The 14th Annual **Guangdong** **C**ollegiate **P**rogramming **C**ontest

Sponsored by

Computer Academy of Guangdong

School of Software & School of Mobile Information Engineering, Sun Yat-sen U

Beijing Institute Of Technology, Zhuhai
Beijing Normal University, Zhuhai
Dongguan University Of Technology
Guangdong Pharmaceutical University
Guangdong Polytechnic Normal University
Guangdong University Of Education
Guangdong University of Finance & Economics
Guangdong University Of Foreign Studies
Guangdong University of Technology
Guangzhou College of South China University of Technology
Guangzhou University
Hanshan Normal University
Hong Kong University of Science and Technology
Huizhou University
Jinan University
Lingnan Normal University
Nanyang Institute of Technology
Neusoft Institute,Guangdong
Shanghai University
ShanTou University
Shenzhen University
South China Agricultural University
South China Institute of Software Engineering.GU
South China Normal University
South China University of Technology
Sun Yatsen University
University Of Electronic Science And Technology Of China，Zhongshan Institute
WuYi Uniwersity
Zhaoqing University
Zhongkai University of Agriculture and Engineering

Sun Yat-sen Univ. East Campus & **Zhuhai Campus, Guangzhou, Guangdong,**
May 8, 2016

The problem set will contain 11 problems on 12 numbered pages. Please inform staffs immediately if something is missing from your problem set.

Problem A. ABCD

Input file: stdin
Output file: stdout

$ABCD$ is a convex quadrilateral(polygon with four edges), while AC and BD are diagonals of it. We are interested in whether $ABCD$ is a cyclic quadrilateral(the four vertices lie on a circumscribed circle) or not, could you tell me?

Input

The first line contains an integer T denoting the number of test cases.

In the following T lines, each line contains a case. In each case, there is 6 integers, lengths of AB, CD, AD, BC, AC, BD. It's guaranteed that input form a convex quadrilateral.

All number inputed are integers in [1, 10000].

Output

For each test case, output Case #t:, to represent this is t-th case. And then output Yes if ABCD is a cyclic quadrilateral, otherwise output No instead.

No blank line between two consecutive test case.

Example

stdin	stdout
2 5 5 5 5 6 8 3 3 4 4 5 5	Case #1: No Case #2: Yes

Problem B. Bob's magical number

Input file: stdin
Output file: stdout

Three is a maigic number. Three is the first odd prime number; three is the first Mersenne prime; etc.

Bob loves three! He wants to create a graph that statisfies:

1. The number of vertices in the graph is n
2. The degree of each vertex in the graph is 3 (The degree of a vertex in a graph is the number of edges incident to it.)
3. The graph is a simple graph, an undirected graph which both multiple edge and self loops are disallowed.

Can you show him one of the graphs that statisfy the conditions?

Input

The first line is an integer T ($T \leq 20$), indicate the number of test case. Then T cases follow. One line for one positive integer $N(1 \leq N \leq 10^5)$.

Output

For each test case, output server lines.

If the graph exists, output $\frac{3n}{2} + 1$ lines.

The first line is "Case $\#X$: Yes", where X is the case number staring from 1.

Then $\frac{3n}{2}$ lines follow. Each line contains two positive integer u and $v(1 \leq u, v \leq n)$, denotes that there is an edge connect u and v. You can output any correct solution.

If the graph does not exists, output one line "Case $\#X$: No".

No blank line between two consecutive test case.

Example

stdin	stdout
2 2 4	Case #1: No Case #2: Yes 1 2 1 3 1 4 2 3 2 4 3 4

Problem C. Color

Input file: stdin
Output file: stdout

Danny is a primary school child who is madly cling to art. One day, Danny feels the wall around his house which is completely white is too tedious. He decides to reprint the wall. He found out that the wall has n blocks forming a circle, and he has k pigments in his collection.

Now Danny is thinking about how to reprint his wall with the k pigments. He would like to learn how many different choices he has, i.e., how many possible essentially different printing outcomes could be. For instance, 1000, 0100, 0010, 0001 is essentially the same.

Please notice that you can rotate, but not reflect, which means 110100 and 110010 are different.

Input

The first line, a integer $T(T \leq 10)$, indicates the number of test cases. Then follows T lines, each line with two integers $n, k(1 \leq k \leq n \leq 10^9)$, represents the number of blocks of the wall, k represents the number of pigments.

Output

T lines, each line a integer for the number of outcomes mod 1000000007.

Example

stdin	stdout
1 3 2	Case #1: 4

Problem D. Dilettante

Input file: stdin
Output file: stdout

Mike is a dilettante.

He wants to place $8n$ blocks of size $1 \times 1 \times 1 \times 2$ in a room of size $2 \times 2 \times 4 \times n$.

He thinks it's too boring to do that, and he wants to know the number of ways to place.

You can rotate the blocks into $2 \times 1 \times 1 \times 1$, $1 \times 2 \times 1 \times 1$ or $1 \times 1 \times 2 \times 1$.

He finds that the number of ways is too large to print, so he only need the number mod 1000000007.

Input

The first line contains an integer $t(t \leq 5)$, denoting the number of test cases.

For each test case, the first line contains a integer $n(1 \leq n \leq 10^9)$.

Output

For each test case, output `Case #t:`, to represent this is the t-th case. And then output the answer.

Example

stdin	stdout
3 1 2 3	Case #1: 121 Case #2: 155969 Case #3: 127027769

Problem E. Expectation

Input file: stdin
Output file: stdout

Mike invents a new game.

Mike has got a table with $2 \times n$ cells, and he will color a random rectangle(of all $\frac{3n(n+1)}{2}$ rectangles) in each step.

He wants to know the expectation number of the steps to color all cells.

Input

First line: an positive integer $t(t \leq 20)$ indicating the number of test cases. There are t cases following. In each case, the only line contains an integer $n(1 \leq n \leq 20)$.

Output

For each test case: output "Case #x: ans" (without quotes), where x is the number of test cases, and ans is the expectation.

Example

stdin	stdout
3 1 2 3	Case #1: 2.000000 Case #2: 3.928571 Case #3: 5.642851

Problem F. Fee

Input file: stdin
Output file: stdout

Students in X university got their summer internship payment, they are unhappy because the amount of money they get is different(as they love each other so much), and they decide to transfer money between each other to make sure the finally every one get the same amount of money. As the bank is evil, it takes away k% charge for each transfer. Of course the students want to keep as much as possible money, could you help them calculate how much money they could keep?

Input

The first line, a integer $T(T \leq 10)$, indicates the number of test cases. For each test case, then follows 1 line with a integer n and a float $k(0 \leq k \leq 1), n(n \leq 100000)$ represents the number of students, k represents the charge rate the bank takes, then follows n lines, each line representing the initial amount of money(≤ 10000) for a student.

Output

T lines, each line represents the result for each testcase(keep 6 decimal digits).

Example

stdin	stdout
1 2 0.2 10 20	Case #1: 14.444444

Problem G. Game

Input file: stdin
Output file: stdout

Alice and Bob play the following number guessing game: Both think of a number from $\{1, 2, \ldots, n\}$, which they reveal simultaneously. If Alice correctly guesses Bob's number she receives 1 dollar from Bob. If Alice guesses x when Bob guesses $x + 1$ then Alice gives 1 dollar to Bob. In all the other cases Alice and Bob do not exchange money.

Bob is aware of the fact, that Alice knows whatever he is thinking, so he always loses money. Therefore he decides to choose numbers with an RNG (random number generator). The details are following.

1. Bob can decide a distribution of each number.
2. Bob can use the RNG to choose numbers according to the distribution, and None knows what the RNG outputs.
3. Alice can also know Bob's distribution, but not the output of the RNG.
4. Then Alice will choose a number, that maximize her income.

Now, Bob wants to minimize the income of Alice, and he wants to know the minimum income of Alice.

Input

The first line contains an integer $t(t \leq 100)$, denoting the number of test cases.

For each test case, the only line contains one number $n(1 \leq n \leq 1000)$.

Output

For each test case, output Case #t: to represent the t-th case, and then output the minimum income of Alice.

Example

stdin	stdout
2 1 2	Case #1: 1.000000 Case #2: 0.333333

Problem H. Heresy

Input file: stdin
Output file: stdout

Mike learned greatest common divisor and encountered a new problem these days. The problem is

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\gcd(i,j).$$

He thinks it is too easy, and he adds something in it.

$$\sum_{i=1}^{n}\sum_{j=1}^{m}i^2j^2\gcd(i,j)$$

He wants you to help him solve this problem.

He finds that the answer is too large to print, so he only need the answer modulo 1000000007.

Input

First line: a positive integer $t(t \leq 1000)$ indicating the number of test cases. There are t cases following. In each case, the first line contains two positive integers $n, m(1 \leq n, m \leq 10^6)$.

Output

For each test case: output "Case #x: ans" (without quotes), where x is the number of the cases, and ans is the answer.

Example

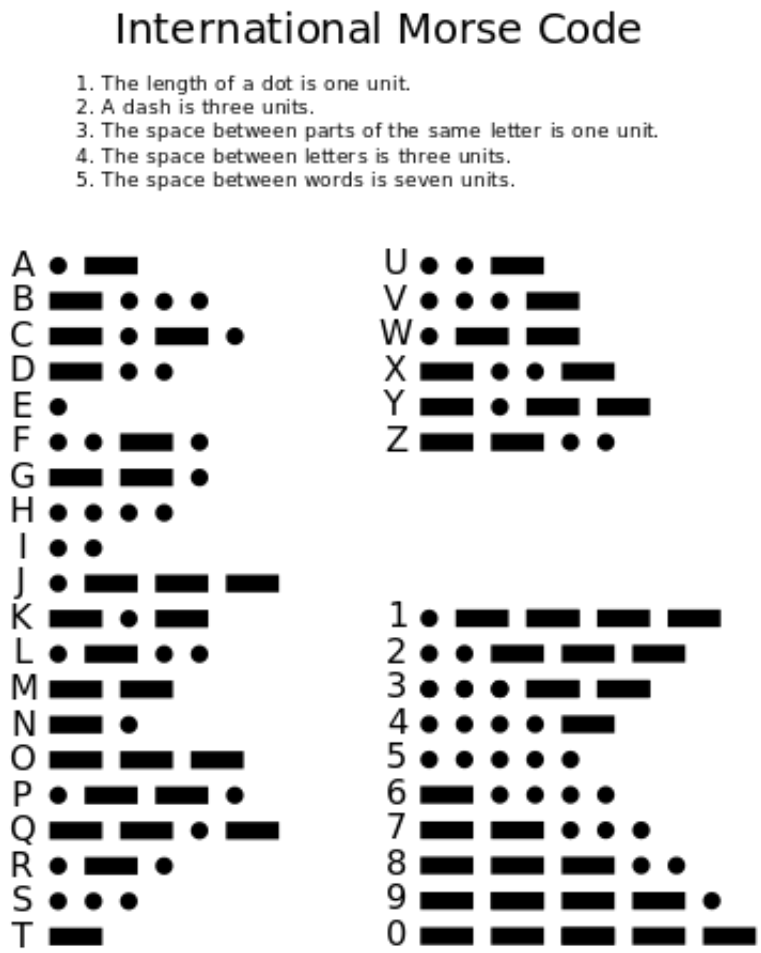
stdin	stdout
3 2 2 3 3 100 100	Case #1: 41 Case #2: 374 Case #3: 671450398

Problem I. International Morse Code

Input file: stdin
Output file: stdout

We are “on the air”.

Morse code is a method of transmitting text information as a series of on-off tones, lights, or clicks that can be directly understood by a skilled listener or observer without special equipment.



1. The length of a dot is 1 unit.
2. The length of a dash is 3 units.
3. The space between parts of the same letter is 1 unit.
4. The space between letters is 3 units.
5. The space between words is 7 units.

You need to convert a sequence of word into signal on and signal off. (NOT DOT”.” AND DASH”-”)

For example, we need to convert ”1 eE”. There are two steps.

1. Convert “`1 eE`” into “`.---- .  .`”
2. Convert “`.---- .  .`” into “`=.===.===.===.===.......=...=`”

Input

First line: an positive integer $0 \leq t \leq 20$ indicating the number of cases. Next t lines: each line has a positive integer $n \leq 20$, which is the number of words. After the number n, there are n words consisting of letters and numbers. The length of each word is no more than 20.

Output

For each test case: output "Case #k: ans" (without quotes), where k is the number of the test cases, and *ans* is a string consisting of "=" and "-".

Example

stdin	stdout
2 2 1 eE 1 SoS	Case #1: =.===.===.===.===.......=...= Case #2: =.=.=...===.===.===...=.=.=

Problem J. Just another binary tree

Input file: stdin
Output file: stdout

Mike is familiar with binary trees.

Mike likes to store binary trees into an array, from 1 to n, which means

1. 1 is the root of the tree.
2. i's left child is $2i$.
3. i's right child is $2i + 1$.

Mike is not satisfied with this, he wants to know the inorder traversal of the tree.

For example, $n = 9$, the inorder traversal is "8 4 9 2 5 1 6 3 7".

Mike only wants to know what the x-th number of inorder traversal is.

The inorder traversal index from 1, which means if $n = 8, x = 1$, the answer is 8.

Mike thinks it's too slow to calculate the whole inorder traversal, and he needs your help.

Input

The first line contains an integer $t(t \leq 100)$ denoting the number of test cases.

In the following T lines, each line contains a case. In each case, there is 2 integers $x, n(1 \leq x \leq n \leq 10^9)$. You need to output the x-th number in the inorder traversal of a binary tree with size n.

Output

For each test case, output `Case #t:`, to represent this is t-th case. And then output the answer.

Example

stdin	stdout
5 100 10 100 20 100 30 100 40 100 50	Case #1: 34 Case #2: 18 Case #3: 39 Case #4: 10 Case #5: 44

Problem K. Kay's function

Input file: stdin
Output file: stdout

Kay has a function $f[N]$.

1. $f[0] = a$
2. $f[1] = b$
3. For every $n > 1$, $f[n] = f[n-1] xor f[n-2]$

Given a number N, output the value of $f[N]$.

Input

The first line is an integer T ($T \leq 20$), indicate the number of test case. Then T cases follow.

One line for three number a, b, $N(0 \leq a \leq 10^6, 0 \leq b \leq 10^6, 0 \leq N \leq 10^9)$.

Output

For each test case, output "Case #X: Y" in a single line (without quotes), where X is the case number staring from 1, and Y is the answer. There is a space between colon and Y.

No blank line between two consecutive test case.

Example

stdin	stdout
2 1 2 0 1 2 1	Case #1: 1 Case #2: 2