2018 Myanmar Collegiate Programming Contest

University of Computer Studies, Yangon

August 18, 2018

Problem Set

Please check that you have **10** problems and **14** pages.

Problem	Problem Name	Balloon Color
A	Taxi Driver	Silver
B	Binary Match Encryption	Lime
C	Pattern Matching	Blue
D	Round Robin Tournament	Green
E	Adventure in a maze	Pink
F	Alphabet Symmetry	White
G	Brace Matching	Red
H	Gold Mining	Orange
I	Babylonian Numbers	Yellow
J	Need for Speed	Purple

Note: The input and output for all the problems are standard input and output.

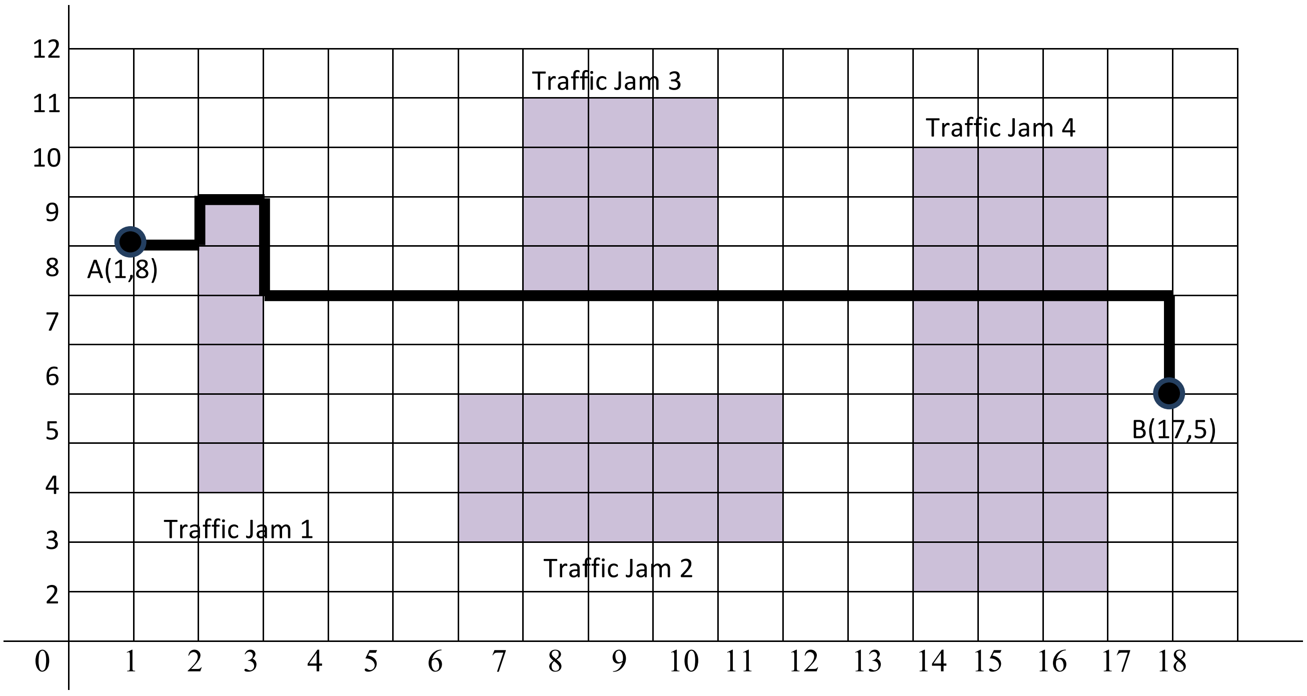
August 18, 2018

Problem A: Taxi Driver

In New Yangon City, streets are designed as a rectangular grid. Each intersection point has integer Cartesian coordinates x and y. The taxi driver wants to travel from city block intersection point A with coordinate (*xa, ya*) to intersection point B with coordinate (*xb, yb*). He needs to drive exactly $|xa-xb|+|ya-yb|$ blocks. Usually, it always takes 10 time units for driving one block in non-traffic time. So, one can easily compute the time it takes to get from *A* to *B*. However, there are a lots of traffic jams that occur in New Yangon City. So, it is difficult to estimate the minimal driving time to reach from *A* to *B*.

Traffic jams area in the city is defined by rectangular shaded region. That shaded area is specified by (x, y) coordinates of its lower-left and upper-right corners. The time to travel one block in the traffic jam is specified. All of the streets that are inside the shaded region are affected by the traffic jam. Sometimes, it is better for taxi driver to drive around the traffic jams to save time. But sometimes it is better to drive through some traffic jams to save time. Your job is to write the program for taxi driver to estimate the minimal time to reach the destination.

The following figure shows the sample simulation of traffic jams.



In the above example, the taxi driver wants to travel from intersection point A(1, 8) to destination point B(17, 5). He drives 18 blocks outside of traffic jams with ***10*** time units per block, and 3 blocks are driven through the traffic jam with ***15*** time units per block to save time. Therefore, 225 time units are required to reach the destination.

Input

The first line of input is the number of test cases. In each test case, the first line contains four integer numbers *xa*, *ya*, *xb*, and *yb* that represents city block intersection coordinate points of the source and destination. The second line of the input contains a single integer number n ($0 \le n \le 100$) which specifies the number of traffic jams. The following n lines describe information about traffic jams area. Each traffic jam is described by five integer numbers, $x1, y1, x2, y2, t$ where first four numbers are coordinates of the bottom-left and top-right corners of the traffic jam area ($x1 < x2, y1 < y2$), and t ($10 < t \le 100$) is the time it takes to travel one block inside this traffic jam. The time units for driving one block in non-traffic area is always 10 time units. The Areas of traffic jams neither intersect nor touch each other. Start and finish points are different and do not lie inside nor on the border of any traffic jam.

Output

Write the output for a single integer that is the minimal driving time from city block intersection point *A* to intersection point *B*.

Sample Input	Sample Output

1
1 8 17 5
4
2 3 3 9 45
6 2 11 5 14
7 7 10 11 20
13 1 16 10 15
	225

Problem B: Binary Match Encryption

You've just learned about binary numbers and encryption and want to create your own encryption system based on binary matching. The encryption system is as follows: The input is a string of text. You convert the text to binary by substituting each character in the string with its ASCII code (8-bits). You must match two binary numbers from left to right (e.g. 10001101=>10, 00, 11, 01). Then you convert the binary into a sequence of integers as follows: The "00" occurrence is encrypted as Perfect square number, "11" occurrence is encrypted as Prime number and ("10" or "01") occurrences are encrypted as others number (neither Perfect square nor Prime). So, for example the binary 0011001001000100 would be converted into the sequence of `4,2,9,6,8,16,10,25`. The problem would just be to take as input string of text and convert that string into the cipher text of a sequence of integers in this way.

(***Hints:***

- The replacing integers N, 2≤N≤1000. All integers (Perfect square, Prime, Others) must replace in the increasing order respectively and don't repeat.
- Perfect square**:** 4, 9, 16, 25, ……
- Prime **:** 2, 3, 5, 7, ………
- Others **:** 6, 8, 10, 12, ……)

Input

The first line of the input contains a single integer T, (1<=T<=25) that indicates the number of test cases. Each test case consists of a string of length S, (1<=S<=20). Each string can include only letters [a-zA-Z] and numbers [0-9] and space.

Output

For each test case, output the cipher text of a sequence of integers. See the samples for the exact output format.

Sample Input	Sample Output

`4`
`A3`
`2D`
`4ab`
`14FA`
	`Case 1: 6,4,9,8,16,2,25,3`
	`Case 2: 4,2,9,6,8,16,10,25`
	`Case 3: 4,2,6,9,8,10,16,12,14,15,25,18`
	`Case 4: 4,2,9,6,16,3,8,25,10,36,12,14,15,49,64,18`

Problem C: Pattern Matching

In computer science, **pattern matching** is the act of checking a given sequence of tokens for the presence of the constituents of some pattern. In this domain, one or more strings called "Pattern" are to be searched within a well-built string or "Text". Some applications are Spell Checkers, Spam Filters, Intrusion Detection System, Search Engines, Plagiarism Detection, Bioinformatics, Digital Forensics and Information Retrieval Systems, etc.

Given two strings X and Y, your task is to count the occurrences of a given string X in a given string Y. In each occurrence, you should also find out all the positions of the first character of X within Y.

Example:

If X=GATGAT, X in Y has found 3 times, at the positions of first character X in 5, 8 and 18 as follows:

Y=ACTCA**GATGATGAT**TGAA**GATGAT**TGCTCTTAC

↑ ↑ ↑

Input

The first line of the input consists of an integer T<=100 given the number of test cases. The first line of each test cases contains a string X, where the length of X <=10. Next, there will be a string Y and length of Y <= 100.

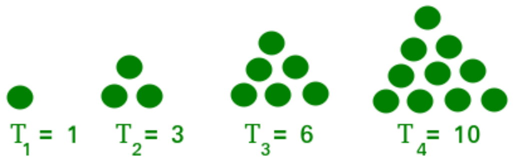
Output

For each test case, the total number of occurrences and positions of occurrences should be printed. The positions output should be sorted in ascending order.

Sample Input	Sample Output
3	3
GATGAT	5 8 18
ACTCAGATGATGATTGAAGATGATTGCTCTTAC	5
A B A B	0 4 22 32 36
A B A B A B D A C D A A B A B C A B A B A B	4
13231	3 7 11 34
112132313231323133221313222113313213231	

Problem D: Round Robin Tournament

The fairest way to determine the champion from among a known and fixed number of participants is a round-robin tournament. Each participant has equal chances against all other opponents. The element of luck is seen to be reduced as compared to a knockout system. Final records of participants are more accurate as they represent the results over a longer period against the same opposition. This is helpful to determine the final rank of all competitors, from strongest to weakest, for purposes of qualification for another stage or competition. In a tournament format that uses a round-robin group stage, the number of matches that need to be played between *n* teams is equal to the triangular number T_{n-1} which counts objects arranged in an equilateral triangle as in the diagram. The *n*th triangular number is the number of dots in the triangular arrangement with *n* dots on a side.



You are supposed to schedule the matches for your football federation. You decide to use round-robin method since you would like to arrange the fair play game. For example, 4 teams require 6 matches, and a group stage with 8 teams requires 28 matches. Then, you have to write a program to check whether it match up the number of teams and the number of guessed matches.

Input

The first line of input contains a single integer **N**, (1≤ **N** ≤50) which is the number of test cases. The input for each test case is given in a single line containing the number of teams **T**, (2≤ **T**≤50) and the number of guessed matches M, (1 ≤ **M**≤3000).

Output

For each test case, display correct or incorrect decision for the number of guessed matches and the correct number of matches if the guess number is wrong.

Sample Input	Sample Output
`5`	`INCORRECT 10`
`5 15`	`CORRECT`
`10 45`	`INCORRECT 6`
`4 8`	`INCORRECT 190`
`20 90`	`CORRECT`
`8 28`	

Problem E: Adventure in a Maze

You have got a maze, which is a N*N Grid. Every cell of the maze contains these numbers 1, 2 or 3.

If it contains 1 : means we can go Right from that cell only.
If it contains 2 : means we can go Down from that cell only.
If it contains 3 : means we can go to two directions, Right and Down from that cell.

You can't go out of the maze at any time. Initially, you are on the Top Left Corner of The maze (Entry). And, you need to go to the Bottom Right Corner of the Maze (Exit). You need to find the total number of paths from entry to exit point.

There may be many paths but you need to select that path which contains the maximum number of adventure. The adventure on a path is calculated by taking the sum of all the cell values that lies in the path. If no valid path exists, adventure is zero. Since the total paths may be very large, output it Modulo (1e9+7).

Input

The first line contains **T** **(1<=T<=10)** denoting the number of test cases. The first line of each test case contains an integer **N** **(1<=N<=1000)** denoting the size of the Matrix. The next **N** lines contain **N** integers, each denoting the cell of the Maze.

Output

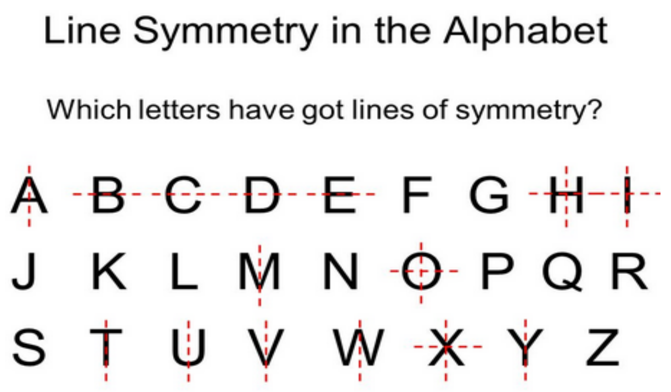
For each test case, output two integers denoting the total number of paths and maximum adventure in a valid path from entry to exit.

Sample Input	Sample Output
2	4 18
5	0 0
1 1 3 2 1	
3 2 2 1 2	
1 3 3 1 3	
1 2 3 1 2	
1 1 1 3 1	
3	
1 2 3	
2 2 1	
2 2 3	

Explanation:
There are total 4 Paths Available out of which The Max Adventure is 18
Total possible Adventures are 18,17,17,16.
Of these 18 is the maximum.

Problem F: Alphabet Symmetry

We experience symmetry every day. Mirror (or reflection) symmetry divides a figure or design into halves that are mirror images. In other words, objects are the same on both sides of a line (usually in the middle). This line or axis can be located either vertically or horizontally. Butterflies are good examples of mirror symmetry in nature. In fact, most animals and plants exhibit some forms of symmetry in their body shape and their markings. Mirror symmetry is found in manufactured objects, too. In this activity, you look for symmetry in letters of the capital alphabet.


Line Symmetry in the Alphabet
Which letters have got lines of symmetry?
A B C D E F G H I
J K L M N O P Q R
S T U V W X Y Z

Do you see any lines of symmetry here? Which letters have zero line of symmetry? Which letters have one line of symmetry? Which letters have two lines of symmetry? The example letters F and G have zero lines of symmetry. Those letters cannot be folded in half in any way with the parts matching up. The example letters A and B have only one line of symmetry. Notice that the A has a vertical line of symmetry, while the B has a horizontal line of symmetry. The example letters H and I have two lines of symmetry (vertical and horizontal).

Based on the number of lines of symmetry a letter has, that's how much the letter is "worth". Letters with zero lines of symmetry are worth zero points. Letters with one line of symmetry are worth one point. Letters with two lines of symmetry are worth two points. Do you think you can find out how much my name is worth (MRS. WISBROCK)?

M R S. W I S B R O C K
1+0+0+1+2+0+1+0+2+1+0 = 8

Now, let's find out how much the words are worth.

Input

The input consists of one or more test cases of words, followed by a final line containing only the value **0**. Each set starts with a line containing an integer, N (1<= N <= 25), which is the number of words in the set, followed by *N* words, one per line. Each word is at most 15 characters long.

Output

For each test case print case number on a line, where *n* starts at 1, followed by the output set as shown in the sample output.

Sample Input	Sample Output
4	Case 1:
MYANMAR	MYANMAR 5
COLLEGIATE	COLLEGIATE 9
PROGRAMMING	PROGRAMMING 7
CONTEST	CONTEST 6
2	Case 2:
ALPHABET	ALPHABET 7
SYMMETRY	SYMMETRY 6
5	Case 3:
HAT	HAT 4
ICEBOX	ICEBOX 9
DECIDED	DECIDED 8
X-BOX	X-BOX 7
WITHOUT	WITHOUT 10
0	

Problem G: Brace Matching

Mg Mg is a computer science student from University of Computer Studies. He usually encounters the most common syntax errors when writing code in text editor because of misplaced and unmatched braces. So he wants to write a program to match braces - square brackets [and], curly brackets { and }, or parentheses (and) . Brace matching is particularly useful when many nested program codes are involved. Brace matching can also be a tool for code navigation. In brace matching, each opening or left brace should be matched by a closing or right brace. If they are not the same type (for example: the opening brace is curly bracket but the closing is parentheses), an error occurs. Also, if there is no opening brace to match a closing one, or if a brace has not been matched, an error occurs. So a pair of *braces* is not matched if the set of *braces* it encloses are not matched. On the other hand, a pair of braces is matched if it contains no unmatched braces, and the subset of braces enclosed within a matched pair of braces is also a matched of braces.

Input

The first line of input contains a single integer T, $1 \leq T \leq 1000$ which is the number of test cases. Each test case contains a single line. It is a string expression to match braces.

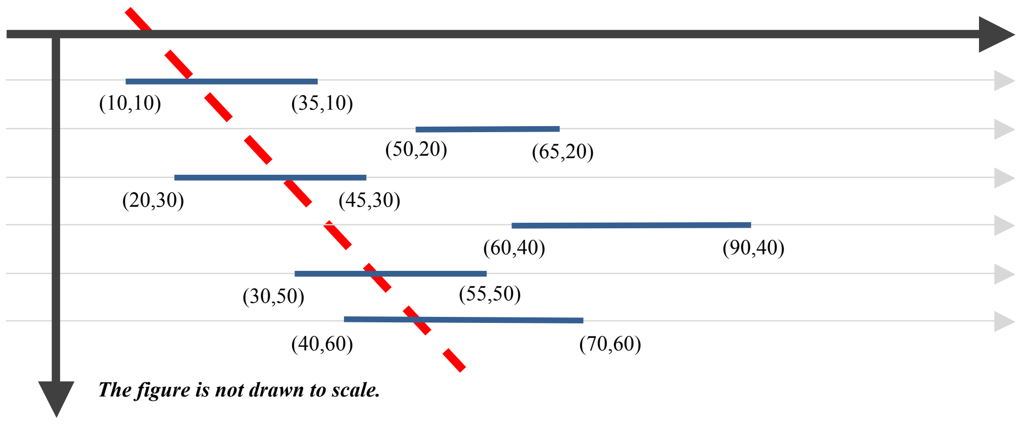
Output

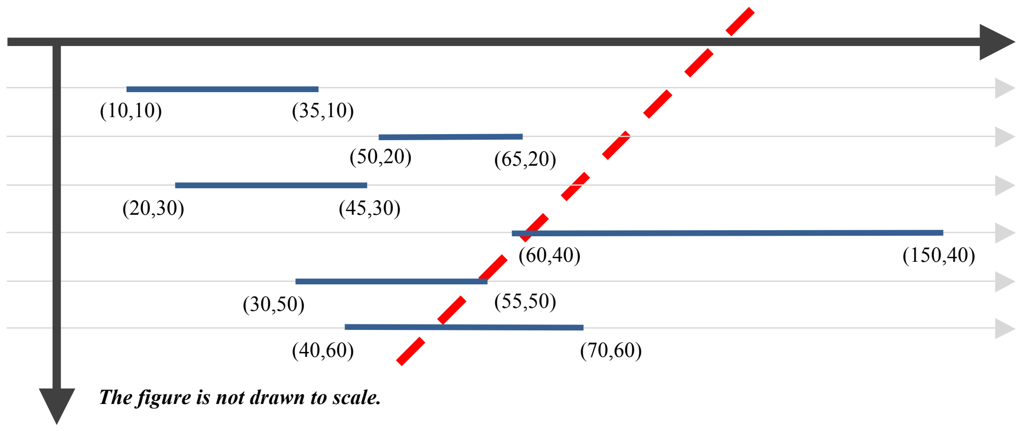
For each test case, there is one line of output. If the braces of input expression is to be a matched pair, print "**YES**", otherwise print "**NO**".

Sample Input	Sample Output
6 [a*(b+c)] a*[b*{c]+d]-e [a+{b*(c-d)+e}] [x*(y-z) x+(x-y)*z) x*(x-y)*[x-z*{x-a}]	YES NO YES NO NO YES

Problem H: Gold Mining

Myanmar, known as the golden land, is rich of natural resources. Among them, almost all of Myanmar people love gold. Gold is the most valuable thing among these people. They wear gold in many forms as jewelries. They also donate gold to their religious places such as pagodas and statues. Golden pagodas can be seen all around the country. They like to call their country as the golden land. There are many gold mines in the country. There are many steps in mining gold with traditional way and it needs a lot of resources (money, human power, electrical power). After doing researches, some engineers can know the layers under the ground in which gold particles are mixed with the soil and the width of these layers. A university project proposes an advanced technique for mining gold. Like oil drilling, they propose drilling gold layer and pulling up this gold deposit on to the ground. The gold well is drilled from the earth surface along a straight line and extracts the mixture of soil and gold particles from this well as shown in Figure. In this model, the amount of gold-soil mixture is equal to one third of the width of the layer.





Input

The first line of input contains a single integer n, which is the number of gold layers. In the following n lines, each line represents a single layer. Each line contains three integers, x_1, x_2 and y describing the layer's width such as (x_1, y) and (x_2, y). It is sure that each layer does not interest with others.

Output

Display the maximum amount of gold-soil mixture that can be extracted by a single gold well. (If the answer contains more than two decimal points, just two decimal points are required).

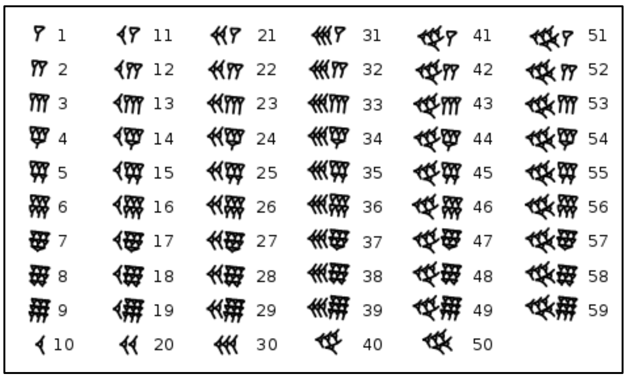
Hint for Decimal Format: System.out.printf("%.2f\n",decimal_value);

Sample Input	Sample Output

3
6
10 35 10
50 65 20
20 45 30
60 90 40
30 55 50
40 70 60
6
10 35 10
50 65 20
20 45 30
60 150 40
30 55 50
40 70 60
3
-30 10 10
-40 -25 20
-30 -10 30
	35.00
	48.33
	25.00

Problem I: Babylonian Numbers

More than 4000 years ago, the ancient Babylonians used a numerical system that is well known for being the first positional numerical system. In such a system, a non-negative integer is represented by a sequence of digits such that the value of a digit depends both on itself and on its position within the sequence. The Babylonians used a base-60 system known as sexagesimal (not unlike our own base-10 decimal system) where each 'digit' can take values between 1 and 59, inclusive. (The Babylonians didn't have a digit to represent 0; instead they would just leave the digit position empty.) According to the Egyptians at the time, the Babylonians carved their equations in solid clay, which allows us to read them thousands of years later.



History Professor, Dr. R, has asked you to decrypt the Babylonian numbers he has found during a recent excavation and to convert them into our own decimal numerical system. This is why we come to you. You now need to write a program that takes the sexagesimal notation and converts it to decimal.

Fortunately, each clay tablet discovered by Dr. R contains numbers in a clean format where digits corresponding to consecutive powers of 60 are separated by commas, with the most significant digit on the left. For instance, one tablet contains the Babylonian number 1, 24, 9 in sexagesimal, which converts to 5049 in decimal, since $1*60^2 + 24*60^1 + 9*60^0 = 5049$.

Input

The first line contains an integer N ($1 \le N \le 20$), the number of test cases to follow. Each of the following N lines represents a single tablet containing a number in sexagesimal format. This number is a non-empty sequence of digits separated by commas. Nonzero digits can be any integers between 1 and 59 (inclusive). The sexagesimal number does not begin with a comma, contains at least one nonzero digit, and consists of D digits in total, where $1 \le D \le 8$.

Output

For each test case, output a line containing the decimal representation of the sexagesimal number.

Sample Input	Sample Output
3	41
41,	5049
1,24,9	216000
1,,,	

Problem J: Need for Speed

Thida is a student and she drives a typical student car: it is old, slow, rusty, and falling apart. Recently, the needle on the speedometer fell off. She glued it back on, but she might have placed it at the wrong angle. Thus, when the speedometer reads s, her true speed is s + c, where c is an unknown constant (possibly negative).

Thida made a careful record of a recent journey and wants to use this to compute c. The journey consisted of n segments. In the i^{th} segment she traveled a distance of d_i and the speedometer read s_i for the entire segment. This whole journey took time t. Help Thida by computing c.

Note that while Thida's speedometer might have negative readings, her true speed was greater than zero for each segment of the journey.

Input

The first line of input is the number of test cases. In each test case, the first line contains two integers n (1 <= n <= 1000), the number of sections in Thida's journey, and t (1 <= t <= 106), the total time. This is followed by n lines, each describing one segment of Thida's journey. The i^{th} of these lines contains two integers d_i (1 <= d_i <= 1000) and s_i ($|s_i|$ <= 1000), the distance and speedometer reading for the i^{th} segment of the journey. Time is specified in hours, distance in miles, and speed in miles per hour.

Output

Display the constant c in miles per hour.

Hint for Output Decimal Format: System.out.printf("%.2f\n",decimal_value);

Sample Input	Sample Output
2	3.00
3 5	-0.51
4 -1	
4 0	
10 3	
4 10	
5 3	
2 2	
3 6	
3 1	