`5`
`1+1=?`
`123*45?=5?088`
`-5?*-1=5?`
`19--45=5?`
`??*??=302?`
	`2`
	`6`
	`0`
	`-1`
	`5`

PROBLEM U — LIMIT 10 SECONDS

Top 25



In college football, many different sources create a list of the Top 25 teams in the country. Since it's subjective, these lists often differ, but they're usually very similar. Your job is to compare two of these lists, and determine where they are similar. In particular, you are to partition them into sets, where each set represents the same consecutive range of positions in both lists, and has the same teams, and is as small as possible. If the lists agree completely, you'll have 25 lists (or n, where n is an input). For these lists:

K&R Poll	Lovelace Ranking
A	A
B	C
C	D
D	B
E	E

You'll have 3 sets:

A

B C D

E

Input

The input will start with a single integer on one line giving the number of test cases. There will be at least one but not more than 100 test cases. Each test case will begin with an integer N, $1 \leq N \leq 1{,}000{,}000$, indicating the number of teams ranked. The next N lines will hold the first list, in order. The team names will appear one per line, consist of at most 8 capital letters only. After this will be N lines, in the same format, indicating the second list. Both lists will contain the same team names, and all N team names will be unique.

Output

For each test case, simply output the size of each set, in order, on one line, with the numbers separated by a single space. Do not output any extra spaces, and do not output blank lines between numbers.

Sample Input	Sample Output
3 5 A B C D E A C D B E 3 RED BLUE ORANGE RED BLUE ORANGE 3 MOE LARRY CURLY CURLY MOE LARRY	1 3 1 1 1 1 3

Problem V — limit 5 seconds

Towers

The Towers puzzle challenges a single player to place towers of varying heights in an n x n grid ($3 \leq n \leq 5$). The heights of each tower can be any integer between 1 and n, inclusive, but the placement of the n^2 towers must be such that no tower of the same height appears twice within the same row or column. Given no other constraints, there is an exponentially large number of ways to place towers. Take, for example, the 5 x 5 puzzle, where one of the solutions looks like this:

1	2	3	4	5
2	1	4	5	3
3	4	5	1	2
4	5	2	3	1
5	3	1	2	4

The puzzles become more interesting (and harder to solve) as they further constrain that one or more grid locations be occupied by towers of specified heights. A 5 x 5 puzzle, for instance, might require that the upper left and lower right corners house towers of height 3, and that the center location house a tower of height 5. The puzzle would look like that on the left below, and a solution—again, one of many—might look like that presented to its right.

3				
		5		
				3

3	1	2	4	5
4	3	1	5	2
2	4	5	3	1
1	5	3	2	4
5	2	4	1	3

Some puzzles include one or more numbers around the perimeter, where each number specifies the exact number of towers visible when looking into the grid from that direction, with the understanding that taller towers fully conceal shorter ones. For example, the 5 x 5 puzzle presented below and on the left would have the (incidentally unique) solution to its right.

					1	
2			3			3
2						2
	1					
				1		2

					1	
	3	1	4	2	5	
2	2	5	3	4	1	3
2	4	2	1	5	3	2
	1	4	5	3	2	
	5	3	2	1	4	2

The number above the final column requires that just a single tower be visible when viewing its sequence of five towers from above (which essentially means that the top row of that column must house a 5.) The second row introduces multiple requirements:

- the center column must house a 3,
- exactly two towers are visible when viewing its tower sequence from the left, and
- exactly three towers are visible when viewing its two sequence from the right.

Input

The input starts with a single integer on a line by itself, giving the number of tests; there will be at least 1 but no more than 100 test cases. Each $n \times n$ puzzle ($3 \leq n \leq 5$) is expressed as a series of $n + 2$ lines of length $n + 2$. The outer perimeter of the grid specifies the visibility constraints (where '`-`' expresses there are no constraints for that row or column from the relevant direction looking in; the corners of the perimeter are always '`-`'] and the interior of the grid specifies those locations where a tower of a specific height must be placed (where '`-`' expresses there is no imposed tower height for that location.)

We guarantee that every character in the grid is either a '`-`' or a digit between 1 and n.

Output

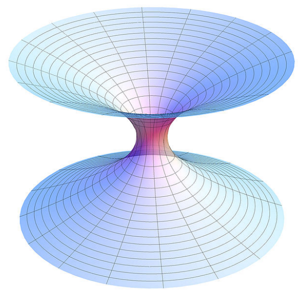
For each input puzzle, output a solution as a sequence of n lines, each of length n, followed by a blank line. If a puzzle has multiple solutions, then output any one of them. If a puzzle cannot be solved, simply print the word "`no`", all by itself, without the delimiting double quotes, followed by a blank line.

Sample Input	Sample Output

<pre>5
5
------- 	
------- 	
------- 	
------- 	
------- 	
------- 	
------- 	
5
-41223-
2-----3
3-----2
2-----1
1-----5
3-----2
-23212-
3
-111-
----3
2---2
----2
-131-
5
--33---
------3
------- 	
------3
------- 	
3------
--324--
5
-----1-
------- 	
2--3--3
2-----2
-1-----
----1-2
-------</pre> 	
	<pre>12345
	21453
	34512
	45231
	53124
	
	15243
	23514
	42135
	54321
	31452
	
	no
	
	51243
	24351
	45132
	13524
	32415
	
	31425
	25341
	42153
	14532
	53214</pre>

Problem W — limit 5 seconds

Wormhole



With our time on Earth coming to an end, Cooper and Amelia have volunteered to undertake what could be the most important mission in human history: travelling beyond this galaxy to discover whether mankind has a future among the stars. Fortunately, astronomers have identified several potentially inhabitable planets and have also discovered that some of these planets have wormholes joining them, which effectively makes the travel distance between these wormhole connected planets zero. For all other planets, the travel distance between them is simply the Euclidean distance between the planets. Given the location of Earth, planets, and wormholes, find the shortest travel distance between any pairs of planets.

Input

- The first line of input is a single integer, T ($1 \leq T \leq 10$) the number of test cases.
- Each test case consists of planets, wormholes, and a set of distance queries.
- The planets list for a test case starts with a single integer, p ($1 \leq p \leq 60$), the number of planets. Following this are p lines, where each line contains a planet name along with the planet's integer coordinates, i.e. *name* $x\ y\ z$ ($0 \leq x, y, x \leq 2 \cdot 10^6$) The names of the planets will consist only of ASCII letters and numbers, and will always start with an ASCII letter. Planet names are case-sensitive (Earth and earth are distinct planets). The length of a planet name will never be greater than 50 characters. All coordinates are given in parsecs.
- The wormholes list for a test case starts with a single integer, w ($0 \leq w \leq 40$), the number of wormholes, followed by the list of w wormholes. Each wormhole consists of two planet names separated by a space. The first planet name marks the entrance of wormhole, and the second planet name marks the exit from the wormhole. The planets that mark wormholes will be chosen from the list of planets given in the preceding section. Note: you can't enter a wormhole at its exit.
- The queries list for a test case starts with a single integer, q ($1 \leq q \leq 20$), the number of queries. Each query consists of two planet names separated by a space. Both planets will have been listed in the planet list.

Output

For each test case, output a line, "`Case` i`:`", the number of the ith test case. Then, for each query in that test case, output a line that states "`The distance from planet`$_1$ `to planet`$_2$ `is` d `parsecs.`", where the planets are the names from the query and d is the shortest possible travel distance between the two planets. Round d to the nearest integer.

Sample Input	Sample Output

<pre>3
4
Earth 0 0 0
Proxima 5 0 0
Barnards 5 5 0
Sirius 0 5 0
2
Earth Barnards
Barnards Sirius
6
Earth Proxima
Earth Barnards
Earth Sirius
Proxima Earth
Barnards Earth
Sirius Earth
3
z1 0 0 0
z2 10 10 10
z3 10 0 0
1
z1 z2
3
z2 z1
z1 z2
z1 z3
2
Mars 12345 98765 87654
Jupiter 45678 65432 11111
0
1
Mars Jupiter</pre>
	<pre>Case 1:
	The distance from Earth to Proxima is 5 parsecs.
	The distance from Earth to Barnards is 0 parsecs.
	The distance from Earth to Sirius is 0 parsecs.
	The distance from Proxima to Earth is 5 parsecs.
	The distance from Barnards to Earth is 5 parsecs.
	The distance from Sirius to Earth is 5 parsecs.
	Case 2:
	The distance from z2 to z1 is 17 parsecs.
	The distance from z1 to z2 is 0 parsecs.
	The distance from z1 to z3 is 10 parsecs.
	Case 3:
	The distance from Mars to Jupiter is 89894 parsecs.</pre>

Problem X — limit 5 seconds

Wrench

Peter works at a factory. He is looking at a list of wrench sizes and needs to find the appropriately sized wrench for various screws and nuts and bolts to do his work. Normally, these sizes are specified using US Customary Unit notation such as 13/16", or 3/8", and so on.

Another way to write 13/16" is 0.8125"

But the reference sheets for various parts round the numbers in weird ways, and give approximations only, so for example 13/16" might turn into 0.812, or 0.813, or sometimes just 0.81, depending on the method of rounding.

Given that Peter is looking for a wrench of size A/B", and it is customary for B to be a power of 2, help Peter find the correct wrench size, where A is a positive integer and B is the minimum possible base (power of 2).

Input

Input starts with the number of test cases, T, on a single line, with $1 \leq T \leq 100$. Each test case is a single decimal number on its own line representing a wrench size, with at most six digits after the decimal point. There need not always be a decimal point. The input value will be greater than zero.

Output

A/B", or C A/B", or C", where B is the minimal power of two such that the exact decimal representation rounded to the number of decimal digits of the input matches the input, using one of the following rounding rules: round up (ceiling), round down (or truncate), or round-to-nearest. The wrench will be less than or equal to 10 inches. There will always be a valid power of two less than or equal to 128.

Sample Input	**Sample Output**

<pre>10
0.81
.8125
0.37
2
2.4
2.99
2.40
1.27
4.
9.242187</pre>
	<pre>13/16"
	13/16"
	3/8"
	2"
	2 3/8"
	2 63/64"
	2 13/32"
	1 17/64"
	4"
	9 31/128"</pre>