Plano West Programming Contest

Wolf Invitational 2016



Problems Index

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Title** | **Shortname** | **Points** | **Writer** |
| **1** | **Coldest Tea in Ba Sing Se** | tea | 2 | Alexander |
| **2** | **Iroh’s Teacups** | cups | 3 | Ritvik |
| **3** | **Fire Nation Secrecy** | secrecy | 3 | Druv |
| **4** | **Cipher of the White Lotus** | lotus | 5 | Ari |
| **5** | **Wan Shi’s Last Theorem** | wanshi | 5 | Alexander |
| **6** | **The Blind Bandit** | bandit | 6 | Aneesh |
| **7** | **Guru Pathik’s Stand** | guru | 7 | Ari |
| **8** | **Bad Signal** | signal | 7 | Ari |
| **9** | **Tale of the Serpent** | serpent | 7 | Marcus |
| **10** | **Finding Appa And Momo** | finding | 8 | Ritvik |
| **11** | **Iroh’s Menu** | menu | 8 | Brian |
| **12** | **Professor’s Zei’s Equation** | zei | 8 | Ari |
| **13** | **Pro Bending Placements** | bending | 9 | Aneesh |
| **14** | **Lost in Si Wong** | lost | 9 | Aneesh |
| **15** | **Word Search** | search | 10 | Aneesh |
| **16** | **Do you even lift?** | lift | 12 | Aneesh |
| **17** | **My Cabbages!** | cabbages | 13 | Alexander |
| **18** | **Identity** | identity | 14 | Jiaming |
| **19** | **Beams of Fire** | beams | 15 | Ari |
| **20** | **Dimensions** | dimensions | 18 | Jiaming |
| **21** | **Pai-Bo** | paibo | 19 | Jiaming |
| **22** | **One Man Army** | army | 20 | Alexander |

1. Coldest Tea In Ba Sing Se (2 Points)

Uncle Iroh is very disappointed with the tea he is given in Ba Sing Se. He does not want to be caught fire bending his tea, but drinking cold tea is a much worse fate! Luckily, he is able to find a thermometer so that he can find out how long it takes him to heat his tea. Unluckily, the Earth Kingdom measures their heat on the Rømer scale, and the Fire Nation measures temperature in Celsius. As a well traveled man, Iroh naturally knows that the conversion between Rømer and Celsius is [°C] = ([°Rø] − 7.5) × 40 ⁄ 21, and that the optimal drinking temperature of Jasmine tea is 85°C, but he needs your help to decide how long to heat his tea.

**Input (tea.in)**

The dataset will begin with an integer N, followed by N lines each consisting of two integers. First, the initial heat of the cup in Rømer, which will never exceed the target temperature. Second, the rate at which Iroh will raise the temperature in Celsius per second.

**Output**

For each line, output the number of seconds it will take Iroh to heat the cup of tea to at least 85°C. Round *up* to the nearest second and do not convert to minutes.

|  |  |
| --- | --- |
| **Sample Input:**  4  20 10  30 5  0 85  23 4 | **Sample Output:**  7  9  2  14 |

2. Iroh’s Tea Cups (3 Points)

Uncle Iroh has put Zuko up to a challenge! He wants Zuko to build character by stacking teacups. However, Zuko has stacked the teacups incorrectly. The teacup stacks are not the same height.

Help Uncle Iroh stack the teacups correctly by figuring out how many cups to move to align everything.

**Input (cups.in):**

The first line contains a number N which denotes the number of data sets. Each data set begins with an integer M, which represents the number of stacks present. The following M lines represent the number of cups in each stack.

**Output:**

Output the minimum number of cups Iroh has to move OR, if it’s impossible to stack evenly, print out the minimum amount of cups he has to sell to make it stack. (See example below)

|  |  |
| --- | --- |
| **Sample Input:**  2  5  1  2  3  4  5  3  3  4  1 | **Sample Output :**  Iroh needs to move 3 cups  Iroh needs to sell 2 teacups to make it stack |

3. Fire Nation Secrecy (3 Points)

Since the gang is now in the Fire Nation, and everyone wishes to keep his/her identity secret, each person must change their name according to a special algorithm for maximum secrecy that Sokka has developed. Sokka determines that the best way to change a name is to take the first half of the name, reverse it, then move it to the end of the name, adjusting capitalization to match the format of a normal name. If a name has an odd number of letters, the “first half” of the name includes the middle letter.

For example: The word “Abcde” becomes “Decba” because C was part of the “first half” that was reversed and moved.

**Input (secrecy.in):**

The input will begin with integer n, followed by n lines of input, each containing one name that must be converted into a codename.

**Output:**

Print each name’s codename on separate lines.

|  |  |
| --- | --- |
| **Sample input:**  4  Appa  Tenzin  Katara  Toph | **Sample output:**  Papa  Zinnet  Aratak  Phot |

4. Cipher of the White Lotus (5 Points)

The Cipher of the White Lotus is a method of encrypting and decrypting secret messages using a special form of modular arithmetic developed by Grand Lotus Iroh, the master of tea and Pai Sho.

The cipher uses two numbers (m and k), to disguise a message (plaintext). Given m, k, and a plaintext to encipher, the resultant encrypted ciphertext is determined by multiplying the numerical value of each letter (A=0, B=1,...,Z=25) by m, then adding k mod 26. For example, if m=5 and k=6, the letter “P” (15) would become 5\*15 + 6 mod 26 which equals 3, or “D”.

Given m, k, and the cipher text, the original message can be determined by subtracting k from the numerical value of the ciphertext letter then multiplying by the modular inverse of m mod 26. The modular inverse of a number is the number it can be multiplied by to equal 1 mod x. For example, 11 and 19 are modular inverses mod 26 because 11 \* 19 mod 26 equals 1. So if we use the example from before: to decrypt the message “D”, we can determine that the modular inverse of 5 mod 26 is 21, then evaluate (3 - 6) \* 21 mod 26 which equals -11. Since no letter equals -11, you “wrap it around” by adding 26 to it to get 15, which equals “P”, the original message.

Encryption: C = (m \* P + k) mod 26

Decryption: P = ((C - k) \* m-1) mod 26

(C is ciphertext, P is plaintext, m and k are constants, m-1 is the modular inverse of m)

**Input (lotus.in):**

An integer N determines the number of data sets to follow. For each data set, there are two lines. The first line contains m which is in the set of all numbers less than 26 that are relatively prime to 26, i.e., { 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25 }, and k, where 1 ≤ k ≤ 25. The next line contains the keyword ENCRYPT or DECRYPT then a string of text.

**Output:**

Print the encrypted or decrypted version of the text in each test case based on the m and k values provided and the keyword preceding the text. The output should be in all caps.

|  |  |
| --- | --- |
| **Sample Input:**  3  5 6  ENCRYPT The chicken crossed the road  5 6  DECRYPT GFFUTA QUDPAN  7 1  ENCRYPT I like pie | **Sample Output:**  XPA QPUQEAT QNYSSAV XPA NYGV  AFFINE CIPHER  F AFTD CFD |

5. Wan Shi’s Last Theorem (5 Points)

Wan Shi Tong, the Owl Spirit that guards the Spirit Library, has mathematically proven that there are no integer solutions to the equation:

Even though he is known as “Wan Shi Tong, he who knows 10,000 things”, he’s not quite versed in the workings of human civilization yet. He believes that radios are made up of miniature men that sing and play musical instruments and is perplexed by the complexity of Java.

When he types into Java the expression:

System.out.println(50 ^ 3 + 1 ^ 3 == 54 ^ 3);

The output is

true

But Blasphemy! Wan Shi Tong is never wrong!

How is this possible?

**Input (wanshi.in)**

You will be given some unknown number of datasets. Each dataset consists of two integers, x and n. For each dataset, you will need to find the smallest positive value of Y > 0 such that X^n + Y^n == Z^n

**Output:**

Print the value of Z.

**Preconditions:**

|  |  |
| --- | --- |
| **Sample Input:**  7 9  60 30  10 11  5 10  53 6 | **Sample Output:**  13  35  6  14  50 |

6. The Blind Bandit (6 Points)

Toph has been gambling again, and she’s “making” more money than ever! However, she’s earned so much money her pockets make a huge racket every time she takes a step. To hide her ‘activities’ from Katara, help Toph exchange her money at the bank to minimize the number of coins she needs in her pocket.

**Input (bandit.in):**

An integer n will denote the number of data sets to follow. For the next n lines, there will be a single integer denoting the value in coins that Toph has. Using the standard American coin values (1, 5, 10, 25, 50, 100), determine how many of each coin Toph should have.

**Output:**

On a single line per data set, print out how many pennies, nickels, dimes, quarters, half dollars, and dollar coins Toph should get from the bank to have the minimum number of coins.

|  |  |
| --- | --- |
| **Sample Input:**  2  16  101 | **Sample Output:**  1 1 1 0 0 0  1 0 0 0 0 1 |

7. Guru Pathik’s Stand (7 Points)

Guru Pathik sells cups of onion-banana juice at his stand. He must calculate the profit he makes over the course of several days. Before each time period, he buys the materials needed to make onion-banana juice, namely onions, bananas, and cups. Then, each day, customers buy a certain number of cups of onion-banana juice. Given the prices of each commodity and the price at which Pathik sells a cup of onion-banana juice, determine his profit or loss for the given period of time.

**Input (guru.in):**

The first line contains O, B, C, and J: the prices of a pound of onions, the price of a pound of bananas, the price of 5 cups, and the price of a single cup of onion banana juice respectively. The next line contains an integer N, the number of days that the stand was open. The next N lines contain four integers for each day -- A, C, D, and E: how many pounds of onions were bought, how many pounds of bananas were bought, the number of cups bought, and the number of cups of onion-banana juice sold.

**Output:**

Print **SUCCESS** if the stand’s profit is positive and **FAILURE** if its profit is negative followed by the total profit (sales - costs) of Guru’s stand over the N days, rounded to the nearest hundredth. When calculating costs, the stand owner never buys single cups. Cups are always bought in packs of 5 and there may be extras that are thrown out. Supplies do not carry over to the next day.

|  |  |
| --- | --- |
| **Sample input:**  2.14 1.05 3.00 2.50  3  2 3 4 5  1 1 1 2  5 4 5 7 | **Sample output:**  FAILURE -20.52 |

8. Bad Signal (7 Points)

A string of bits is transmitted from Korra to President Raiko to warn of Kuvira’s attack on Republic City. However, one or more bits are accidentally flipped during the transmission, leaving Raiko unprepared for the invasion.

Luckily, Raiko was prepared for this possibility. Korra and Raiko have a set of predetermined codes that represent different messages. Each transmitted code has a distance, or number of flips needed to match one of the predetermined codes. Help President Raiko determine the message he was supposed to receive based on the distances of the actual codes he receives and their closest matching predetermined strings.

It’s up to you to save Republic City!

**Input (signal.in):**

An integer N determines the number of data sets to follow. For each data set, the first line contains three integers X, Y, and Z. X is the number of bits in each string, Y is the number of predetermined strings, and Z is the maximum number of errors in a string. The next Y lines will contain a predetermined string representing the decoded message followed by its code (comprised of 0’s and 1’s), with the message and its code separated by a space. Following the last line, the transmitted codes are displayed with up to Z errors in each code. There may be no errors in any given code, and there will be exactly one ‘best match’ for each transmitted code.

**Output:**

For each test case, print out the intended message on a separate line.

|  |  |
| --- | --- |
| **Sample Input:**  1  5 4 1  Aang 11111  Asami 10001  Korra 00100  Bolin 01010  11111 10101 01110 11011 10100 | **Sample Output:**  Aang Asami Bolin Aang Korra |

9. The Tale of the Serpent (7 Points)

Aang and the gang are planning to cross the dreaded Serpent’s Pass, infamous for the guardian Serpent that blocks their way. Toph, who cannot “feel” where she is, becomes anxious, wondering where the Serpent is at every moment. Given the Serpent’s circular Path, given that it always starts from (0, 0), inform Toph where the beast is using Cartesian coordinates so she can hurl a massive boulder at it if needed.

Converting a circular pattern into Cartesian coordinates:

Pattern: always East, then North, then West, then South

The rate of travel starts at 1, then doubles every 2 moves. See the picture above.

**Input (serpent.in):**

The data set will begin with a single integer n, denoting the number of test cases in the file. The next n data sets will be the length of the pattern stretched out in a straight line.

**Output:**

Find the coordinate point (x, y) of the serpent after travelling AT LEAST the specified distance.

|  |  |
| --- | --- |
| **Sample input:**  6  1  3  6  9  15  17 | **Sample output:**  (1.00, 0.00)  (0.00, 1.00)  (-1.00, -1.00)  (2.00, -1.00)  (-1.00, 2.00)  (-2.00, 1.00) |

10. Finding Appa and Momo (8 Points)

Aang can’t find Appa or Momo (his pets) among the tall grass. However, he can see their feet!

If he can see a pair of two consecutive front feet - “((“ (Appa) or “[[“ (Momo) - and a pair of consecutive back feet - “))” (Appa) or “]]” (Momo) - then he can mark that as a possible location for either pet.

Note: Each “foot” can be used in more than one location. i.e. “((())” would be two different locations.

Note: sample input file contains sample input 1

**Input (finding.in):**

The input will be a string containing only types of characters that make up the feet given above.

**Output:**

Given the input string print the number of possible locations for Appa and for Momo

|  |  |
| --- | --- |
| **Sample Input 1:**  (([[))]] | **Sample Output 1:**  Appa: 1  Momo: 1 |
| **Sample Input 2:**  (((([[[)]] | **Sample Output 2:**  Appa: 0  Momo: 2 |

11. Iroh’s Menu (8 Points)

Iroh is making his tea menu, but some nefarious villain has tampered with the delimiters in his names so that they do not all match up. The delimiters are the parenthesis ( and ), braces { and }, and brackets [ and ]. Help Iroh out by determining whether a given expression’s delimiters are formatted correctly. Each opening delimiter should be matched by its corresponding closing delimiter, and opening delimiters that occur later in the expression should be closed before those appearing earlier. Some examples include:

a [d] //correct

a (b + [c]) //correct

a {b \* (c + d)] //incorrect, ] doesn’t match {

a((b+c) //incorrect, nothing matches opening (

1 + bc //correct

**Input (menu.in):**

The data file will contain multiple data sets, with each line representing a different data set. Each line will each contain an expression involving any possible range of characters, with the number of characters (not including whitespace) being strictly less than 20.

**Output:**

For each test case, output “True” if the expression contains correct pairs of delimiters, and “False” otherwise.

|  |  |
| --- | --- |
| **Sample Input:**  1 + (2\*3)) + 5  acd([5])  bcg | **Sample Output:**  False  True  True |

12. Professor Zei’s Equations (8 Points)

Professor Zei needs to solve some unknown equations he found at Wan Shi Tong's Library. However he doesn’t have enough time to convert from postfix to infix quick enough. He needs your help to write a program to do this.

Evaluate the value of an equation given in postfix notation. In postfix notation, the operators come after the operands. For example, 2 3 + is postfix for 2 + 3. In a longer expression, when each operator is evaluated the resultant value is plugged back in for use with any later operators. For example, 2 3 + 5 - can be solved as follows: 2 3 + becomes 5, making the expression 5 5 - which evaluates to 0. Note that when dividing, remainders are ignored, i.e., Wan Shi Tong used integer division.

**Input (zei.in):**

An integer N determines the number of data sets to follow. The next N lines contain a postfix expression using integers and one or more of the following operators: +, -, \*, /.

**Output:**

Print the value of the postfix expression. The answer will always be an integer value.

|  |  |
| --- | --- |
| **Sample input:**  2  3 6 \* 2 /  1 2 + 3 \* 6 + 2 3 + / | **Sample output:**  9  3 |

**Explanation:**

The first expression is the same as (3 \* 6) / 2 in infix (regular) notation. This evaluates to 9.

The second expression is the same as (((1 + 2) \* 3) + 6) / (2 + 3) in infix notation. This evaluates to 3.

13. Pro Bending Placements (9 Points)

Despite his calm and severe demeanor, Tenzin is actually an avid fan and a surprisingly informed analyst of pro bending matches. Before he can start his analysis and make predictions about which teams will place in the league, he likes to alphabetically list out all possibilities of team placements, especially since the number of “winners” changes from league to league (some years top 3, some top 5, just a single top team, etc.). Help Tenzin make his list.

**Input (bending.in):**

The first line of the data file will contain a single integer x denoting the number of data sets in the file. The first line of each data set will contain two integers n and m. The next n lines contain the names of all the pro bending teams registered in the league.

**Output:**

For each data set, print out an alphabetized list of all the possible permutations of m winners, with each possibility printed on its own line with the format <team1>, <team2>, <team3>...,<team n> where each <team#> is the name of the team that achieved that respective place. Print a blank line between each data set.

|  |  |
| --- | --- |
| **Sample input:**  1  3 3  Fire Ferrets  Badgermoles  Rabaroos | **Sample output:**  Badgermoles, Fire Ferrets, Rabaroos  Badgermoles, Rabaroos, Fire Ferrets  Fire Ferrets, Badgermoles, Rabaroos  Fire Ferrets, Rabaroos, Badgermoles  Rabaroos, Badgermoles, Fire Ferrets  Rabaroos, Fire Ferrets, Badgermoles |

14. Lost in Si Wong (9 Points)

Appa has broken free from the clutches of sandbenders and is now trying to find Aang in the vast Si Wong desert! However, the desert is far too large for a single sky bison to traverse alone, and to make matters worse, the sandbenders have sent tribesmen everywhere to capture their lost prize. Given Appa’s initial position, the position of all the sand-bending tribesman, and Aang’s last known position on the desert grid, help Appa find his beloved master.

**Input (lost.in):**

An integer N will denote how many data sets are to be processed.

Each data set will consist of a single line of two integers X and Y, indicating the width and height (respectively) of the desert grid. The next y lines will consist of x characters.

* o denotes a safe area that Appa can move to
* a denotes Appa’s initial position
* A denotes Aang’s last known position
* S denotes a Sandbender tribesman

**Output:**

Print out the desert grid as given in the input, but fill in the path Appa should take with #. Only replace open spaces (“o”) that Appa must pass through, leave “A” and “a” as they appear in the original grid. Appa can only move in the cardinal directions (North, South, East, or West). There will be exactly one solution to every data set. Print out a blank line after every answer grid.

|  |  |
| --- | --- |
| **Sample input:**  1  4 4  SSAS  SSoo  SSSo  aooo | **Sample output:**  SSAS  SS##  SSS#  a### |

15. Word Search (10 Points)

While Iroh is locked away in prison after Zuko betrays him, he finds himself with plenty of spare time on his hands. Other than exercising, which takes up most of his day, he has occupied himself with a new hobby: solving word searches.

The word searches he has are the standard word search puzzles: the grid is of size x by y, and words can be found in cardinal directions, diagonally, and *backwards****.***

***Note: the words to find are case sensitive***

**Input (search.in):**

The data set will begin with a single integer n, denoting how many test cases are in the data file. The next n data sets will consist of two positive integers x and y, which indicate the height and width of the grid of letters (respectively). The next x lines will each consist of y letters. After the grid, there will be another integer z that indicates the number of queries, and the next z lines will consist of strings of varying lengths to find.

**Output:**

For each query, print out “I AM THE DRAGON OF THE WEST! “ if the word is found and “I have been betrayed.“ if it is not. Print out a blank line between the answers to each dataset.

|  |  |
| --- | --- |
| **Sample input:**  1  4 4  aang  thea  irbe  nder  3  aang  bean  red | **Sample output:**  I AM THE DRAGON OF THE WEST!  I have been betrayed.  I AM THE DRAGON OF THE WEST! |

16. Do you even lift? (12 Points)

Toph has started her metalbending academy, and business is booming! Unfortunately, her first batch of students isn’t exactly rock-solid in bending, so they keep trying to find an easy way out. One exercise Toph has set up involves lifting up a certain value in coins off the floor and into a box. To minimize the amount of effort for the students, determine the minimum number of coins that a student must lift to complete the task for a given value.

**Input (lift.in):**

An integer n will denote how many data sets are in the following file. Each data set is two lines. The first line in each data set will give the values of each coin of a currency in ascending order (For example, the United States coin set would be 1 5 10 25). The second line of each data set will contain a single integer denoting how much value in coins the student must lift.

**Output:**

For n data sets, print out a single integer per set denoting the minimum number of coins a student must lift to complete the task.

|  |  |
| --- | --- |
| **Sample Input:**  2  1 5 10 25 50 100  261  1 2 3 4 5  7 | **Sample Output:**  5  2 |

17. My Cabbages! (13 Points)

|  |  |
| --- | --- |
| The cabbage merchant is pushing his cart through Ba Sing Se when he hears the sound of whooshing wind. He knows the sound all too well and immediately starts to head for the exit of this part of town. Meanwhile, our young air-bending hero is fleeing the same part of town from a different location. Assuming they are both taking the optimal path out of this part of the city and cannot move diagonally, do they collide?  Note that paths crossing is not enough to determine if there was a collision. They must occupy the same space at the same time. There will be no ties, each dataset will have exactly one definite answer.  **Input (cabbages.in):**  Input may contain any number of datasets.  Each dataset’s first line gives the number of rows and columns for a maze. Every line after that (up until the next dataset) is part of the maze.   * . marks an open space * @ marks a wall * A marks Aang’s initial position * C marks the cabbage merchant’s initial position   **Output:**  For each dataset, if there was no collision, print the name of the person who go out first (Aang or Cabbage Merchant). If there was a collision, print “My Cabbages!”. | **Sample Input:**  9 8  @@@@@@@@  @...C..@  @.@@.@.@  @.@@.@.@  @.@@.@..  @.@..@.@  @.@@.@.@  @...A..@  @@@@@@@@  5 7  @@@@@@@  ..A...@  @@@@@.@  @C....@  @@@@@@@  7 7  @@@@@@@  ..@...@  @...@A@  @@@.@.@  @@@.@.@  .C..@.@  @@@@@@@  **Sample Output:**  My Cabbages!  Aang  Cabbage Merchant |

18. Identity (14 Points)

Dock… Xu? …Bushi…? ...uh… Some guy suffers from severe identity disorder. Depending on the hat that he’s wearing, he converts his numbers into different negative bases.

A numerical system that uses a negative base instead of a positive one can be used to uniquely represent all integers, positive or negative, without the use of a positive or negative sign. Like a positive base, the value of the number is determined by calculating the sum of each digit multiplied by the base raised to a power that corresponds to the position of the digit, with the right most digit corresponding to a power of 0 and every other digit corresponding to a power that is one greater than the power of corresponded to by the digit to its right. For example, the decimal number 10 is written as 11110 in base -2 since 1\*(-2)4 + 1\*(-2)3 + 1\*(-2)2 + 1\*(-2)1 + 0\*(-2)0 = 10. The value of each digit must be less than the magnitude of the base. Since we will be using bases with magnitudes greater than 10, allowing the value of each digit to span multiple digits in decimal representation, you must separate the value of each digit with a space to distinguish them.

**Input (identity.in)**

The first line of the input gives the number of test cases that follow. Each test case consists of a line with two integers. The first integer is the decimal representation of a number to be converted. The second integer will always be less than -2 and gives the base to convert the first integer to.

**Output**

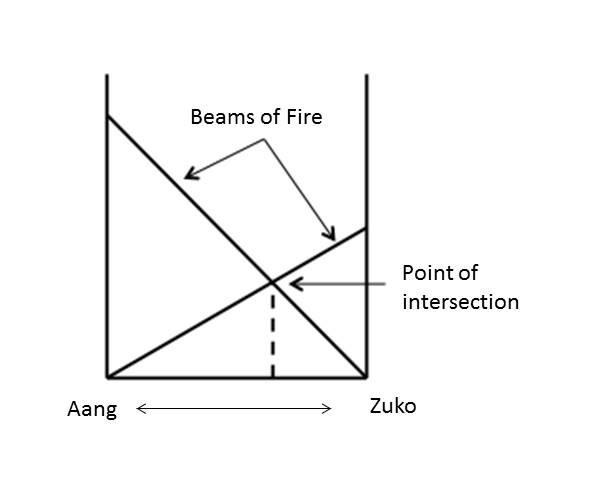
For each test case, output one line containing the decimal value of each digit separated with spaces. Order the digits from greatest magnitude to least magnitude. Do not print leading zeros. If the number is zero leave the line empty.

|  |  |
| --- | --- |
| **Sample Input:**  6  10 -2  0 -4  -415 -12  -1000 -52  683 -209  1582723 -2842 | **Sample Output:**  1 1 1 1 0  1 10 11 5  20 40  1 206 56  1 2286 2571 |

19. Beams of Fire (15 Points)

During Zuko and Aang’s ceremony to meet the Fire Dragon Spirits, they must perform one last ceremony to finish their training: shooting a powerful beam of rainbow flames in the sky to discover the true purpose of fire. However, they must be at a certain distance apart from each other to perfect the ceremony and allow the dragons to enter their souls.

They shoot two beams of fire upwards with varying lengths, intersecting one another in the sky. Given the lengths of each beam of fire and the height at which they intersect, calculate how far apart the two are standing.



**Input (beams.in):**

An integer N will determine the number of data sets to follow. The next N lines contain three integers S, L, H: the full length of the shorter beam, the full length of the longer beam, and the height that they intersect respectively.

**Output:**

Print out the distance between Zuko and Aang rounded to four decimal places. Each test case should be displayed on its own line.

|  |  |
| --- | --- |
| **Sample Input:**  1  20 30 10 | **Sample Output:**  12.3119 |

20. Dimensions (18 Points)

Korra’s battle with Vaatu resulted in the opening of several portals, leading to new and unexplored dimensions beyond the Avatar World. However, when these portals were opened, Korra lost Raava (the spirit of peace and light) to another dimension.

Help Korra find the length of the shortest path to Raava , as represented by an orthogonal maze. It may have more than two dimensions.

**Input (dimensions.in):**

The first line contains an integer t representing the number of test cases to follow. Each test case contains two lines. The first line begins with an integer d representing the number of dimensions of the maze. The rest of the line will contain d integers (I0, I1, I2, ... Id-1) that represent the length of each of the dimensions. The second line will contain I0 \* I1 \* … \* Id-1 characters that represent each cell of the maze. If we index each dimension from 0 to In - 1 where n is the dimension number, then index each of the characters in the line from 0 to I0 x I1 x ... x Id-1 - 1, then character 0 will represent cell (0, 0, …, 0), character 1 represents cell (1, 0, …, 0), character I0 represents cell (0, 1, …, 0), character I0 x I1 x … x Id-1 - 1, represents cell I0 -1, I1 -1, ... Id-1 - 1), etc. The starting cell is represented by the character “A”. The ending cell is represented by the character “B”. A cell can be traveled through if it is the starting cell, the ending cell, or if it is represented by the character “.”. Otherwise, the cell will be represented by the character “#”.

**Output**

For each test case, output one line containing the minimum number of steps necessary to travel from the starting cell to the ending cell. If there is no open path from the starting cell to the ending cell, output “No Solution”.

**Sample Input:**

1

3 3 5 4

A...#.##.#######.....##.#B.###...##.##.#.###.#.##.#..##.##..

**Sample Output:**

7

21. Pai-Bo (19 Points)

|  |  |
| --- | --- |
| Iroh’s adventures in the Spirit World have driven him mad since he is without his favorite pastime Pai-Sho. However, with the help of his spirit friends, he develops a new game in remembrance of his beloved board game, conveniently named Pai-Bo.  In Pai-Bo, two players alternatingly make marks on an orthogonal grid to try to form certain predetermined arrangements of marks that grant victory. Each game will use a different (given) set of rules and specifications, assuming both players are masters and always make the best series of moves, determine the victor of the game.  **Input (paibo.in):**  The first line contains the integer t, the number of test cases to follow.  Each test case begins with a line containing four space separated integer values - r, the number of rows in the orthogonal grid; c, the number of columns in the orthogonal grid; n, the number of arrangements of marks that grant victory to the first player; and m, the number of arrangements of marks that grant victory to the second player.  Following this line will be n + m representations of arrangements of marks that grant victory. The first n arrangements will grant victory to the first player if any of them are formed with the first player’s marks. The next m arrangements will grant victory to the second player if they are formed by the second player’s marks.  For each arrangement of marks that grant victory there will be a line of two space separated integers, h and w, representing the height and width of each arrangement respectively, followed by a grid representing the arrangement where the character ‘X’ represents a cell marked by the winning player and the character ‘.’ represents a cell of any state.  Each arrangement will grant victory if they appear anywhere on the orthogonal grid, but will only grant victory in the orientation it is given in. Reflections and rotations of each given arrangement will only grant victory if the reflection or rotation in question is also given.  **Output:**  For each test case, output “1st Player” if the first player wins, “2nd Player” if the second player wins, or “Tie” if neither player wins. | **Sample Input:**  1  3 3 4 4  1 3  XXX  3 1  X  X  X  3 3  X..  .X.  ..X  3 3  ..X  .X.  X..  1 3  XXX  3 1  X  X  X  3 3  X..  .X.  ..X  3 3  ..X  .X.  X..  **Sample Output:**  Tie |

22. One Man Army (20 Points)

While locked away in a fire nation prison, old Uncle Iroh decides to get in shape. While doing crunches, push-ups, and other fire nation favorites, Iroh gets bored. To spice things up, he defines a few fun rules to amend his counting style.

1. He will pick some integer X to start counting from.
2. He will pick some integer N to count by.
3. When he gets to the number 2147483647, he overflows, down to -2147483648
   1. He follows the rules for basic 32 bit integer overflow.
4. He will stop his exercise when he gets to 3.
5. He is wise enough to pick only numbers that will allow him to stop.

As a guard, you get bored too, so you decide to figure out how many repetitions of each exercise Iroh will do with a given X and N.

**Input (army.in):**

You will be given an unknown number of datasets, each only one line long. Each line consists of two integers, X and N, corresponding to the X and N Iroh is using.

**Output:**

For each dataset, output one line with the number of repetitions Iroh makes.

Be careful. We are impatient.

|  |  |
| --- | --- |
| **Sample Input:**  7 9  60 31  10 11  5 10  53 6 | **Sample Output:**  477218588  3048041305  3904515723  858993459  1431655757 |

