KTH Challenge 2015

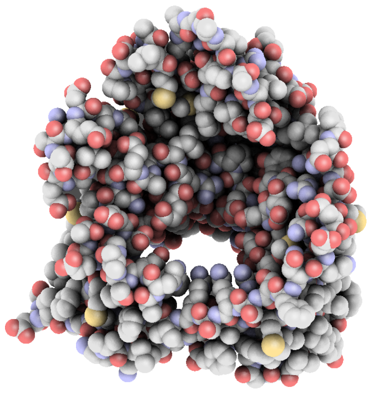
Stockholm, 26th April 2015

Problems

A	Proteins
B	Black Friday
C	Absurdistan Roads III
D	Xortris
E	Shibuya Crossing
F	Spock
G	A1 Paper
H	Odd Binomial Coefficients
I	The Addition Game

This page is intentionally left (almost) blank.

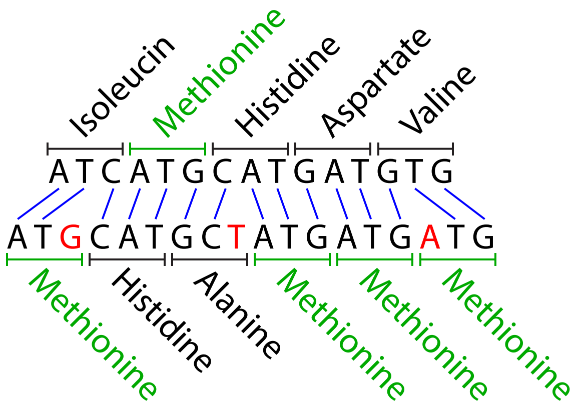
Problem A
Proteins
Problem ID: proteins


Protein molecule by ALoopingIcon

Magnus is a biologist. He is playing with proteins all day long and now he wants to know what these molecules look like. He has heard that X-ray crystallography can be used to get images of proteins that contain a lot of sulfur atoms. Magnus does not think his proteins contain enough sulfur, but he is willing to change them to get this to work. Magnus has bacteria producing his proteins for him, and he is planning to mutate these bacteria to change the proteins.

Magnus knows the DNA strings coding his proteins and how the DNA is translated into the amino-acid sequence making up the protein. The first three letters in the code determine the first amino acid, the following three letters determine the second, and so on. Whenever those three letters are ATG (in that order) the amino acid methionine will be incorporated into the protein. Methionine contains a sulfur atom, so Magnus wants to have many methionines in his proteins. Magnus can only change the DNA code by inserting letters. This, however, takes a lot of time for each letter he wants to insert. Knowing that you are good at computer stuff, he asks you for help. Can you figure out the smallest number of letters that need to be inserted into the DNA code to make it code for n methionines?

For example, the DNA string TGATGC codes for no methionines, but adding an A in the beginning turns it into ATGATGC which has two ATG blocks and thus codes for two methionines. This is the first sample input and a solution to the second sample input is shown in the figure below.



Input

The first line of input contains a single positive integer $n \leq 10^6$, the number of methionines to be included in the protein. The second line contains a non-empty DNA string of at most 1000 letters, each either A, T, G, or C.

Output

Output a single integer, the smallest number of letters that can be inserted into the DNA string to make at least n of its three-letter blocks be ATG.

Sample Input 1	Sample Output 1
2 TGATGC	1

Sample Input 2	Sample Output 2
4 ATCATGCATGATGTG	3

Problem B
Black Friday
Problem ID: blackfriday

Black friday by Powhusku

Black Friday is the Friday following Thanksgiving Day in the United States (the fourth Thursday of November). Since the early 2000s, it has been regarded as the beginning of the Christmas shopping season in the US, and most major retailers open very early and offer promotional sales. (Source: Wikipedia)

You work in the IT support division of an electronics store. This year, in an attempt to prevent overcrowding, the management has decided to limit the number of people entering the store. They divide the people at the entrance into groups of size n and process them as follows: all n participants roll a die, and report the outcomes $a_1, a_2, \ldots, a_n$. To prevent cheating, instead of selecting the one with the highest outcome, the rule is to select the participant with the highest unique outcome. Everybody not selected has to move to the back of the queue. If there is no winner, the experiment is repeated.

For example, if there are three players and the outcomes are 6, 6 and 5, then the third player wins, because the first and second player lose even though their outcomes are higher, since they both have the same outcome. If instead the third player also rolls 6, then nobody wins.

They asked you to write a program for selecting the winner.

Input

The first line of the input contains one integer n, $1 \leq n \leq 100$, the group size. The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 6$ for all $1 \leq i \leq n$): the outcome of each participant's die roll.

Output

Output the index of the participant that has the highest unique outcome, or “none” (without the quotes) if nobody has a unique outcome.

Sample Input 1

```
8
1 1 1 5 3 4 6 6
```

Sample Output 1

```
4
```

Sample Input 2

```
3
4 4 4
```

Sample Output 2

```
none
```

This page is intentionally left (almost) blank.

Problem C
Absurdistan Roads III
Problem ID: absurdistan3

Photo by Ivan McClellan

The people of Absurdistan discovered how to build roads only last year. After the discovery, each city decided to build its own road, connecting the city to some other city. Each newly built road can be used in both directions.

You bought a tourist guide which has a map of the country with the newly built roads. However, since you are very interested in history, you would like to know which city built which road.

Given the description of n roads, can you find an assignment of roads to n cities, such that each city built one road? If there are multiple assignments, you are happy with any one. At least one solution is guaranteed to exist.

Input

The first line contains an integer n $(2 \leq n \leq 100\,000)$ – the number of cities and roads. Then follow n lines with 2 numbers each. A line containing "a b" indicates that there is a road between cities a and b, $1 \leq a, b \leq n, a \neq b$. There can be multiple roads between the same pair of cities.

Output

Print n lines with two integers "a b" denoting that a road between a and b was built by city a. Each road from the input must appear exactly once in the output. If there are multiple solutions, you can print any one and you can print the roads in any order.

Sample Input 1

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

Sample Output 1

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

Sample Input 2

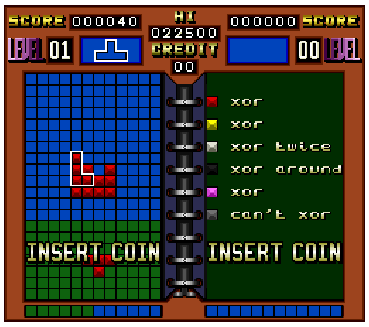
```
2
1 2
1 2
```

Sample Output 2

```
2 1
1 2
```

This page is intentionally left (almost) blank.

Problem D
Xortris
Problem ID: xortris



Screenshot of World of Xor

It is 1990 and you are in the development team of a video game that is going to revolutionize the future of arcades. The player is given a rectangular board with some white and black squares. The goal is to turn the whole board white. At each turn, the player may choose a tetromino from an infinite supply, move and rotate it within the limits of the board, and toggle the colour of the four squares covered by the tetromino. A tetromino is a connected set of 4 squares (see Figure D.1).

Unfortunately, the testing team has been complaining about some levels being impossible to solve. You know that testers are skilled enough to place a piece in any position and rotation needed, so the problem may be somewhere else. Your next debugging step is to write a program that checks whether a level is solvable.

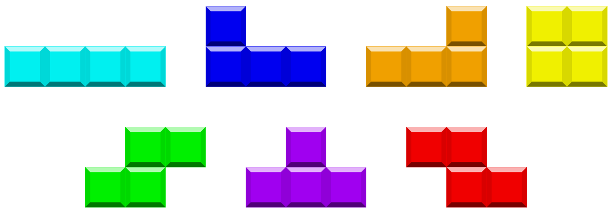


Figure D.1: All tetrominoes. From Wikimedia.

Input

The first line contains two integers m and n ($1 \leq m, n \leq 100$), the dimensions of the board. m lines with n characters each follow. The character '`.`' represents a white square, and the character '`X`' represents a black square.

Output

One line with the word "`possible`" if the level is solvable and "`impossible`" if it is not.

Sample Input 1	Sample Output 1

`3 3`
`...`
`...`
`...`
	`possible`

Sample Input 2	Sample Output 2

`3 3`
`XXX`
`XXX`
`XXX`
	`impossible`

Problem E
Shibuya Crossing
Problem ID: shibuyacrossing

Shibuya Crossing by Guwashi

The Shibuya scramble crossing in Tokyo is infamous for being heavily used, resulting in people bumping into each other. The crossing can be modeled as a convex polygon, where the n people about to cross initially stand at a point that is on the perimeter of the polygon and in its lower half. When the traffic lights change, each person starts to walk towards a unique point on the perimeter in the upper half of the polygon. The path each person takes may look like spaghetti (it may even cross itself), but it will never leave the polygon and no two paths will cross more than once.

Oskar who is a badass geek observes the crossing from the Starbucks nearby. He has numbered the people in the crossing consecutively 1 through n in counter-clockwise order (starting with the person at the very left). Sadly he doesn't know the intended paths of the people at the crossing, but he has gathered some intelligence telling him exactly which persons' paths will cross one another (and this information is consistent with the physical reality).

Being a nerd he obviously knows about Murphy's Law saying "Anything that can go wrong, will go wrong!". So all people who could possibly bump into each other, i.e., all people whose paths cross, will actually bump into each other! He now asks himself, "After all the n people have crossed, what is the size of the largest group of people where everyone has bumped into each other?". Now that is a geeky and tough question, can you help him?

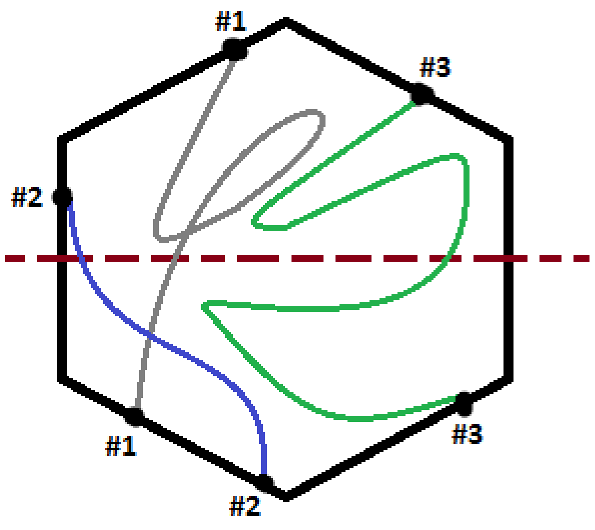


Figure E.1: A beautiful illustration of a possible interpretation of the first sample test case.

Input

The first line contains an integer $1 \le n \le 800$, the number of people at the crossing, and an integer $0 \le m \le 10\,000$, the number of paths that will cross, i.e., intersect one another. The next m lines each contain two integers a and b, $1 \le a < b \le n$, meaning that the path taken by person a will cross the path taken by person b. (No pair will occur twice in the input.)

Output

Output a single integer giving the size of the largest group of people where everyone has bumped into one another.

Sample Input 1

```
3 1
1 2
```

Sample Output 1

```
2
```

Sample Input 2

```
5 7
1 3
1 5
1 4
2 4
3 4
2 5
2 3
```

Sample Output 2

```
3
```

Problem F
Spock
Problem ID: spock

Leo is participating in a friendly game of *rock-paper-scissors-lizard-Spock* against his computer. The game proceeds in rounds. In each round, Leo and his computer both choose, simultaneously, between five options: rock, paper, scissors, lizard, and Spock. Each of these five options wins over two of the other options, as illustrated by Figure F.1. For example, rock wins over lizard and scissors, but loses against paper and Spock. If both players choose the same option, the round is a draw. In the end, each of the two players gets a score which is the number of rounds they won.

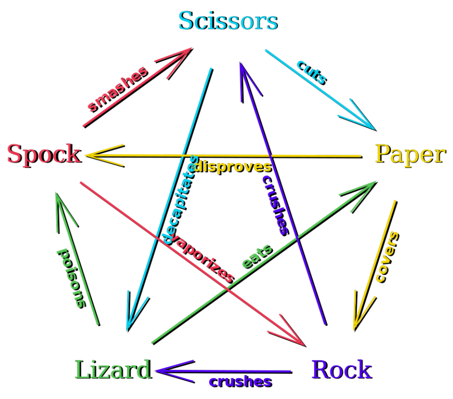


Figure F.1: The mechanics of the game. Illustration by VidTheKid via Wikimedia Commons

Alas, Leo's computer is not the sharpest tool in the shed, and simply follows a strategy where in each round it selects one of the five options uniformly at random. This makes the game quite boring, because regardless of Leo's strategy, each player is expected to win 40% of the rounds (and 20% of the rounds are expected to be draws).

Did we mention that Leo's computer is a bit lacking in mental capacities already? Well, it gets worse: in order to pick random options, Leo's computer uses a very simple linear congruential generator (LCG). The LCG has three parameters: a *known* prime number $p = 127$, and two fixed but *unknown* integers $0 \leq a < p$ and $0 \leq b < p$. Additionally, it has a state, an integer $0 \leq x < p$. To generate a random option, Leo's computer first updates the state according to the rule

$$x \leftarrow (a \cdot x + b) \bmod p,$$

and then chooses one of the five options based on the value of $x \bmod 5$, according to the following table:

x mod 5:	0	1	2	3	4
Option chosen:	rock	paper	scissors	lizard	Spock

Logically, knowing how Leo's computer chooses its random numbers should give Leo an advantage in the game. But unfortunately Leo died, so it is now up to you to finish this. Write a program which, when playing against Leo's computer for several rounds, wins at least 80% of them.

Interaction

This problem is interactive, proceeding in the rounds of the game. Before the game starts, there will be one line of input containing an integer $100 \leq r \leq 1000$, the number of rounds to play.

Then, for each round, your program should write one line containing one of the five strings "`rock`", "`paper`", "`scissors`", "`lizard`", or "`Spock`", indicating the option Leo should choose for the next round. After writing this line you need to make sure to flush standard output. Then, one line of input will be available, containing the name of the option chosen by Leo's computer in the round, and the game proceeds to the next round.

A Java jar file SpockDebugger.jar is provided to aid in testing your solution. Running `java -jar SpockDebugger.jar` in a terminal gives information about usage.

Sample Interaction

Suppose that Leo's computer uses the parameters $a = 17$, $b = 23$, and is initially in state $x = 42$. Then in the first round, the state is updated to $(17 \cdot 42 + 23) \bmod 127 = 102$, and since $102 \bmod 5 = 2$, the first move made by the computer is `scissors`. In the next round, the state is updated to $(17 \cdot 102 + 23) \bmod 127 = 106$, resulting in the move `paper`. The first 10 rounds of the game are shown below. In this example, Leo has opted to simply make the move `Spock` in every single round, which seems to have been a surprisingly good choice – of these 10 rounds Leo actually wins 7 (all but the second, sixth, and seventh) – but Leo still is not winning 80% of the rounds. In general, this strategy will be too naive to defeat the computer.

The extra newlines in this example are for clarity only, to demonstrate the order of events. You should print exactly one newline character after each move, and the input contains no blank lines either.

Sample Input 1

```
500

scissors

paper

scissors

rock

scissors

paper

Spock

scissors

rock

scissors

...
```

Sample Output 1

```

Spock

Spock

Spock

Spock

Spock

Spock

Spock

Spock

Spock

Spock

...
```

Problem G
A1 Paper
Problem ID: a1paper

Björn likes the square root of two, $\sqrt{2} = 1.41421356\ldots$ very much. He likes it so much that he has decided to write down the first $10\,000$ digits of it on a single paper. He started doing this on an A4 paper, but ran out of space after writing down only 1250 digits. Being pretty good at math, he quickly figured out that he needs an A1 paper to fit all the digits. Björn doesn't have an A1 paper, but he has smaller papers which he can tape together.

Taping two A2 papers together along their long side turns them into an A1 paper, two A3 papers give an A2 paper, and so on. Given the number of papers of different sizes that Björn has, can you figure out how much tape he needs to make an A1 paper? Assume that the length of tape needed to join together two sheets of papers is equal to their long side. An A2 paper is $2^{-5/4}$ meters by $2^{-3/4}$ meters and each consecutive paper size (A3, A4, ...) have the same shape but half the area of the previous one.

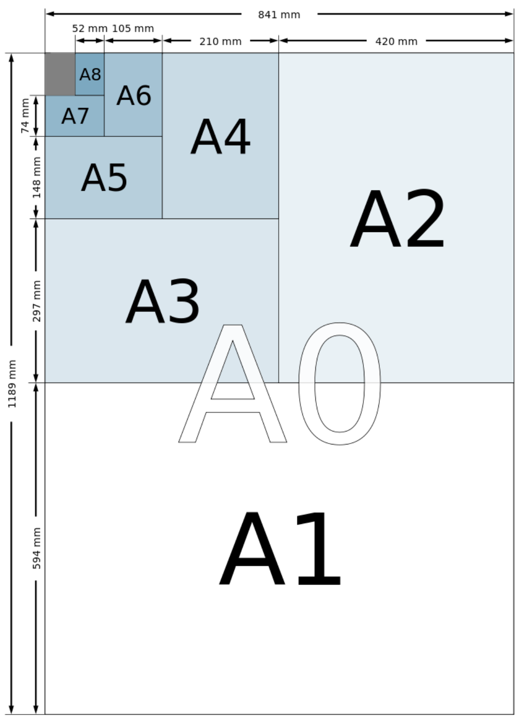


Illustration of the A series paper sizes by Bromskloss

Input

The first line of input contains a single integer $2 \le n \le 30$, the A-size of the smallest papers Björn has. The second line contains $n-1$ integers giving the number of sheets he has of each paper size starting with A2 and ending with An. Björn doesn't have more than 10^9 sheets of any paper size.

Output

If Björn has enough paper to make an A1 paper, output a single floating point number, the smallest total length of tape needed in meters. Otherwise output "`impossible`". The output number should have an absolute error of at most 10^{-5}.

Sample Input 1

```
4
1 0 5
```

Sample Output 1

```
1.60965532263
```

Sample Input 2

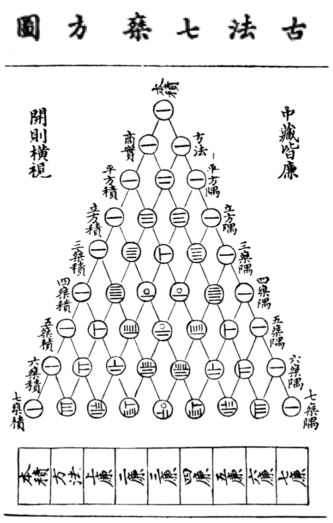
```
3
0 3
```

Sample Output 2

```
impossible
```

This page is intentionally left (almost) blank.

Problem H
Odd Binomial Coefficients
Problem ID: oddbinom



You might be familiar with the binomial coefficient $\binom{m}{k}$ defined as $\binom{m}{k} = \frac{m!}{k!(m-k)!}$, where m and k are non-negative integers and $k \leq m$. Let $T_2(n)$ be the number of odd binomial coefficients such that $0 \leq k \leq m < n$. The most useful mathematical inequality you will learn during this competition is

$$0.812556n^{\log_2 3} \leq T_2(n) \leq n^{\log_2 3}.$$

Emma doesn't like such imprecise inequalities and would like to calculate $T_2(n)$ exactly. Can you help her?

Input
The input contains one line with one integer $n, 1 \leq n \leq 10^{11}$.

Output
Output one line with the value of $T_2(n)$.

Sample Input 1	Sample Output 1
4	9

Sample Input 2	Sample Output 2
6	15

This page is intentionally left (almost) blank.

Problem I
The Addition Game
Problem ID: additiongame

Alan works for a company specialising in computer security. He recently came up with what he thinks is a great public key cryptosystem, in which the private key consists of two permutations π and σ of $\{1, \ldots, n\}$. The public key $(a_1, \ldots, a_n)$ is then given by $a_i \equiv \pi_i + \sigma_i \pmod n$ for $1 \le i \le n$. The expression $x \equiv y \pmod n$ means that x and y have the same remainder after division by n.

As an example with $n = 5$, consider

$$\begin{aligned}\pi &= (3, 1, 5, 2, 4),\\ \sigma &= (5, 1, 3, 4, 2), \text{ and}\\ a &= (3, 2, 3, 1, 1).\end{aligned}$$

Here, for example, $a_5 \equiv 1 \equiv 4 + 2 \equiv \pi_5 + \sigma_5 \pmod 5$, and all the entries in π and σ respectively are $\{1, \ldots, 5\}$, each number occurring exactly once.

Alan's coworkers have some doubts about this system being secure, since finding any private key corresponding to the public key would break the system. Your task is to help them out. Given n and a sequence $a = (a_1, \ldots, a_n)$, determine whether there are two permutations π and σ such that $\pi_i + \sigma_i = a_i \pmod n$ for each i. If there are more such pairs, print any of them.

Input

The first line contains the length n of the sequence and the permutation is written. The second line contains integers $a_1, \ldots, a_n$, satisfying $1 \le a_i \le n$. The length n satisfies $1 \le n \le 1000$.

Output

If there is no solution, output "`impossible`". If there is a solution, output any of them, writing the two permutations on one line each.

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

Sample Input 2	Sample Output 2
4 3 1 1 4	impossible

This page is intentionally left (almost) blank.