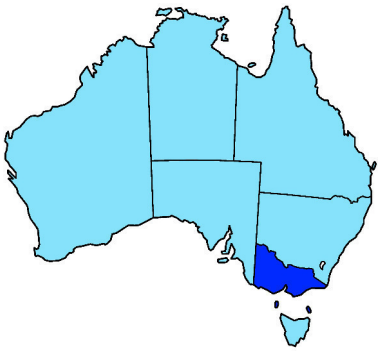


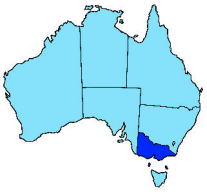
Victoria Collegiate Programming Contest

August 15, 2015

Contest Problems

A: Acronyms
B: Bullseye Coverage
C: Cinema
D: Double Free Sets
E: Environmental Concern II
F: Fantastic Mountains
G: Graph Game
H: Hotel VM
I : Interesting DNA
J : Jumbling Cards
K: Kind Words

This Page Is Intentionally (Almost) Blank.



A: Acronyms

Time Limit: 1 second(s)

An acronym is an abbreviation of a phrase. You simply take the first character of each significant word represented as a capital letter. It's as simple as that!

It really makes items easy to remember. Instead of remembering "International Collegiate Programming Contest", you can just remember "ICPC". For this problem, I need you to tell me the acronym of the phrase. A word is considered significant if it is not a filler word. For this problem, the only filler words are: "and", "or", "a", "the", "is", "in", "on", "for" and "to" (or any capitalization of any of these words).

Input

The input will contain a single test case.

The line will have the phrase containing words separated by single spaces. The words will only contain alphabetic characters and the length of the phrase will not exceed 100 characters. At least one word in the line will not be a filler word.

Output

Output the acronym of the phrase.

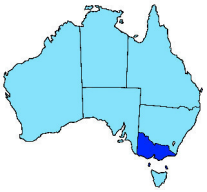
Sample Input and Output

Sample Input	Sample Output
`Victoria Collegiate Programming Contest`	`VCPC`

Sample Input	Sample Output
`Mind Your Own Business`	`MYOB`

Sample Input	Sample Output
`A happy antelope is particularly pesky`	`HAPP`

This Page Is Intentionally (Almost) Blank.



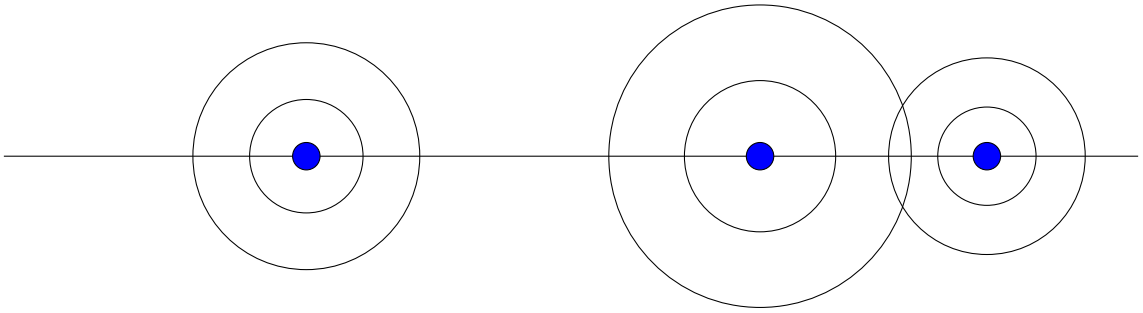
B: Bullseye Coverage

Time Limit: 3 second(s)

Have you ever heard of the great spy named Adriano George from Nlognia? Of course you haven't! Very few *great spies* are well-known, because if they are, they're probably not doing a very good job being a spy!

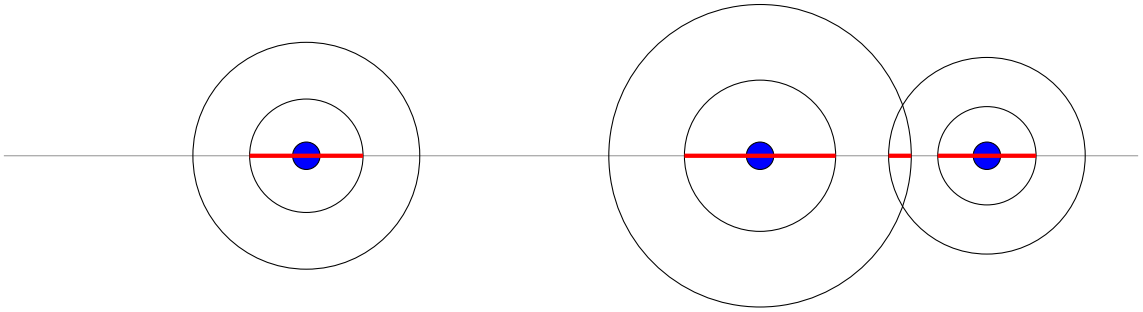
Anyways, Adriano George has been spotted in a neighbouring country and you have been appointed by your country's leader to assess how secure your border is. To determine such a thing, you have decided to look at the distribution of guards along your border. For the sake of this problem, you may assume that the border is a long, straight line and that each guard is standing directly on the border.

You have come up with a system for determining the border security and have called it "bullseye coverage". The idea is this: each guard has k concentric circles surrounding them with radii $r_i, 2r_i, 3r_i, \ldots, kr_i$. For example, here is a border with 3 guards and $k = 2$:

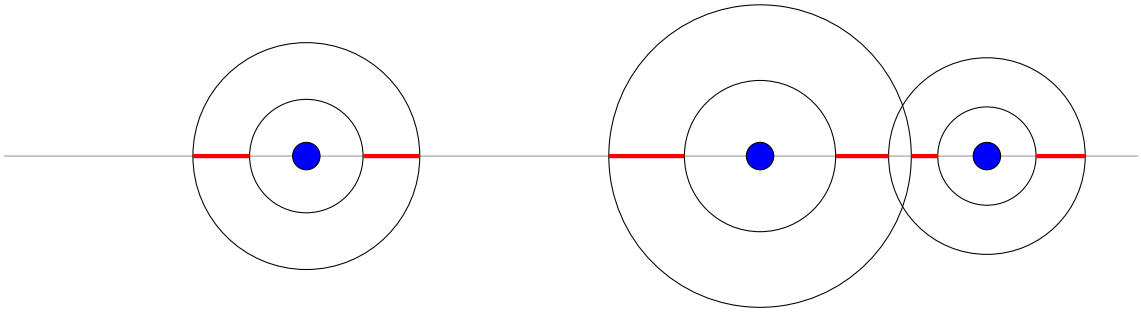


You should note that each guard can have a different radius associated to them. These radii determine how much of the border each guard can effectively patrol. The better the guard, the larger of an area they can patrol. The rings are to indicate that a guard can patrol the part of the border closest to him easier than the part of the border that is further away. For this reason, any part of the segment which is inside the innermost ring gets a *score* of k. The second most inner ring gets a *score* of $k-1$ and so on until you arrive at the outer-most ring, who gets a score of 1. The score of an interval is additive. For example, in the example above, there is overlap between the two rightmost guards. In that intersection, the *score* would be 2 (one from the second guard and one from the third guard).

For clarity, here are the sections of the border which have a score of 2:



And here are the sections of the border which have a score of 1:



Everywhere not covered in the above cases has a score of 0.

Obviously, the higher the score, the more protected that section of border is. But is having a score of 150 really that much better than a score of 149? To account for this, you will be required to take the ceiling of the logarithm (base 2) of the score+1 at each point on the line (call this the *protection factor*). Here are the first few conversions:

Score	Protection Factor
0	0
1	1
2	2
3	2
4	3
5	3
6	3
7	3
8	4
9	4

The total security of the border is the sum of the protection factors of the entire wall. For example, a section of border that is 2 metres long and has a protection factor of 4 will contribute 8 to the total security of the border. You may assume that the wall stretches on indefinitely in each direction.

Input

The input will contain a single test case.

The first line will contain two integers, n ($0 \leq n \leq 100\ 000$) and k ($1 \leq k \leq 10$) denoting the number of guards and the number of rings surrounding each guard, respectively.

The next n lines will contain two integers each, x_i ($0 \leq x_i \leq 10^{12}$) and r_i ($1 \leq r_i \leq 10^{12}$), denoting the $x-$coordinate of this guard and the radius associated with him/her.

Output

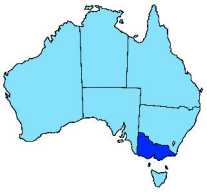
Output one integer, the total security of the border.

Sample Input and Output

Sample Input	Sample Output
3 2 40 8 100 10 130 7	150

Sample Input	Sample Output
1 1 3 2	4

This Page Is Intentionally (Almost) Blank.



C: Cinema

Time Limit: 1 second(s)

Alice loves movies and she always goes to Wonderland Cinema. Here, she can watch any movie she would like to, and moreover, she even gets a discount for watching many consecutive hours of movies! She has a list of movies that she would like to watch. For each of them, she knows the length of the movie and the price of admission. She wants to know the cheapest way for her to watch all of these movies (possibly spread out over multiple days).

Wonderland Cinema has a table that shows how much of a discount you get for watching h consecutive hours of movies. For safety reasons, they never allow you to watch more than k consecutive hours of movies in one sitting.

For example, say Alice wants to watch two movies (A and B). The length of A is 2 hours and the price is \$3 and the length and price of B are 3 hours and \$4. If Wonderland Cinema gives the following discounts:

Hours	1	2	3	4	5	6	7
Discount	0	0	1	5	6	7	7

then Alice could watch both movies back-to-back and receive the \$6 discount (so she only pays \$1) or she could watch the movies separately and receive no discount for the 2 hour movie and a \$1 discount for the 3 hour movie (so she pays \$6).

Note that the discount cannot bring the cost of showings below zero. For example, if both movie X and Y have a length of 3 hours and a price \$2 and the discount for 6 hours is \$7, then Alice can watch X and Y for free (but she does not get extra money for a future transaction).

Alice can watch the movies in any order and wants to minimize the cost of watching all n movies, can you help her?

Input

The input will contain a single test case.

Each test case has 4 lines.

The first line has an integer k ($1 \leq k \leq 1000$), the maximum number of consecutive hours Alice can watch movies. The second line has n ($1 \leq n \leq 10$) space separated integers, the ith integer is the length of the ith movie, which is no greater than k. The third line also has n space separated integers, the ith integer is the price of watching the ith movie (each price will be positive and no more than 100). The last line has k space separated integers, the first among them is the discount for watching 1 hour and the ith among them is the discount for watching i consecutive hours.

All movies have a length no greater than k.

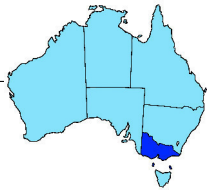
Output

Output one integer, the minimum cost for Alice to watch all the movies on her list.

Sample Input and Output

Sample Input	Sample Output
8 2 3 3 4 5 5 0 1 1 1 1 2 2 2	11

Sample Input	Sample Output
8 2 3 2 4 3 2 5 2 4 3 3 5 4 6 3 5 6 4 3 4 0 0 1 1 1 2 2 2	34



D: Double Free Sets

Time Limit: 1 second(s)

A set of integers S is called *double free* if for every entry in the set, twice that number is not in the set. For example, the set of odd numbers and the set of prime numbers are both double free sets:

$$\{1, 3, 5, 7, 9, 11, 13, \ldots\}$$

$$\{2, 3, 5, 7, 11, 13, 17, 19, 23, \ldots\}$$

Whereas the following sets are not double free:

$$\{1, 2, 3, 5, 7, 10\}$$

$$\{3, 5, 7, 10, 12\}$$

In this problem, we are looking for the largest double free set which is a subset of the first n integers $(1, 2, \ldots, n)$.

Input

The input will contain a single test case.

The test case will contain a single integer n $(1 \leq n \leq 10^{16})$.

Output

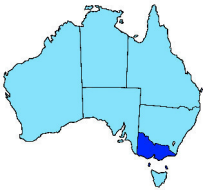
The output should contain one positive integer: the largest size of a double free subset of the first n positive integers.

Sample Input and Output

Sample Input	Sample Output
3	2

Sample Input	Sample Output
10	6

This Page Is Intentionally (Almost) Blank.



E: Environmental Concern II

Time Limit: 1 second(s)

In last year's VCPC, we helped a local group of environmentalists who had formed together to stop the tearing down of the forest. Thankfully, we were successful and the forest is still there today!

This year, we must save another precious piece of the environment. A very rare species of grass has been found growing in a garden in Melbourne's CBD. This species of grass has long thought to be extinct, so its discovery is very special. Unfortunately, this species of grass tastes delicious (at least to small insects like the house fly!).

I have recently created an insect killing machine that can be mounted on a small pole and kill any fly that gets within k metres of it (the exact value of k can be different for different machines). The patch of land where the grass is growing can be described by a non-intersecting simple polygon. We will be placing an insect killing pole on each of the vertices (each with their own k value). You may assume that the height of the poles is negligible, we are only interested in the area covered in the 2-D space when looking from directly above.

Given the polygon that describes the distribution of the grass as well as the k value on each vertex, I need you to tell me if all of the grass is safe! That is, I need to know if every blade of grass is covered by at least one of the insect killing devices.

Input

The input will contain a single test case.

The first line will contain an integer n ($3 \leq n \leq 50$) denoting the number of vertices on the polygon. The next n lines contain three integers: x ($-1000 \leq x \leq 1000$), y ($-1000 \leq y \leq 1000$) and k ($1 \leq k \leq 1000$). These are the (x, y) coordinate of the vertex of the polygon and the k value associated with the insect killing machine mounted on that vertex.

The first vertex in the input is connected with the last vertex and the second vertex, the second vertex is connected with the first vertex and the third vertex, and so on. You may assume that the polygon has positive area and that there is no point that is the intersection of 3 or more objects (line/line or line/circle or circle/circle).

Output

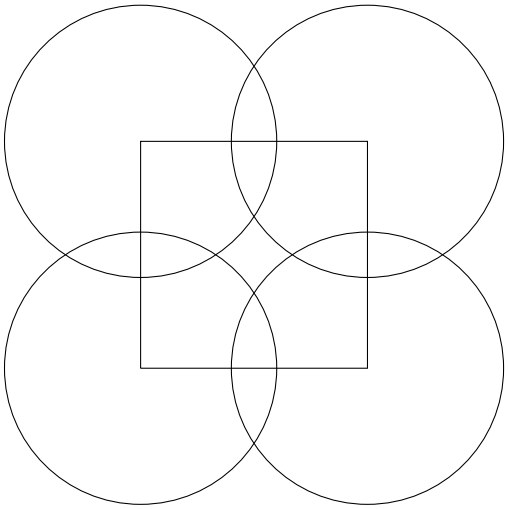
If all of the grass is safe, output `Safe`. Otherwise, output `Not Safe`.

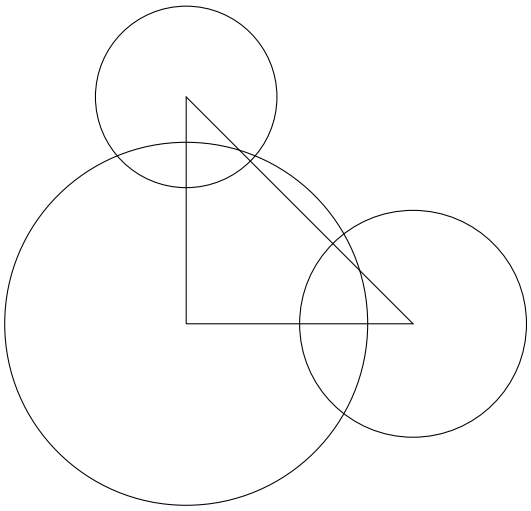
Sample Input and Output

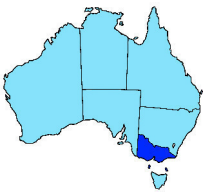
Sample Input	Sample Output
4 -5 -5 6 -5 5 6 5 -5 6 5 5 6	Not Safe

Sample Input	Sample Output
3 0 0 8 0 10 4 10 0 5	Safe

For your convenience, here are the images for the sample inputs:







F: Fantastic Mountains

Time Limit: 1 second(s)

Mountains are one of the most beautiful sights in the world. In this problem, I want you to draw me the skyline of the mountains in ASCII. All mountains will be simple triangles with a slope of 1 or -1. For example, here is a mountain of height 3:

```
..*..
.***.
*****
```

When two (or more) mountains overlap, all we can see is the skyline—we cannot distinguish between different mountains. For example:

```
......*...
..*..***..
.********.
**********
```

Input

The input will contain a single test case.

The first line will contain an integer n ($1 \leq n \leq 100$), the number of mountains in the input. The next n lines contain two integers each x_i ($1 \leq x_i \leq 100$) and h_i ($1 \leq h_i \leq 100$) describing the n mountains. The ith mountain's peak is of height h_i and is located in the x_ith column.

Output

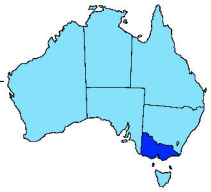
Output an image of the mountains. Your image should contain only two different characters: '.' and '*'. A * should be used to denote part of a mountain and a . should be used elsewhere. The outputted image should be a rectangle. The image should contain all of the mountains from the leftmost left endpoint of a mountain to the rightmost right endpoint of a mountain. All horizontal lines printed must contain at least one piece of a mountain.

Sample Input and Output

Sample Input	Sample Output
1 3 3	..*.. .***. *****

Sample Input	Sample Output
3 3 3 7 4 14 2	......*........ ..*..***....... .********....*. **********..***

This Page Is Intentionally (Almost) Blank.



G: Graph Game

Time Limit: 2 second(s)

Andrew and Benedict play a game on a directed graph that initially contains n vertices labelled $1, 2, \ldots, n$. This graph already contains m edges and may contain multi-edges (two or more edges connecting the same nodes). The game is played in two stages.

First, Andrew takes his turn: he is given a multiset S of k new edges from u_j to v_j with cost w_j. He must choose a subset (or, rather, a "sub-multiset") of these edges from this set with total cost not greater than A. These edges will be added to the original graph.

Now, Benedict may take up to B turns: on each turn, he may choose an edge in the graph (from those initially in the graph, or those added by Andrew) and remove it from the graph. After the B turns, the game ends. If, at any point, he runs out of edges to remove, he does nothing.

If at the end of the game, there is a path from vertex 1 to vertex n, then Andrew wins. If there is no such path, then Benedict wins. Andrew and Benedict want to know if both players play optimally (make the best possible moves), which player will win the game. Please help them!

Note that Andrew may choose to effectively add multiple edges between a pair of vertices u and v, if such pair appears multiple times in the multiset S. However, each addition may have a different cost. Similarly, he may choose an edge in S to be added to the graph even if the same edge is already present in the original graph.

Input

The input will contain a single test case.

The first line of input contains five integers: n ($2 \leq n \leq 500$), m ($0 \leq m \leq 1000$), k ($0 \leq k \leq 1000$), A ($0 \leq A \leq 10000$) and B ($0 \leq B \leq 2000$).

The next m lines each contain two integers u_i and v_i ($1 \leq u_i, v_i \leq n$) on each giving an edge initially in the graph going from u_i to v_i. The next k lines each contain three integers u_j, v_j and w_j ($1 \leq u_j, v_j \leq n$ and $0 \leq w_j \leq 10000$) giving an edge Andrew can choose to add from u_j to v_j and its cost.

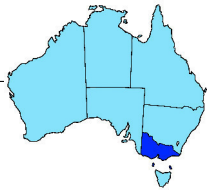
Output

Print the name of the winner – `Andrew` or `Benedict`.

Sample Input and Output

Sample Input	Sample Output
4 2 3 25 1 1 2 3 4 1 4 8 2 4 19 1 3 17	Andrew

Sample Input	Sample Output
3 1 2 1 1 1 2 1 2 1 2 3 1	Benedict



H: Hotel VM

Time Limit: 1 second(s)

Hugo is the lift boy in Cloudscrape tower, charged with ferrying visitors from the lobby on the ground floor to the observation deck on the 314th floor. One day, a group of vampires and maidens arrives at the tower, requesting to travel to the top. But the lift is too small to fit all the visitors at once, so Hugo must make multiple trips. To complicate the situation, the vampires have worked up quite a thirst for blood, and so will attempt to drink the blood of a maiden if ever the vampires outnumber the maidens. This is against company policy.

If the vampires ever outnumber the maidens at the top of the tower or at the bottom of the tower when the lift boy is not present, the vampires will attack a maiden. Conversely, if there are no maidens present, or the maidens number equal or more than the vampires, there is no problem. The maidens are always safe when the lift boy is there (this includes in the elevator and in the *transition period* of people getting in and out of the elevator).

Additionally, it is company policy that the lift must carry at least one passenger for all trips (both up and down), and the lift cannot be overfilled. There are no intermediate stops between the ground and the observation deck. What is the fewest number of trips that the lift must take to carry all the visitors from the ground to the observation deck?

Input

The input will contain a single test case.

The input will contain 3 integers: V ($1 \leq V \leq 100$), M ($1 \leq M \leq 100$) and L ($1 \leq L \leq 50$) denoting the number of vampires, number of maidens and the maximum capacity of the elevator, respectively.

Output

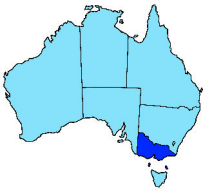
The first line of the output is either the minimum number of trips necessary, n, or the string `IMPOSSIBLE` if it is impossible to get everyone to the top without having the vampires outnumber the maidens. If it is possible, the next n lines should show one solution of how to get everyone to the top in n steps. Each line represents one trip of the lift, containing two numbers, v and m, where v is the number of vampires in the lift and m is the number of maidens in the lift, and either the string `UP` or `DOWN`, as appropriate. If there are multiple solutions to get everyone to the top of the tower in the minimum number of steps, any will be accepted.

Sample Input and Output

Sample Input	Sample Output
2 2 3	3 1 1 UP 0 1 DOWN 1 2 UP

Sample Input	Sample Output
8 4 3	IMPOSSIBLE

This Page Is Intentionally (Almost) Blank.



I: Interesting DNA

Time Limit: 5 second(s)

Define translation function Z as a function $Z : X \to Y$ that translates from elements in finite set X to elements in finite set Y. Furthermore, let P be a finite sequence with elements drawn from set X. Then let's define translation of a sequence by $Z(P)$ which is done by replacing each element x_i in sequence P with its translation $y_i = Z(x_i)$. So $Q = Z(P)$ is a finite sequence with same length as P with elements drawn from set Y.

Andrew has two sequences S and T of Alien DNA (both of length n). S and T have elements which are taken from sets A and B, respectively. Andrew has a target alphabet C which is a set of size k. Andrew wants to construct translation functions $F : A \to C$ and $G : B \to C$ such that the longest common prefix (LCP) of $F(S)$ and $G(T)$ is as long as possible. However, he also needs to ensure that neither $F(S)$ nor $G(T)$ are simply the same element from C repeated n times. These sequences are called invalid.

For example, suppose we have

$$A = \{a, b, c\} \text{ and } S = (a, b, c, a)$$

$$B = \{x, y\} \text{ and } T = (x, y, x, y)$$

and C is a set of 2 elements, $\{g, h\}$.

Now consider F which sends $a \to g$, $b \to h$ and $c \to g$ and G which sends $x \to g$ and $y \to h$. These functions give us $F(S) = (g, h, g, g)$ and $G(T) = (g, h, g, h)$ which have an LCP of 3 (which is the answer).

Another possible pair of translation functions F which sends $a \to h$, $b \to h$, $c \to h$ and G which sends $x \to h$, $y \to h$ gives us $F(S) = (h, h, h, h)$ and $G(T) = (h, h, h, h)$, which gives an LCP of 4, but both $F(S)$ and $G(T)$ are now invalid, so we should not consider this case.

Input

The input will contain a single test case.

Elements of A and B are denoted here by non-negative integers.

The first line contains two integers n ($1 \le n \le 200\,000$) and k ($1 \le k \le 400\,000$). The second line contains the sequence $S = (s_1, s_2, \ldots, s_n)$. The sequence is a represented as a list of n integers s_i ($0 \le s_i < 200\,000$). Each of these integers represent an element from alphabet A. The third line contains the sequence $T = (t_1, t_2, \ldots, t_n)$. The sequence is represented as a list of n integers t_i ($200\,000 \le t_i < 400\,000$). Each of these integers represent an element from the alphabet B.

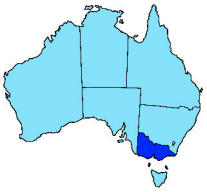
Output

Output maximum length of longest common prefix of translated strings $F(S)$ and $G(T)$. If all translation functions F and G cause $F(S)$ or $G(T)$ to be invalid, output `invalid` instead.

Sample Input and Output

Sample Input	Sample Output
3 100 0 0 7 200000 200001 200001	1

Sample Input	Sample Output
2 1 8 9 200009 200008	invalid



J: Jumbling Cards

Time Limit: 3 second(s)

When casinos shuffle decks of cards, they need to ensure that the shuffling algorithm is very effective. One such algorithm for shuffling is to fix a permutation and then repeatedly place a deck through this permutation. The *productivity* of a permutation is defined as the number of steps that it takes for the deck to get back to its original position.

For example, consider the permutation:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 2 & 1 \end{pmatrix}.$$

For those of you who do not know, this notation means that the *new* card in the first position is the *old* card in the third position. Similarly, the new card in the second, third and fourth position is the one that was in the fourth, second and first position, respectively. In the output for this problem, we will ignore the first line in this notation.

Step	Before Permutation	After Permutation
1	ABCD	CDBA
2	CDBA	BADC
3	BADC	DCAB
4	DCAB	ABCD

So the productivity of this permutation is 4. It turns out that this productivity is the largest that can be achieved by any permutation of size 4. Given an n, I need you to give me a permutation who maximizes the productivity over all permutations of length n.

Input

The input will contain a single test case.

The input will contain exactly one integer n ($1 \leq n \leq 50$).

Output

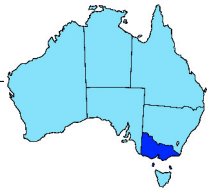
Output a permutation of length n whose productivity is maximal across all permutations of length n. If there are multiple solutions, any will do. Output the permutation on one line as a list of space separated integers. The line must only contain the numbers $1, 2, \ldots, n$.

Sample Input and Output

Sample Input	Sample Output
4	3 4 2 1

Sample Input	Sample Output
3	2 3 1

This Page Is Intentionally (Almost) Blank.



K: Kind Words

Time Limit: 2 second(s)

Betty loves to create new words. However, she cannot distinguish the letter 'O' with the digit '0', so she never uses 'O' to create words. One day, she learnt the concept of Edit Distance between words, which is the minimum number of operations needed to change a word A into another word B. There are 3 valid operations:

1. Insert a letter into A at any location, including the position before the first letter and the last letter.
2. Delete a letter from A.
3. Change a letter in A to another letter.

Betty is exited about this new concept, and decided to play with Lucy. For each word A that Betty created, she wonders if Lucy can create another word B of the same length, and has edit distance d from A. However, since Lucy has no interest in words, she needs you to write a program to help her.

Input

The input will contain a single test case.

The first line contains an integer d ($0 \leq d \leq 1000$) denoting the edit distance we are interested in. The second line contains a non-empty word A. A will only contain uppercase letters (except 'O') and will have a length at most 1000.

Output

For each test case, output a word B of the same length with A, and has edit distance exactly d from A. B must have only uppercase letters (including 'O'). If there are multiple valid words, output the one that is the maximum in lexicographic order. If there are no such words that are exactly an edit distance of d from A, output `None`.

Sample Input and Output

Sample Input	Sample Output
3 ABCD	ZZZD

Sample Input	Sample Output
3 YZYZ	ZZZY

Sample Input	Sample Output
5 ABCD	None

This Page Is Intentionally (Almost) Blank.