Problem A. Tobby's Ancestors

Source file name:	ancestors.c, ancestors.cpp, ancestors.java, ancestors.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Sebastián Gómez - UTP Colombia`

Tobby, the small and cute dog, wants to prove to the rest of the world he is a descendent of Tutantobby, the great dog pharaoh. As Tutantobby was mummified, his internal organs were dried and separated in different bottles. To prove that Tobby is a descendent of Tutantobby, the DNA of some of these organs must be extracted and compared against Tobby's DNA. This DNA comparison is not a problem for nowadays science, but extracting the DNA from a 5000 years old mummified organ is a real challenge.

The DNA, as you might probably know, is represented as a sequence of letters `A`, `G`, `C` and `T`. All Tobby can expect is to obtain fragments of Tutantobby's DNA corrupted by the years. Tobby requires to assemble Tuntantobby DNA fragments, to do that Tobby takes two DNA fragments and computes the best way to assemble them as in the following example. Lets say Tobby has the two following Tutantobby DNA segments:

- `GATTACCA`
- `TACAACAG`

Note the following alignment and the resulting string:

```
GATTACCA
   TACAACAG
------------
GATTACXACAG
```

The score of this alignment is 4, since there are 4 characters in this alignment that match. The resulting assembled DNA would then be `GATTACXACAG`, note that the resulting string might have an X letter representing that Tobby can't tell what that letter would be. Now Tobby wants a program that given only two sequences, outputs the assembled DNA sequence for the alignment with the maximum score. If there are two alignments that produce the same score output the one where the first string is more to the left with respect to the second. For example if the first string is `ATTG` and the second is `GCCA`, the alignment `ATTGCCA` is preferable over `GCCATTG` since both have a score of 1.

Input

The input consists of several test cases. Each test case contains two strings S_1 and S_2 that indicate the two DNA fragments extracted from Tutantobby. These two strings will only contain the uppercase letters `A`, `G`, `T` and `C`. After each test case there will be a blank line.

- $1 \leq |S_1|,\ |S_2| \leq 10^5$

Output

For each test case output two lines. The first one with the best alignment score, and on the second line output the respective assembled sequence as explained in the problem statement. If the best score is 0, print on the second line the string `No matches`. There should be a blank line after each test case.

Example

Input	Output
GATTACCA TACAACAG AAAA GGGG ATTG GCCA	4 GATTACXACAG 0 No matches 1 ATTGCCA

Use fast I/O methods

Problem B. Tobby and Array

Source file name:	tobbyarray.c, tobbyarray.cpp, tobbyarray.java, tobbyarray.py
Input:	Standard
Output:	Standard
Author(s):	Jhon Jimenez, Manuel Pineda & Santiago Gutierrez - UTP Colombia

As it is known, Tobby loves arrays and queries (he also hates long statements :D). One day Tobby came up with the following: there is an array of integers and multiple queries. For each query, Tobby wants to know the value of the $k-th$ position in the subarray $[l,\ r]$ $(r \geq l)$ $(1 \leq k \leq r-l+1)$, **if the subarray [l, r] was sorted in non-decreasing order**.

Input

The input has several test cases. The first line contains n $(1 \leq n \leq 10^6)$ and $q(1 \leq q \leq 10^6)$, the length of the array and the number of queries respectively. The next line contains n integers $a_i(1 \leq a_i \leq 10^9)$. Then q lines follow, each line containing a query with three integers l, r and k $(1 \leq l,\ r \leq n)$.

Output

For each query print the answer in a single line (**Look at the samples**).

Example

Input	Output
4 3 1 3 4 3 1 2 2 2 4 1 1 4 4 8 3 4 7 8 5 3 6 1 2 4 5 1 1 8 3 3 5 3 10 10 8 6 2 1 7 3 10 9 5 4 1 8 3 7 7 1 7 8 1 9 9 1 2 10 9 2 7 2 5 7 1 10 10 1 9 10 2 7 10 4	3 3 4 3 3 8 3 10 9 5 10 2 3 4 5 10

Use fast I/O methods

Explanation

For the first sample.

indexes: 1 2 3 4
$array = \{1,\ 3,\ 4,\ 3\}$

For first query $[1,\ 2]$ we have the subarray $\{1,\ 3\}$, after sorting we have $\{1,\ \overline{3}\}$, so the value in the $2-th$ position is 3.

For second query $[2,\ 4]$ we have the subarray $\{3,\ 4,\ 3\}$, after sorting we have $\{\overline{3},\ 3,\ 4\}$, so the value in the $1-th$ position is 3.

For third query $[1,\ 4]$ we have the subarray $\{1,\ 3,\ 4,\ 3\}$, after sorting we have $\{1,\ 3,\ 3,\ \overline{4}\}$, so the value in the $4-th$ position is 4.

Problem C. Counting Edges and Graphs

Source file name:	counting.c, counting.cpp, counting.java, counting.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Yonny Mondelo - UCI Cuba`

You must construct a directed graph with exactly N nodes conveniently numbered between 1 and N. But this is not an ordinary graph; this is a special graph for which each node must have K or less directed edges going to their proper divisors (nodes numbered with those divisors values). If some node has only $k \leq K$ proper divisors, then that node will have exactly k edges to those k divisors. Also note that if some node has $M > K$ proper divisors then that node will have exactly K edges to some group of K proper divisors of the M available. A proper divisor of some integer number P is any divisor of P, excluding P itself. For example, 1, 2 and 3 are proper divisors of 6; but 6 is not a proper divisor of itself.

Given the value for K and the number of nodes N in the graph you must construct, can you find the number of edges on it after it is constructed? Also, can you determine the number of possible graphs which can be constructed fulfilling the above specifications?

Input

The first line contains an integer T ($4*10^5 \leq T \leq 5*10^5$) representing the number of graphs to construct. The next T lines contain two integer numbers N and K ($1 \leq N,\ K \leq 5*10^3$) representing the number of nodes in the graph and the maximum number of edges per node. Scenarios must be answered in the same order of the graphs given in the input.

Output

For each graph you must print a line containing two integer numbers representing the number of edges of the graph and the number of possible graphs which can be constructed, respectively. As those values could be large, print them modulo 1000000007 ($10^9 + 7$).

Example

Input	Output
3 4 2 5 3 6 2	4 1 5 1 7 3

Use fast I/O methods

Problem D. DPA Numbers I

Source file name:	dpa01.c, dpa01.cpp, dpa01.java, dpa01.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

In number theory, a positive integer belongs to one and only one of the following categories: Deficient, Perfect or Abundant (DPA).

To decide the category of a positive integer n, first you have to calculate the sum of all its proper positive divisors. If the result is less than n then n is a deficient number, if the result is equal to n then n is a perfect number and if the result is greater than n then n is an abundant number. Remember that the proper divisors of n don't include n itself.

For example, the proper divisors of the number 8 are 1, 2 and 4 which sum 7. Since $7 < 8$ therefore 8 is a deficient number. The proper divisors of the number 6 are 1, 2 and 3 which sum 6. Since $6 = 6$ therefore 6 is a perfect number. The proper divisors of the number 18 are 1, 2, 3, 6 and 9 which sum 21. Since $21 > 18$ therefore 18 is an abundant number.

The task is to choose the category of a positive integer n as a deficient, perfect or abundant number.

Input

Input begins with an integer t ($400 \leq t \leq 500$), the number of test cases, followed by t lines, each line containing an integer n ($2 \leq n \leq 10^3$).

Output

For each test case, you should print a single line containing the word `deficient`, `perfect` or `abundant` that representing the category of the number n.

Example

Input	Output
10 5 6 16 18 21 28 29 30 40 43	deficient perfect deficient abundant deficient perfect deficient abundant abundant deficient

Problem E. Tobby and the quaseEquals strings

Source file name:	tobbystrings.c, tobbystrings.cpp, tobbystrings.java, tobbystrings.py
Input:	Standard
Output:	Standard
Author(s):	Carlos Arias - UTP Colombia

Tobby always enjoys playing with strings, and now he brings to you a nice problem with them. Of course, since Tobby is a lazy dog, he has not solved it yet and hopes that you can solve it for him.

Tobby got a set of strings S of size N (where every string has the same length L). He also has Q queries.

For each query a string A of size L is given and Tobby wants to know how many strings in S are quaseEquals to A for every i $(1 \leq i \leq L)$.

Two strings are quaseEquals to one another for an index i if they are equal after deleting the i-th character from both strings.

Input

The input consists of several test cases, read until the end of file (EOF). In the first line of each test case there are three integers: N, Q, L $(1 \leq N,\ Q,\ L \leq 10^5)$. The next N lines contain the strings in S, all of length L. Finally Q strings of length L are given, those are the queries. It is guaranteed that $(1 \leq N * L \leq 100000)$ and $(1 \leq Q * L \leq 100000)$ and that all strings in the input contain only english lowercase letters (a-z).

Output

For each query print the number of strings in S that are quaseEquals to the string in the query for every position $1 \leq i \leq L$.

Example

Input	Output
3 1 3 aab aba aaa aaa 6 3 6 tobyis having funwhi leyoua resolv ingitD tobbis cobyis cobbis	5 1 1 0

Use fast I/O methods

Explanation

For the first sample, if the character $i = 1$ is removed, then $S = \{ab,\ ba,\ aa\}$ and $A = \{aa\}$ and we got 1 pair of quaseEquals strings. If the character $i = 2$ is removed, then $S = \{ab,\ aa,\ aa\}$ and $A = \{aa\}$ and we got 2 pairs of quaseEquals strings. If the character $i = 3$ is removed, then $S = \{aa,\ ab,\ aa\}$ and $A = \{aa\}$ and we got 2 pairs of quaseEquals strings, so our answer is $1 + 2 + 2 = 5$.

Problem F. Felipe and the Sequence

Source file name:	sequence.c, sequence.cpp, sequence.java, sequence.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

On February 19, 2017, Red Matemática proposed the following mathematical challenge on its twitter account (@redmatematicant): "Felipe, how many terms of the next sequence of numbers must be added to make the result equal to 200?"

$$\frac{1}{\sqrt{1}+\sqrt{2}}+\frac{1}{\sqrt{2}+\sqrt{3}}+\frac{1}{\sqrt{3}+\sqrt{4}}+\frac{1}{\sqrt{4}+\sqrt{5}}+\cdots=200$$

Using this interesting puzzle as our starting point, the problem you are asked to solve now is: Given a positive integer S ($1 \leq S \leq 10^9$) representing the result obtained for the sum of the terms in the sequence, find out the number n that represents the total number of terms in the sequence to sum up.

$$\frac{1}{\sqrt{1}+\sqrt{2}}+\frac{1}{\sqrt{2}+\sqrt{3}}+\frac{1}{\sqrt{3}+\sqrt{4}}+\frac{1}{\sqrt{4}+\sqrt{5}}+\cdots+\frac{1}{\sqrt{n}+\sqrt{n+1}}=S$$

Input

Input begins with an integer t ($4 * 10^5 \leq t \leq 5 * 10^5$), the number of test cases, followed by t lines, each containing an integer S ($1 \leq S \leq 10^9$).

Output

For each test case, your program must print one positive integer denoting the number n that represents the total number of terms in the sequence to sum up.

Example

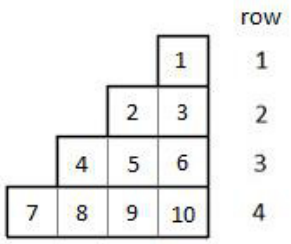
Input	Output
3 200 725 1333	40400 527075 1779555

Use fast I/O methods

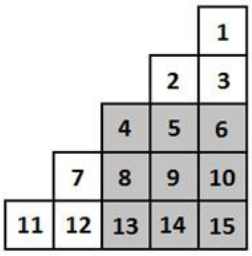
Problem G. Rectangular Sum

Source file name:	rectangular.c, rectangular.cpp, rectangular.java, rectangular.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Gabriel Gutiérrez Tamayo - UTP Colombia`

In this challenge, you are given a triangular board of n rows. The first row has one block, and the following rows have a block more than the previous row. All the blocks have the same size and are numbered as follows:



First, you must find the biggest rectangular area inside the triangular board, and then calculate the value of S which corresponds to the sum of the values belonging to the area found. If there are several areas with the same size, choose the area that maximizes the value of S. For example, when $n = 5$:



The maximum rectangular area is (3×3), which is represented in the previous image.

$S = 4 + 5 + 6 + 8 + 9 + 10 + 13 + 14 + 15 = 84$

Remember that the area of a rectangle is the multiplication of the two sides of the rectangle

Input

The first line of input contains an integer t ($10^4 \leq t \leq 10^5$) indicating the number of test cases that follow, one for line. Each test case contains a positive integer n ($1 \leq n \leq 10^{11}$) indicating the number of rows.

Output

For each test case, you should print a line containing `Case #x: y`, where `x` is the test case number (starting from 1) and `y` is the sum obtained. Note that this value is very large, so print the result modulo $10^9 + 7$.

Example

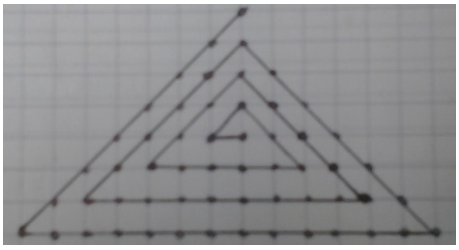
Input	Output
6 1 2 3 4 5 14	Case #1: 1 Case #2: 5 Case #3: 16 Case #4: 42 Case #5: 84 Case #6: 3612

Use fast I/O methods

Problem H. Humbertov and the Triangular Spiral

Source file name:	triangular.c, triangular.cpp, triangular.java, triangular.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

Recently the professor *Humbertov Moralov* was sick, he had a fever and when he went to bed, he began to have a delirious dream. In the dream he draw over and over again a triangular spiral that began in the origin of the cartesian plane (coordinate $(0, 0)$) and the triangular spiral got bigger every time linking integer coordinates in the cartesian plane. For clarity, the triangular spiral is presented below:



The dream was so disturbing and it was repeated so many times, that when *Moralov* woke up, he remembered perfectly the triangular spiral, and for this reason he drew the previous graphic.

In the dream *Moralov* was disturbed and intrigued because he didn't know if all the integer coordinates could be reached at some point in the triangular spiral and, if that was the case, he also didn't know what would be the coordinate in the cartesian plane of the n-th point that is reached when drawing the triangular spiral. The first doubt was immediately resolved when the professor did the graphic ... all the points (integer coordinates) of the cartesian plane are eventually reached by the triangular spirals! Now the professor *Moralov* needs your help to indicate the coordinate in the cartesian plane of the n-th point that is reached when drawing the triangular spiral.

Input

Input begins with an integer t ($4 \times 10^5 \leq t \leq 5 \times 10^5$), the number of test cases, followed by t lines, each line contains an integer n ($1 \leq n \leq 10^{12}$).

Output

For each test case, you should print a single line containing two integers, separated by a space, denoting the coordinates x and y in the Cartesian coordinate system of point n in the triangular spiral.

Example

Input	Output
15 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	0 0 -1 0 0 1 1 0 2 -1 1 -1 0 -1 -1 -1 -2 -1 -3 -1 -2 0 -1 1 0 2 1 1 2 0

Use fast I/O methods

Problem I. Rockabye Tobby

Source file name:	rockabye.c, rockabye.cpp, rockabye.java, rockabye.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Yeferson Gaitan Gomez - UTP Colombia`

"Rockabye baby, don't you cry".

Tobby is very good at catching the ball, he loves that game so much, that one day he decided to go out and play, even though it was raining. He played for a long time and in addition to catching the ball many times, he also got a cold, poor Tobby. That is why now his mother will take care of him, Big doggie momma, singing that beautiful lullaby (rockabye) and giving him the medications in the moments that must be taken.

In the medical prescription sent by the doctor, he specifies the name of the medications and how often they should be taken. The doctor told him that if he takes the medications constantly, he will be relieved after taking k medicines. Tobby does not like being sick (in fact no one likes to be), so he promises his mother to be constant with the drugs, that is why he now wants to know what are the first k drugs that he has to take to feel better. Can you help him?

Input

Input begins with a line containing an integer t, the number of test cases.

For each test case, the medical prescription is written as follows:

Each test case begins with a line containing two integers, n ($1 \leq n \leq 3 * 10^3$) and k ($1 \leq k \leq 10^4$), indicating the number of medications sent by the doctor and the minimum number of medicines Tobby must take to feel better.

The following n lines will be of the form, `name frecuency` ($1 \leq |$ `name` $| \leq 10$, $1 \leq$ `frecuency` $\leq 3 * 10^3$), indicating the name of the medication and how often it should be taken.

The medicines are listed according to their degree of priority, i.e. the first one will be the most important drug and the last one, the least important.

Output

For each test case, the output must have k lines, each of the form, t m, indicating that in the moment t Tobby must take the drug m.

If there are two or more drugs that must be given at the same time t, they should be printed according to their priority.

Example

Input	Output
1 2 5 Acetaminophen 20 Loratadine 30	20 Acetaminophen 30 Loratadine 40 Acetaminophen 60 Acetaminophen 60 Loratadine

Use fast I/O methods

Problem J. Tobby Primes

Source file name:	tobbyprimes.c, tobbyprimes.cpp, tobbyprimes.java, tobbyprimes.py
Input:	Standard
Output:	Standard
Author(s):	Santiago Gutierrez - UTP Colombia & Google

Tobby the boston terrier is trying to escape from the pyramid of the egyptian pharaoh. To escape, Tobby has to solve a riddle that asks him to factor a list of large integers.

The fastest way Tobby knows of factoring integers is iterating over all primes up to the square root of the target number, checking for prime factors. The problem is that in this case the number of primes to check would be very large, so he cannot use that method. Can you help him solve the riddle?

Input

Input begins with an integer t ($1 \leq t \leq 100$), the number of integers to factor, followed by t lines, each line contains an integer n ($2 \leq n \leq 2^{63} - 1$).

Output

For each test case, you should print a single line containing the prime factors of n sorted in increasing order. Note that in case of repeated prime factors, such factors have to be printed several times.

Example

Input	Output
3 2 91 40	2 7 13 2 2 2 5

Problem K. Tobby and the Skeletons

Source file name:	tobbyskeletons.c, tobbyskeletons.cpp, tobbyskeletons.java, tobbyskeletons.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Diego Agudelo-España - UTP Colombia`

There is nothing that Tobby, as a dog, enjoys more than bones. That's why he has been studying the skeleton of certain species to figure out how much fun he could have with their bones.

Tobby models a skeleton as a weighted tree whose edges represent the bones and their weights denote the lengths of the bones. Thus, the nodes are simply the joints connecting different bones. For Tobby it is quite hard to play with an entire skeleton, so he prefers to take a chain of connected bones instead, or, in other words, a simple path connecting two nodes in the skeleton tree. Moreover, since Tobby is a greedy dog, his happiness with a particular chain of bones doesn't depend only on the chain's size but on the length of the largest bone present in the chain as well, and here is where Tobby needs some help from you.

It turns out that even for members of the same species there are variations on the length of the bones that make up the skeleton (the skeleton itself keeps fixed across members). Therefore, Tobby decided to model the bone lengths (i.e. edge weights) as discrete uniform random variables. This means that the weight w_i associated with the $i-th$ edge takes integer values in the closed interval $[a_i,\ b_i]$.

Tobby has Q queries for you. Given a description of the tree and the weight random variables, for each query Tobby wants to know the expected value over the length of the largest bone present in a bones chain from joint x_q to join y_q. Formally, if $w_1,\ w_2,\ \ldots,\ w_s$ are the random variables denoting the edge's weights in the simple path from x_q to y_q, you are required to compute $E[max(w_1,\ w_2,\ \ldots,\ w_s)]$.

Input

The input contains multiple test cases. For each test case the first line contains an integer N ($2 \leq N \leq 50000$) denoting the number of nodes in the tree of bones. Each of the following $N-1$ lines describes an edge with 4 integers: $x_i,\ y_i,\ a_i,\ b_i$ ($x_i \neq y_i,\quad 1 \leq x_i,\ y_i \leq N,\quad 0 \leq a_i \leq b_i \leq 100$) where x_i and y_i represent two different nodes connected by the edge whose weight can take discrete values uniformly in $[a_i,\ b_i]$ (It's guaranteed that the given graph is a tree). Next, the number of queries Q is given and then, Q ($1 \leq Q \leq 100000$) more lines follow describing each query with two different nodes x_q and y_q ($x_q \neq y_q,\quad 1 \leq x_q,\ y_q \leq N$) which represent both ends in a bones chain of interest for Tobby. The input specification ends with EOF.

Output

For each test case there must be Q output lines answering the Q test case queries. For each of these queries print in a single line the expected value of the largest edge weight in the simple path connecting the queried nodes. Your answer will be considered correct if the absolute difference with the jury's answer is less than $1e^{-5}$.

Example

Input	Output
6 1 2 0 50 2 3 30 40 2 4 10 80 4 5 50 90 4 6 95 100 3 1 6 3 5 2 4 2 1 2 0 100 1 2 1	97.5 71.70388182755067 45.000000000000014 49.999999999999986

Use fast I/O methods

Problem L. Rotations

Source file name:	rotations.c, rotations.cpp, rotations.java, rotations.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

Humbertov Moralov in his student days, enrolled in the Systems Engineering program at "University of the Missing Hill" in *The Heaven's Branch Office* (Colombia, South America). He then attended a course of Assembly Language (in the first half of 1997).

The course was fantastic, with very interesting topics such as bit manipulation, right shifts, left shifts, rotations, masks and other bitwise operations (and, or, xor, not). And the best, in that course he worked interesting programming challenges. One of those programming challenges follows.

The chosen programming challenge is named "Rotations". By that time he used to work with unsigned integers of eight bits (a byte), and the challenge consisted of figuring out if a particular number n could generate all the eight numbers from 0 to 7 taking groups of consecutive bits of size 3.

For example, the number 226 has the binary representation 11100010 ($b_7b_6b_5b_4b_3b_2b_1b_0$), the eight sequences of consecutive three-bits that can be generated are the following:

- $b_2b_1b_0 = (010)_2 = 2$
- $b_3b_2b_1 = (001)_2 = 1$
- $b_4b_3b_2 = (000)_2 = 0$
- $b_5b_4b_3 = (100)_2 = 4$
- $b_6b_5b_4 = (110)_2 = 6$
- $b_7b_6b_5 = (111)_2 = 7$
- $b_0b_7b_6 = (011)_2 = 3$
- $b_1b_0b_7 = (101)_2 = 5$

20 years have passed. Since today computers are more powerful and faster, the professor *Humbertov Moralov* wants you to solve this programming challenge for unsigned integers of 32 bits (four bytes). You must validate if the number n generates all the 32 numbers from 0 to 31 with sequences of consecutive five-bits.

Input

Input begins with an integer t ($2 \times 10^5 \leq t \leq 3 \times 10^5$), the number of test cases, followed by t lines, each line containing an integer n ($0 \leq n \leq 4 \times 10^9$).

Output

For each test case, you should print a single line containing the word `yes` or `no` depending if the integer number n produces or not all the numbers from 0 to 31 with sequences of consecutive five-bits.

Example

Input	Output
15	no
65535	no
65259922	yes
81354525	no
112805325	no
122525196	yes
192052550	yes
225525450	no
299525510	yes
318353525	no
344152934	yes
502445252	yes
522595252	no
1296752550	yes
3999995011	no
4000000000	

Use fast I/O methods

Problem M. DPA Numbers II

Source file name:	dpa02.c, dpa02.cpp, dpa02.java, dpa02.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

In number theory, a positive integer belongs to one and only one of the following categories: Deficient, Perfect or Abundant (DPA).

To decide the category of a positive integer n, first you have to calculate the sum of all its proper positive divisors. If the result is less than n then n is a deficient number, if the result is equal to n then n is a perfect number and if the result is greater than n then n is an abundant number. Remember that the proper divisors of n don't include n itself.

For example, the proper divisors of the number 8 are 1, 2 and 4 which sum 7. Since $7 < 8$ therefore 8 is a deficient number. The proper divisors of the number 6 are 1, 2 and 3 which sum 6. Since $6 = 6$ therefore 6 is a perfect number. The proper divisors of the number 18 are 1, 2, 3, 6 and 9 which sum 21. Since $21 > 18$ therefore 18 is an abundant number.

The task is to choose the category of a positive integer n as a deficient, perfect or abundant number.

Input

Input begins with an integer t ($1000 \leq t \leq 1100$), the number of test cases, followed by t lines, each line containing an integer n ($2 \leq n \leq 10^{12}$).

Output

For each test case, you should print a single line containing the word `deficient`, `perfect` or `abundant` that representing the category of the number n.

Example

Input	Output
10	deficient
5	perfect
6	deficient
16	abundant
18	deficient
21	perfect
28	deficient
29	abundant
30	abundant
40	deficient
43	

Use fast I/O methods