# Problem A
## Integer Division
### Time limit: 1 second

Write a program to perform division of two integers, $x$ divided by $y$. The value of $y$ is less than the magic number

$$922337203685477579 = 2^{63}/10 - 1.$$

The value of $x$ can be very large, but less then 1000 decimal digits.

## Input Format

The input file may contain many positive integers, two integers $y$ and $x$ per line. The divisor $y$ comes before the dividend $x$ in the input line. The input data $y$ and $x$ are both decimal numbers, and they are separated by exactly 1 space in the input line. The value of $y$ is less then $2^{60}$. The value of $x$ is at most 1000 decimal digits.

The last line of the input file contains a “0”. It indicates the end of the test data.

## Output Format

For each pair of integers $x$ and $y$ in the input file compute the quotient $q$ and remainder $r$ for $x$ divided by $y$, so that $y = qx + r$, $0 \leq r < y$. If $y$ divides $x$ evenly ($r = 0$), then print the quotient $q$. Otherwise, print the quotient $q$ followed by the remainder $r$, with a space between them. Leading zeros in every output positive integer should not be printed.

## Sample Input

```
8 9
12 625
0
```

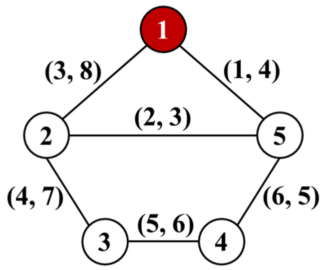
## Output for the Sample Input

```
1 1
3 2
```

# Problem B
## Propagation Route
### Time limit: 1 second

The epidemic of COVID-19 seriously affects the development of economis. The epidemiologist analyzed the route of infection. They found the route of infection is highly related to the distance and the frequency of people's movement between cities. The cities are connected by bidirectional roads, and the distance between two cities are the sum of the road lengths on a shortest route between them. If there is no road between two cities $a$ and $b$, then the frequency of people's movement between $a$ and $b$ is defined as infinity. It is known that for any two pairs of adjacent cities $(a, b)$ and $(c, d)$, either the "distance" between $a$ and $b$ is different from that between $c$ and $d$, or the "frequency of people's movement" between $a$ and $b$ is different from that between $c$ and $d$. The research report of epidmiologist analysis is as follows. Among the uninfected cities, the one with the highest frequency of people's movement to an infected city will be infected first. If there are more than one uninfected cities whose frequencies of people's movement are the highest, then the one which is closest to an infected city will be infected first.

For example, there are five cities and six roads between these cities (see the following figure). For each edge $e$, there is a pair $(dis, freq)$ denoting the distance and the frequency of people's movement between two cities at the endpoints of $e$. Assume the city #1 (red node) was the first infected city. Then the propagation route is #1, #2, #3, #4, #5.




In order to prevent the spread of the epidemic, the command center hopes that the Department of Information Technology can predict the possible route of infection via the distance and the frequency of people's movement between cities. Assume that only one city will be infected every day. Please write a program to predict the propagation route.

## Input File Format

There are more than one test cases in the input file. The first line of each test case contains three postive integers, $n, m, city$, where $n$ $(2 \leq n \leq 200)$ indicates the total number of cities, $m$ $(1 \leq m \leq 20,000)$ indicates the total number of roads between these $n$ cities, and $city$ $(1 \leq city \leq 200)$ indicates the first infected city. Each of the next $m$ lines contains four integers, $x, y, dis, freq$ ($1 \leq x, y, \leq 200, 1 \leq dis, freq \leq 20,000$), representing the distance between city $x$ and city $y$ is $dis$, and the frequency of people's movement is $freq$. The input is terminated by '`0 0 0`'.

# Output Format

For each test case, output the sequence of city numbers (consecutive members of the sequence are separated by a space) describing the predicted propagation route.

# Sample Input

```
5 6 1
1 2 3 8
2 3 4 7
3 4 5 6
4 5 6 5
5 1 1 4
5 2 2 3
4 5 3
1 2 1 2
2 3 2 3
3 4 3 3
4 1 4 2
1 3 5 2
0 0 0
```

# Output for the Sample Input

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

# Problem C
## Covid-19
### Time limit: 3 seconds

In December 2019, Covid-19 spread worldwide on the earth. An important parameter to the Covid-19 is $R_0$ (R naught). $R_0$ is the basic reproduction number that is used to measure the transmissibility of infectious agents. $R_0$ is relative to the duration of infectivity and the likelihood of transmission of infection per contact between a susceptible person and an infectious individual.

Assume the duration of infectivity to Covid-19 is two weeks. Once a person is infected, there will be $R_0$ persons infected after two weeks. Doctor Alice wants to calculate how many persons will be infected after $2N$ weeks. For simplification, we assume that there will be $1 + R_0 + R_0^2 + R_0^3 + \cdots + R_0^N$ after $2N$ weeks. Now, give you the value of $R_0$ and $N$, can you calculate the number? In order to check your answer easily, please modulo this number by a given prime number $D$.

For example, let $D = 5$, $R_0 = 2$, and $N = 5$. We calculate the infected number as $1 + R_0 + R_0^2 + R_0^3 + \cdots + R_0^N = 1 + 2 + 4 + 8 + 16 + 32 = 63$. So we have to output 3 because 63 mod $5 = 3$.

## Technical Specification

1. $1 \le D \le 2 \times 10^9$ is a prime number.
2. $1 \le R_0 \le 10^{1024}$
3. $1 \le N \le 10^{1024}$

## Input Format

The first line contains an integer $T (\le 60)$ representing the number of test cases. For each test case, there are 3 integers $D$, $R_0$, and $N$.

## Output Format

For each test case, please output the infected number modulo $D$.

## Sample Input

```
5
5 2 5
7 6 4
5 7 6
3 7 5
2 6 8
```

## Sample Output for the Sample Input

```
3
1
2
0
1
```

# Problem D
## Was it IN?
### Time limit: 1 second

The Autumn Festival has started, and those who stand inside the hot zone during the event can participate in the Festival Lottery and win a chance to stand on the podium!

The T-shaped hot zone consists of a horizontal strip and a vertical half-strip. You are given the coordinates of the strips and your current location, represented as a point in the plane. Your task is to determine whether or not you are in the hot zone!

Note that, points on the boundary of the strips are considered inside the hot zone.

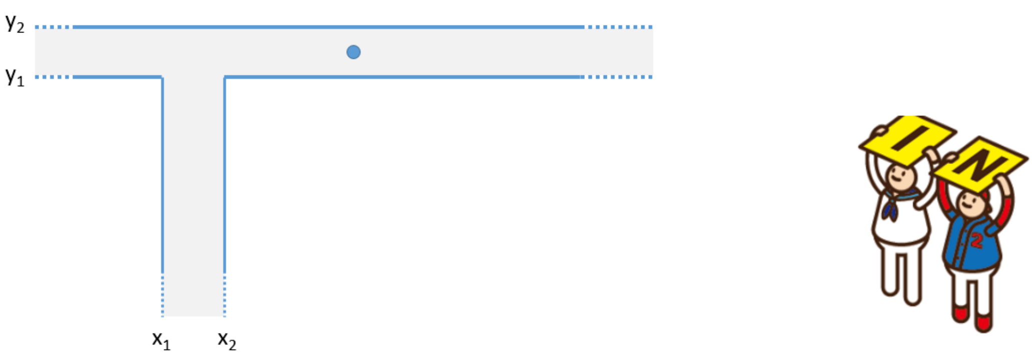



Figure 1: The left part of the figure shows an example with a hozontal strip that spans from $y_1 = 0$ to $y_2 = 2$ and a vertical half-strip that anchors at $y_1$ and spans from $x_1 = 4$ to $x_2 = 6$. The input point locates at $(7, 1)$. So your program should output "IN" in a line, as the right part of the figure shows.

## Input Format

There will be multiple test cases in the input file, separated by a blank line.

Each test case starts with four integers $y_1, y_2, x_1, x_2$, which are the y-coordinates of the lower-boundary and the upper-boundary of the horizontal strip and the x-coordinates of the vertical strip, respectively. The second line of the test case contains two integers $x$ and $y$, which are your current location.

A test case with $y_1 = y_2 = x_1 = x_2 = 0$ indicates the end of the input.

You may additionally assume the following.

- $y_1 \leq y_2$, $x_1 \leq x_2$.
- The absolute value of all coordinates does not exceed $10^9$.

## Output Format

For each test case, print "IN" in a line if you are in the hot zone. Otherwise, print "OUT" in a line.

## Sample Input

```
0 2 4 6
7 1

0 2 4 6
7 -1

0 0 0 0
```

## Output for the Sample Input

```
IN
OUT
```

# Problem E
## Fenced Area
### Time limit: 1 second

The NCPC park has many squared fenced areas of grasslands and ponds. Aerial image of a fenced area with two ponds on a 10x10 grid is shown in Fig. 1, in which the grassland is shown in green and the fence and ponds are shown in black. The image can be encoded by a run of grassland squares (green) and non-grass squares (black). For Fig. 1, the encoding is 12 6 3 1 6 1 2 1 1 1 4 1 2 1 3 2 1 4 3 1 2 6 4 2 3 7 3 3 14. This encoding indicates that, from top-left to bottom-right, the aerial image contains 12 green squares followed by 6 black squares and then 3 green squares, and then 1 black squares, ..., 3 black squares, and finally 14 green squares. Note that the run of green or black squares can continue onto the next row when it reaches end of a row.

Given encoding of an aerial image of a fenced area, determine the number of grass squares within this fenced area.

## Technical Specification:

1. The aerial image has at most 75 rows and 75 columns.
2. There is exactly one fenced area within each given aerial image. Fence thickness may vary at different places.
3. There can be 0 or more irregular shaped ponds within the fenced area. There are no ponds outside of the fenced area.

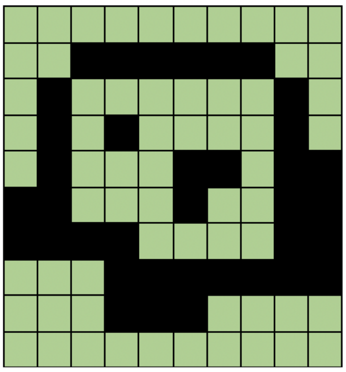


Figure 1: There are 24 grass squares within the fenced area.

## Input File Format

The first line contains an integer, indicating the number of test cases to follow. For each test case, there are two lines of integers: the first line contains one integers $m$, indicating the given image is represented by a grid of $m$ rows by $m$ columns. The second line contains a number of integers which is the encoding of the grid image. Encoding always start with green squares. The end of the encoding is denoted by -1.

## Output Format

For each test case, print on one line the number of grassy grids within the enclosed fenced area.

## Sample Input

```
3
5
0 6 3 2 3 2 3 6 -1
7
1 5 2 1 3 1 2 3 1 1 6 1 2 6 1 2 1 2 2 5 1 -1
10
12 6 3 1 6 1 2 1 1 1 4 1 2 1 3 2 1 4 3 1 2 6 4 2 3 7 3 3 14 -1
```

Note: Sample input 3 corresponds to Fig. 1.

## Output for the Sample Input

```
9
1
24
```

# Problem F
## Summing up subarrays
### Time limit: 1 second

The NCPC king has $n$ numbers in $\{1, 2, \ldots, 99999\}$, namely $a_0$, $a_1$, ..., $a_{n-1}$; these $n$ numbers are not multiples of $n$. He wants to find indices $i \in \{0, 1, \ldots, n-1\}$ and $j \in \{i, i+1, \ldots, n-1\}$ such that

$$a_i + a_{i+1} + \cdots + a_j$$

is a multiple of $n$.

## Input File Format

The first line is the number of test cases. Each test case consists of $n$ in one line and $a_0$, $a_1$, ..., $a_{n-1}$ (in that order) in the next line. Two consecutive numbers in a line are separated by a space. There will be at most 10 test cases, and $2 \leq n \leq 100000$.

## Output Format

For each test case, output $i$ and $j$ such that $0 \leq i \leq j \leq n-1$ and $\sum_{k=i}^{j} a_k$ is a multiple of $n$. If there are many correct outputs, just output one of them. It is guaranteed that a solution exists.

## Sample Input

```
5
8
2 7 5 5 1 1 4 6
8
1 4 1 1 4 1 6 5
8
3 4 6 5 3 2 7 4
8
1 6 5 6 2 3 3 5
8
1 6 6 2 6 1 4 2
```

## Output for the Sample Input

```
2 6
4 7
3 4
3 4
2 3
```

# Problem G
## The Number of 2's
### Time limit: 1 second

How many factors of 2 can you extract from factorial of $n$? For example, you know $10! = 1 \times 2 \times \cdots \times 10 = 3628800$. By successively extracting factor 2 from 10!, you get the following sequence

$$3628800, 1814400, 907200, 453600, 226800, 113400, 56700, 28350, 14175\ .$$

The sequence finally stops at 14175 because it is not a multiple of 2. Since there are 9 numbers in this sequence, you can get totally 8 factor 2's from 10!. We use $\epsilon_2(n!)$ to denote this number. In this example, it tells you that $\epsilon_2(10!) = 8$.

The first approach to find $\epsilon_2(n)$ is to write a computer program that computes $n!$ and then extracts 2 successively from it. However, this approach is not practical because $n!$ is easy to be very large. Thus, a more elegant way is to derive a *formula* that can compute this. Fortunately, in many textbooks about number theory, you can find such an amazing formula

$$\epsilon_2(n!) = \left\lfloor \frac{n}{2} \right\rfloor + \left\lfloor \frac{n}{4} \right\rfloor + \left\lfloor \frac{n}{8} \right\rfloor + \cdots = \sum_{k \geq 1} \left\lfloor \frac{n}{2^k} \right\rfloor\ ,$$

where $\lfloor x \rfloor$ returns the largest integer that does not exceed $x$. In our example, $\epsilon_2(10!) = \lfloor 5 \rfloor + \lfloor 2.5 \rfloor + \lfloor 1.25 \rfloor = 5 + 2 + 1 = 8$ by using this formula.

The third way, possibly the most efficient one, is based on the function $v_2(n)$ that computes the number of 1's in the binary representation of $n$. For example, 10 is $1010_2$ in binary, and there are two 1's in its binary representation. Thus, $v_2(10) = 2$. A surprising fact is $\epsilon_2(n!) = n - v_2(v)$. Based on it, you can easily calculate $\epsilon_2(10!) = 10 - v_2(10) = 10 - 2 = 8$. The computation of $v_2(n)$ can be done without using any division. Several acceleration techniques can be applied to speed up the computation of $v_2(n)$. Therefore, the third approach is possibly the most efficient way to computer $\epsilon_2(n!)$. On input $n$, your task is to output the value of $\epsilon_2(n!)$. Notice that either the second or third method can lead to a correct implementation that passes this task.

## Input File Format

The first line gives you the total number of test cases, which is a positive integer that does not exceed 1000. Each test case occupies a single line, which contains only one positive integer $n$, where $1 \leq n < 2^{31}$. Test cases are listed consecutively, so there is no empty line between two adjacent cases.

## Output Format

For each test case, output the value of $\epsilon_2(n!)$ in one line.

## Sample Input

```
4
1
5
10
16
```

## Output for the Sample Input

```
0
3
8
15
```

# Problem H
## Cut Power
### Time limit: 3 seconds

Since the climate changes, the problem of drought is a big issue for the government. Sometimes the government proposes a plan of intermittent supply by districts. The plan adopts the 5-day supply and 2-day cut-off water restriction measure. Suppose the government has $N$ administrative districts, which are numbered as $1, 2, ..., n$. The plan of cutting off water supply is that the government would pick a random number '$m$' ($1 \leq m \leq N$). The district $m$ would first be cut off water supply. And then cut off the water supply in every $m$th district after that, wrapping around to 1 after $N$, ignoring districts already cut off. For example, if $N = 7$ and $m = 3$. Water supplies of the districts would be cut off in the order: 3, 6, 2, 7, 5, 1, 4.

Another big issue is power shortage crisis. The manager of the power company hopes his house will not suffer from power cut off or water cut off. He thinks that if his house is in the last order of power cut off and water cut off, maybe his house can avoid power faliure and water cut.

For example, if the government has eight districts. The order of water cut off with the random number $m = 2$ is 2, 4, 6, 8, 3, 7, 5, 1. Suppose that the manager lives in district #1. If the manager selects the random number $m = 6$ for the plan of power cut off. The order of power cut off is 6, 4, 3, 5, 8, 7, 2, 1. Therefore, the manager's house may have more chance avoiding power faliure and water cut.

The government had announced the random number for water cut off is 2. And the manager knows his house is the last district for water cut. Please write a program to help the manager to choose the random number for power cut off so that his house is also in the last district to have the power cut off. Note that the random number for power cut off must differ from the random number for water cut off.

## Input File Format

There are more than one test cases in the input file. Each test case contains one positive integer, $N$, where $N$ ($2 \leq N \leq 10,000$) indicates the total number of administrative districts. The input is terminated by '`0`'.

## Output Format

For each test case, find the maximum number $m$, with $1 \leq m \leq N$, ensuring that the order of cut power for manager's house is the last. If you cannot find such number, output '`-1`'.

## Sample Input

```
5
8
0
```

## Output for the Sample Input

```
-1
6
```