KTH Challenge 2012

Stockholm, 3rd March 2012

Problems

This page is intentionally left (almost) blank.

Problem A
Spam Filter
Problem ID: spam

Photo by AJ Cann

Goo is working in a well-known Slovak antivirus company which unfortunately cannot be named. In addition to antivirus software, they are developing a spam filter. Recently, Goo has made a few improvements to the filter and he wants to demonstrate his progress to his boss. As you can imagine, demonstrating low-level ideas of your implementation is not a good way to impress your boss, so Goo instead decided to make a presentation with plenty of graphs showing filtering results. The company has a huge database of e-mails and each e-mail is marked as a spam or ham (i.e. not spam). These e-mails were all correctly marked by people – every time someone in the company receives an e-mail, he marks it as either spam or ham and adds it to the database.

The success of Goo's program can be measured in a simple way. Goo ran his program on all e-mails in the database. For each message he noted if his program correctly decided whether the message was spam or ham. The messages were processed in order from the oldest to the newest one. To impress the boss, Goo wants to select e-mails from a period of time and calculate the success rate only for this period. Of course, a period containing only one e-mail won't impress anyone, so Goo wants to choose a period which is long enough.

Task

You are given a sequence of test results and a number k. Your task is to find a continuous subsequence of length at least k which has the highest possible success rate among all such subsequences. The success rate of a subsequence is defined as the number of successfully classified e-mails divided by the length of the subsequence.

Input description

On the first line there is an integer k ($1 \leq k \leq 100$) denoting the minimal subsequence length. The second line contains a string consisting of characters 0 and 1, denoting answers of the program for each e-mail in the database. Number 1 indicates that Goo's program gave a correct answer and 0 that it failed. The length of the string will be at least k and at most 100 000 characters.

Output description

The first and only line of output should consist of two integers f and ℓ, separated by a single space. The integer f is the 1-based index of the first element of subsequence with the best success rate and ℓ is its length. If there are multiple optimal solutions, you can output any one of them.

Sample Input 1	Sample Output 1
1 01	2 1

Sample Input 2	Sample Output 2
4 0110011	2 6

This page is intentionally left (almost) blank.

Problem B
Birds on a Wire
Problem ID: birds

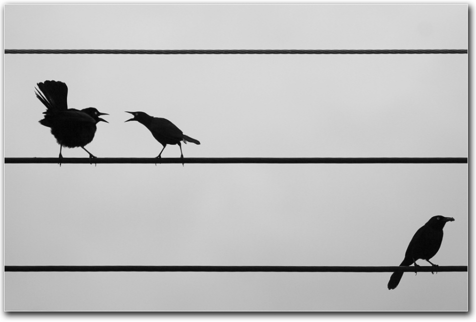

Photo by Tarik Browne

There is a long electrical wire of length ℓ centimetres between two poles where birds like to sit. After a long day at work you like to watch the birds on the wire from your balcony. Some time ago you noticed that they don't like to sit closer than d centimetres from each other. In addition, they cannot sit closer than 6 centimetres to any of the poles, since there are spikes attached to the pole to keep it clean from faeces that would otherwise damage and weaken it. You start wondering how many more birds can possibly sit on the wire.

Task

Given numbers ℓ and d, how many additional birds can sit on the wire given the positions of the birds already on the wire? For the purposes of this problem we assume that the birds have zero width.

Input

The first line contains three space separated integers: the length of the wire ℓ, distance d and number of birds n already sitting on the wire. The next n lines contain the positions of the birds in any order. All number are integers, $1 \leq \ell, d \leq 1\,000\,000\,000$ and $0 \leq n \leq 20\,000$. (If you have objections to the physical plausibility of fitting that many birds on a line hanging between two poles, you may either imagine that the height of the line is 0 cm above ground level, or that the birds are ants instead.) You can assume that the birds already sitting on the wire are at least 6 cm from the poles and at least d centimetres apart from each other.

Output

Output one line with one integer – the maximal number of additional birds that can possibly sit on the wire.

Sample Input 1	Sample Output 1
22 2 2 11 9	3

Sample Input 2	Sample Output 2
47 5 0	8

This page is intentionally left (almost) blank.

Problem C
Lifting Walls
Problem ID: walls


Photo by Richard Štefanec, used with permission.

The local building firm needs your help. They are building an apartment building where the walls are prefabricated and lifted in place using cranes. The building firm has located n possible locations for cranes, and needs to choose some of these so that the center of each wall can be reached by at least one crane. The cranes are quite expensive, so they want to use as few of them as possible. A crane can reach a wall if the wall's center is at most a distance r away.

The house that is to be built is rectangular with a length ℓ and width w.

Task

Find the minimum number of cranes required to reach the center of all four walls.

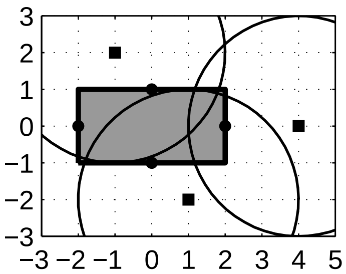

Figure C.1: This example corresponds to sample input 1.

Input

The first line of input contains four space-separated positive integers ℓ, w, n and r, all at most 30. ℓ and w denote the length and width of the house, n denotes the number of possible crane locations, and r denotes the reaching distance of each crane.

This is followed by n lines, each containing two integers x and y ($-100 \leq x, y \leq 100$), denoting a possible location for a crane. The coordinate system has its origin in the center of the building and the x-coordinate along the length of the house. The walls thus have their centers at $(x, y) = (-\ell/2, 0), (\ell/2, 0), (0, -w/2), (0, w/2)$.

Output

Output one integer, the minimum number of cranes required to reach all wall segments, or `Impossible` if not all wall segments can be reached.

Sample Input 1	Sample Output 1
4 2 3 3 1 -2 4 0 -1 2	2

Sample Input 2	Sample Output 2
6 1 1 1 1 0	Impossible

Problem D
Toilet Seat
Problem ID: toilet

Photo by Henry Stern

Many potential conflicts lurk in the workplace and one of the most sensitive issues involves toilet seats. Should you leave the seat "up" or "down"? This also affects productivity, particularly at large companies. Hours each week are lost when employees need to adjust toilet seats. Your task is to analyze the impact different bathroom policies will have on the number of seat adjustments required.

The classical assumption is that a male usually uses a toilet with the seat "up" whereas a female usually uses it with the seat "down". However, we will divide the population into those who prefer the seat up and those who prefer it down, regardless of gender.

Now, there are several possible policies that one could use, here are a few:

1. When you leave, always leave the seat up
2. When you leave, always leave the seat down
3. When you leave, always leave the seat as you would like to find it

So, a person may have to adjust the seat prior to using the toilet and, depending on policy, may need to adjust it before leaving.

Task

Your task is to evaluate these different policies. For a given sequence of people's preferences, you are supposed to calculate how many seat adjustments are made for each policy.

Input

The first and only line of input contains a string of characters 'U' and 'D', indicating that a person in the sequence wants the seat *up* or *down*. The string has length at least 2 and at most 1000.

The first character indicates the initial position of the toilet seat, and the following $n-1$ characters indicate how a sequence of $n-1$ people prefer the seat. You should compute the total number of seat adjustments needed for each of the three policies described above.

Output

Output three numbers, each on a separate line, the total number of seat adjustments for each policy.

Sample Input 1	Sample Output 1
UUUDDUDU	6 7 4

This page is intentionally left (almost) blank.

Problem E
Pub-lic Good
Problem ID: pubs

A bit over three years ago, you were elected president of the glorious and picturesque country of Molvanîa, a land untouched by modern dentistry. To secure your landslide victory in the election you had to make a few promises, some of which, with the clarity of hindsight, may have been a tad exaggerated.

Molvanîa's economy is quite simple compared to that of most other countries. The two main professions in Molvanîa are pub-owners, and beer-drinkers. These two groups combined account for over 75% of the Molvanîan GDP (Gross Domestic Product). Slightly simplified, the system works like this: the beer-drinkers borrow money to pay for their beer. This creates income for the pub-owners. The pub-owners use their income to purchase AAA-rated bonds, backed by loans to beer-drinkers. This system is locally referred to as pub-prime lending.

Photo by National Library of Australia

Task

One of your election-time promises was to further optimize Molvanîa's Tiger economy through improved city planning. You have identified a number of suitable construction sites in which either a pub or a house of a beer-drinker can be built. There are walkways between some of these sites. To fully optimize the economy, you want to place buildings such that each house has at least one pub at only a walkway's distance, and each pub has at least one house at only a walkway's distance. It might happen that this is impossible, but you will try your best.

Beware that the city has a quite peculiar lay-out, and it may not even be possible to draw it on a normal map. Molvanîa is special that way.

Input

There are n construction sites and m walkways in the city ($1 \leq n \leq 10\,000$ and $0 \leq m \leq 100\,000$). The first line contains n and m, separated by a single space. The next m lines contain integers x and y ($1 \leq x, y \leq n$) indicating that there is a walkway between x and y. There are no loops (i.e., $x \neq y$) and all lines with walkway descriptions are distinct.

Output

If it is impossible to build pubs and houses such that every pub is next to a house and every house is next to a pub, print `Impossible` on a line. Otherwise output n space separated words. Print `pub` or `house` for each construction site. The first word indicates what to build at construction site 1, the next at construction site 2, and so on. If there are multiple valid solutions, you can output any of them.

Sample Input 1

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

Sample Output 1

```
pub house pub house
```

Sample Input 2

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

Sample Output 2

```
pub house pub house
```

This page is intentionally left (almost) blank.

Problem F
Xor Maximization
Problem ID: xormax

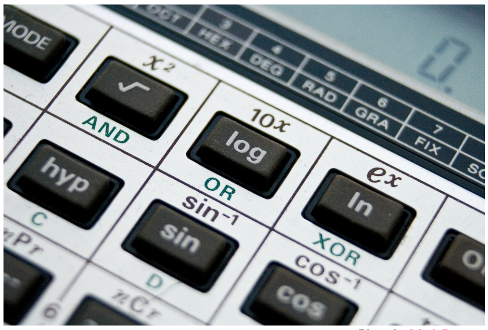

Photo by Mark Ramsay

As you might have heard, Gunnar is an old and forgetful researcher. Most of his research is in security and he cares a bit too much about his own security, so for each website he has a different password. It would be very hard for him to remember all passwords, so for every website he only remembers the method he used to create the password.

For one of the very important websites he started with a file containing a long list of non-negative integers. Since he very much likes the operation $\oplus$ (xor), his password is a xor of some integers in the list. Note that the operation xor is defined on boolean values as $0 \oplus 0 = 1 \oplus 1 = 0$ and $0 \oplus 1 = 1 \oplus 0 = 1$. We can then extend this definition to integers, namely we first write the two integers in binary and then do xor for each two corresponding bits in the numbers. For example the xor of $12 = (1100)_2$ and $5 = (101)_2$ is $9 = (1001)_2$. Instead of addition, we can use the operation xor when summing numbers and we call this modified sum *xor-sum*.

Task

Gunnar's file contains a list of numbers and he selected a subset of the numbers such that its xor-sum is as large as possible. The resulting number was his password. Unfortunately, he forgot the algorithm to find the subset with the largest xor-sum, so he is asking you for help with restoring his password. Of course, he will not tell you for which website this password is.

Input

The first line of input contains an integer n $(1 \leq n \leq 100\,000)$: the length of the list of numbers in Gunnar's file. The second line contains n space separated integers $a_1, \ldots, a_n$ $(1 \leq a_i \leq 10^{18})$, the numbers in the file.

Output

Output one line with the answer – the maximal number Gunnar can get by selecting a subset of the list of numbers and calculating the xor-sum of the subset.

Sample Input 1	Sample Output 1
3 1 3 5	7

Sample Input 2	Sample Output 2
4 2 6 4 8	14

This page is intentionally left (almost) blank.

Problem G
Restaurant Orders
Problem ID: orders

A friend of yours who is working as a waiter has a problem. A group of xkcd-fans have started to come to the restaurant and order food as in the comic strip below. Each order takes him a lot of time to figure out, but maybe you can help him.

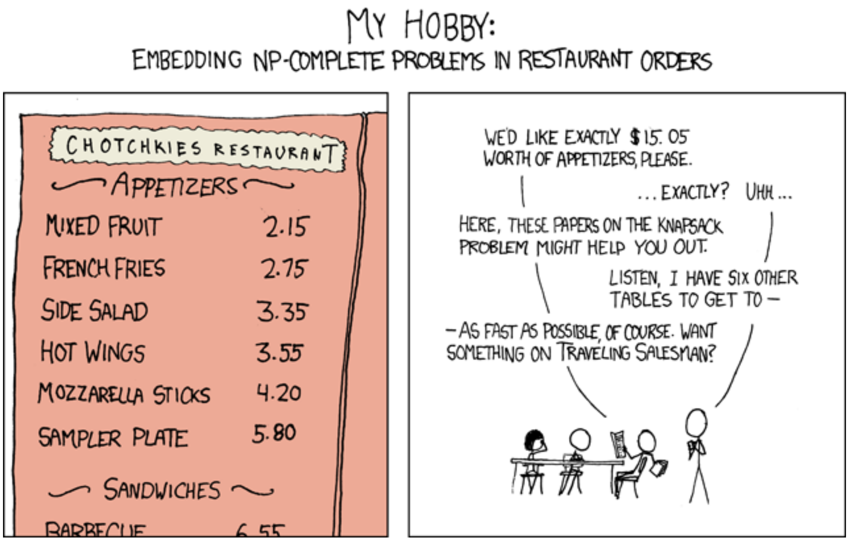

Figure G.1: Comic strip xkcd.com/287.

Task

You are to write a program that finds out what was ordered given the total cost of the order and the cost of each item on the menu.

Input

The input starts with a line containing one integer n ($1 \leq n \leq 100$), the number of items on the menu. The next line contains n space-separated positive integers $c_1, c_2, \ldots, c_n$, denoting the cost of each item on the menu in Swedish kronor. No item costs more than 1 000 SEK.

This is followed by a line containing m ($1 \leq m \leq 1\,000$), the number of orders placed, and a line with m orders. Each order is given as an integer s ($1 \leq s \leq 30\,000$), the total cost of all ordered items in SEK.

Output

For each order in the input output one line as follows. If there is one unique order giving the specified total cost, output a space-separated list of the numbers of the items on that order in ascending order. If the order contains more than one of the same item, print the corresponding number the appropriate number of times. The first item on the menu has number 1, the second 2, and so on.

If there doesn't exist an order that gives the specified sum, output `Impossible`. If there are more than one order that gives the specified sum, output `Ambiguous`.

Sample Input 1	Sample Output 1
3 4 5 8 3 11 13 14	Impossible Ambiguous 1 2 2

Sample Input 2	Sample Output 2
6 215 275 335 355 420 580 1 1505	Ambiguous

Problem H
Three Digits
Problem ID: threedigits

Photo by Sjoerd van Oosten

Per is obsessed with factorials. He likes to calculate them, estimate them, read about them, draw them, dream about them and fight about them. He even has the value of $12! = 479\,001\,600$ tattooed on his back.

He noticed a long time ago that factorials have many trailing zeroes and also wrote a program to calculate the number of trailing zeroes. For example 12! ends with 600, so it has 2 trailing zeroes. Now he wants to make one step further, look at the 3 digits right before the trailing zeroes. In the case of 12!, the last 3 digits before the trailing zeroes are 016.

Task

Given an integer n, find the last 3 digits before the trailing zeroes in $n!$. If there are fewer then 3 such digits, find all of them.

Input

The input contains one line with one integer n ($1 \leq n \leq 10\,000\,000$).

Output

Output one line with the 3 digits before trailing zeroes of $n!$. If there are fewer than 3 such digits, output all of them.

Sample Input 1	Sample Output 1
5	12

This page is intentionally left (almost) blank.