ACM ICPC SOUTH PACIFIC REGIONALS

NOVEMBER 26, 2016

Contest Problems

A: Anticlockwise Motion
B: Balloon Warehouse
C: Crazy Rotations
D: Dendroctonus
E: Election Frenzy
F: False Intelligence
G: Graphics Design
H: Hilbert's Hash Browns
I : Intuidiff II
J : Just Terraffic!
K: Kiwis vs Kangaroos

This contest contains eleven problems. The top teams will advance to the ACM-ICPC World Finals 2017.

Judging team:
Darcy Best, Mike Cameron-Jones, Malcolm Corney, Tim French, Walter Guttmann, Andrew Haigh, Henning Koehler, Richard Lobb, Kourosh Neshatian, Evgeni Sergeev, Kevin Tran, Max Ward-Graham

Problem A
Anticlockwise Motion
Time limit: 1 second

You have come up with an idea for a board game. The game is played on a board that is made up of $32\,001 \times 32\,001$ numbered squares. The centre square contains the number 1 and the other numbers are arranged in an anticlockwise spiral outwards (first moving downwards, then to the right, then upwards, then to the left, then downwards again, and so on). Figure A.1 displays the 5×5 squares in the middle of the board and Figure A.2 displays the 21×21 squares in the middle of the board for further clarification. When playing the game, players will only be able to move up, left, down and right. To help work out the rules for the game, you would like to know the shortest distance between two squares on the board using only these moves.

21	20	19	18	17
22	7	6	5	16
23	8	1	4	15
24	9	2	3	14
25	10	11	12	13

Figure A.1: The middle 25 squares.

Input

The input consists of a single line containing two integers a ($1 \leq a \leq 10^9$), which is the starting square, and b ($1 \leq b \leq 10^9$), which is the ending square.

Output

Display the shortest distance between a and b.

Sample Input 1	Sample Output 1
`12 2`	`2`

Sample Input 2	Sample Output 2
`16 24`	`6`

421	420	419	418	417	416	415	414	413	412	411	410	409	408	407	406	405	404	403	402	401
422	343	342	341	340	339	338	337	336	335	334	333	332	331	330	329	328	327	326	325	400
423	344	273	272	271	270	269	268	267	266	265	264	263	262	261	260	259	258	257	324	399
424	345	274	211	210	209	208	207	206	205	204	203	202	201	200	199	198	197	256	323	398
425	346	275	212	157	156	155	154	153	152	151	150	149	148	147	146	145	196	255	322	397
426	347	276	213	158	111	110	109	108	107	106	105	104	103	102	101	144	195	254	321	396
427	348	277	214	159	112	73	72	71	70	69	68	67	66	65	100	143	194	253	320	395
428	349	278	215	160	113	74	43	42	41	40	39	38	37	64	99	142	193	252	319	394
429	350	279	216	161	114	75	44	21	20	19	18	17	36	63	98	141	192	251	318	393
430	351	280	217	162	115	76	45	22	7	6	5	16	35	62	97	140	191	250	317	392
431	352	281	218	163	116	77	46	23	8	1	4	15	34	61	96	139	190	249	316	391
432	353	282	219	164	117	78	47	24	9	2	3	14	33	60	95	138	189	248	315	390
433	354	283	220	165	118	79	48	25	10	11	12	13	32	59	94	137	188	247	314	389
434	355	284	221	166	119	80	49	26	27	28	29	30	31	58	93	136	187	246	313	388
435	356	285	222	167	120	81	50	51	52	53	54	55	56	57	92	135	186	245	312	387
436	357	286	223	168	121	82	83	84	85	86	87	88	89	90	91	134	185	244	311	386
437	358	287	224	169	122	123	124	125	126	127	128	129	130	131	132	133	184	243	310	385
438	359	288	225	170	171	172	173	174	175	176	177	178	179	180	181	182	183	242	309	384
439	360	289	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	308	383
440	361	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	382
441	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381

Figure A.2: The middle 441 squares.

Problem B
Balloon Warehouse
Time limit: 1 second

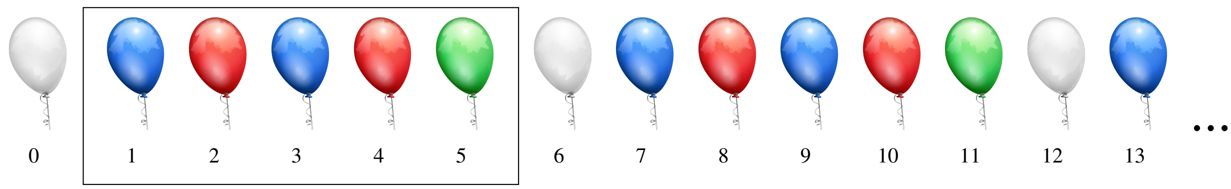


Figure B.1: An infinitely long line of balloons

Darcy has taken over management of a balloon warehouse. Unfortunately, it stocks an infinite supply of balloons, so he has requested your help! At the start of today, the warehouse consists of an infinitely long line of white balloons. In this problem, we describe a coloured balloon by an integer whose value uniquely represents a particular colour, so initially (taking 0 to mean a white balloon) the contents of the line is given by

$$0 \quad 0 \quad 0 \quad \ldots$$

Specific balloons are identified by their 0-indexed position in the line starting from the front of the line $(0, 1, 2, \ldots)$.

Throughout the day, Darcy receives n deliveries of balloons from his supplier. The ith delivery consists of an infinite number of balloons of colour y and comes with the instruction (x, y). If there are any balloons of colour x in the line, then he should insert exactly one balloon of colour y into his infinitely long line immediately after each balloon of colour x already present in the line. Otherwise, he will send the balloons of colour y back.

After all the deliveries of the day are completed, Darcy receives one final instruction from his supplier to check if he has properly followed the instructions he was given: what colour are all the balloons in the positions between l (inclusive) and r (exclusive) in the line? You should help Darcy answer this question.

Consider an example day with four deliveries $(0, 1), (1, 3), (0, 1), (1, 2)$ (for example, it may be that 1 means a blue balloon, 2 means a red balloon and 3 means a green balloon; 0 is white as above). At the end of the day, Darcy's line resembles the line of balloons in Figure B.1; we describe it by

$$0 \quad 1 \quad 2 \quad 1 \quad 2 \quad 3 \quad 0 \quad 1 \quad \ldots$$

If he is asked for $\ell = 1$ and $r = 6$, he should report the numbers `1 2 1 2 3` corresponding to blue, red, blue, red and green balloons in those positions in order.

Input

The first line of the input contains three integers n ($1 \leq n \leq 200\,000$), which is the number of deliveries, ℓ and r ($0 \leq \ell < r \leq 10^6$ and $r - \ell \leq 100\,000$), which are the positions between which to report the balloons' colours at the end of the day.

The next n lines describe the deliveries. Each of these lines contain two distinct integers x ($0 \leq x < 200\,000$) and y ($0 \leq y < 200\,000$), which is the instruction for this delivery.

Output

Display the colours of the balloons between positions ℓ (inclusive) and r (exclusive).

Sample Input 1	Sample Output 1
4 1 6 0 1 1 3 0 1 1 2	1 2 1 2 3

Sample Input 2	Sample Output 2
1 101000 101008 0 1	0 1 0 1 0 1 0 1

Problem C
Crazy Rotations
Time limit: 1 second

Cynthia has a long line of n coloured lights. These lights will be animated by rotating the colours of the lights. To rotate the colours by k spots, the colour of light i at time t is the same as the colour of light $(i - k) \mod n$ at time $t - 1$. When Cynthia does this, there may be several lights that change colour. The *craziness* of a rotation by k spots is the number of lights that change colour. For example, in Figure C.1, rotating the colours by one spot changes the colour of six lights, so the craziness of this rotation is 6.

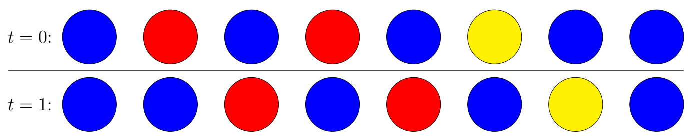


Figure C.1: Rotation by one spot.

Cynthia wishes to create an *insane sequence*, which is a sequence of $n - 1$ different rotations so that the craziness never decreases. To be specific, an insane sequence is a permutation $(p_1, p_2, \ldots, p_{n-1})$ of the first $n - 1$ positive integers such that the craziness of a rotation by p_{i-1} spots is not more than the craziness of a rotation by p_i spots for all i $(2 \leq i < n)$. Figure C.2 is an example of an insane sequence.

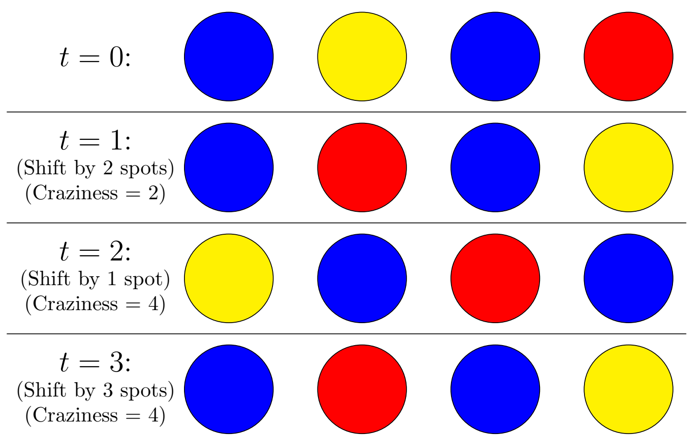


Figure C.2: An insane sequence of rotations (with the permutation $2, 1, 3$).

In Figure C.2, a shift of 1 is in the second location ($t = 2$) of the insane sequence. Given the initial colours of the lights and an integer p, what is the smallest positive integer that can be in the pth location ($t = p$) of an insane sequence?

Input

The first line of input contains two integers n $(2 \leq n \leq 500\,000)$, which is the number of lights, and p $(1 \leq p < n)$, which is the location in the sequence that we are interested in.

The second line contains a string of length n, which is the initial configuration of the lights. Each light's colour is identified by a single character (`R` for red, `B` for blue or `Y` for yellow).

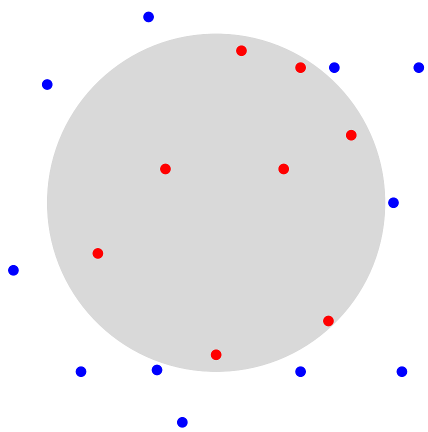
Output

Display the smallest number that could be in the pth location of an insane sequence.

Sample Input 1	Sample Output 1
4 2 BYBR	1

Sample Input 2	Sample Output 2
8 3 BRBRBYBB	4

Problem D
Dendroctonus
Time limit: 1 second



Mountain pine beetles (*Dendroctonus ponderosae*) are small pests that bore into trees and cause a huge amount of damage. Recently, a large increase in their population has occurred and scientists would like some more information about the origin of the outbreak(s). In particular, they want to know if there was a single outbreak or multiple outbreaks. If there is more than one outbreak, then they must raise the alert level.

For each outbreak, the beetles start at a single location (known as the *initial infection point*) and slowly work their way outwards. If a tree is inside the infection area, then it is infected. If a tree is outside of the infection area, then it is not infected. If a tree is on the boundary of the infection area, then it may or may not be infected. The infection area is always a circle centred at the initial infection point.

Given the locations of the infected and non-infected trees, the scientists need you to determine if there is enough evidence to raise the alert level.

Input

The first line of the input contains a single integer n ($1 \leq n \leq 100$), which is the number of trees.

The next n lines describe the trees. Each of these lines contains two integers x ($-250 \leq x \leq 250$) and y ($-250 \leq y \leq 250$), which is the location of the tree, as well as a single character p (`I` or `N`), denoting if the tree is infected or not. If p is `I`, then the tree is infected. If p is `N`, then the tree is not infected. Trees are single points on the plane. Note that the initial infection point for an outbreak does not need to be a tree and does not have to be at an integer location. The n trees are at distinct locations.

Output

If it is guaranteed that there is more than one outbreak, display `Yes`. Otherwise, display `No`.

Sample Input 1	Sample Output 1
7 0 0 I 1 0 I 0 1 I 4 4 N 4 -4 N -4 4 N -4 -4 N	No

Sample Input 2	Sample Output 2
10 5 1 I 5 -1 I 6 4 N 6 -4 N 0 2 N 0 -2 N -5 1 I -5 -1 I -6 4 N -6 -4 N	Yes

Problem E
Election Frenzy
Time limit: 1 second


Source: Reboot Wiki

Checks and balances are some of the most important parts of any democratic government system. After a long campaign, the election between Phong, from the Sprites Party, and Megabyte, from the Viruses Party, has just finished. The only thing left to do is count the votes and declare a winner.

The votes are counted in a room with t tables. At each table, there is a person counting votes. These people are called *counters*. Members from the two political parties are allowed to be in the room to ensure that the counting is done fairly. These people are called *scrutineers*. In a perfect system, one scrutineer from each political party would be present at each table. Unfortunately, this is not possible since there is only enough room for one scrutineer per table.

We need your help in deciding how to assign the scrutineers to the tables. Each table must have exactly one scrutineer: from either the Sprites or the Viruses. Some of the tables are close to one another. This means that a scrutineer will be able to monitor the counting at their table and all surrounding tables. An assignment of scrutineers is considered fair if every table is monitored by at least one Sprite and at least one Virus.

The counter at each table was asked to submit a list of all tables that can be seen from their table. For some reason, some of the counters submitted a list of the tables that *can* be seen from their table and some of the counters submitted a list of tables that *cannot* be seen from their table.

Given this information, find a fair assignments of scrutineers.

Input

The first line of input contains a single integer t ($1 \leq t \leq 200\,000$), which is the number of tables.

The next t lines describe the lists that the counters submitted for each table. Each of these lines starts with a letter p (either `C` or `N`), which is the type of the list, and an integer k ($0 \leq k \leq t-1$), which is the length of the list. This is followed by k distinct integers $a_1, \ldots, a_k$ ($1 \leq a_i \leq n$), which are the items on the list. If $p = \texttt{C}$, then each table a_i can be monitored by the scrutineer at this table and all other tables cannot be monitored. If $p = \texttt{N}$, then each table a_i cannot be monitored by the scrutineer at this table and all other tables can be monitored. The list does not contain its own table number since the scrutineer can always monitor the table they are sitting at.

The first table in the input is table 1, the second table is table 2, and so on. It is guaranteed that if table x can monitor table y, then table y can monitor table x. The total length of all lists is at most $500\,000$.

Output

Display any fair assignment of scrutineers. The assignment should be described as a string of length t containing only the letters `S`, for a Sprite scrutineer, and `V`, for a Virus scrutineer. The first letter of the string is the scrutineer for table 1, the second letter is the scrutineer for table 2, and so on. If there are multiple solutions, display any of them. If there is no such assignment, display `Impossible` instead.

Sample Input 1	Sample Output 1

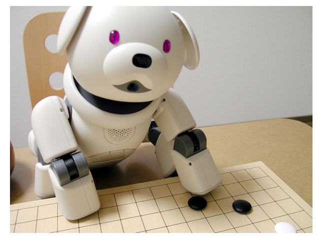
`5`
`C 3 3 4 5`
`C 3 3 4 5`
`C 2 1 2`
`C 2 1 2`
`C 2 1 2`
	`SSVVV`

Sample Input 2	Sample Output 2

`4`
`C 2 2 4`
`N 1 4`
`N 2 1 4`
`C 1 1`
	`SSVV`

Sample Input 3	Sample Output 3
`1` `N 0`	`Impossible`

Problem F
False Intelligence
Time limit: 1 second


Source: Hiroaki Maeda, Flickr CC BY-ND 2.0

Artificial Intelligence is taking over the world, or at least planning to do so soon. Machines will become so clever they can win all games, answer all your questions and make all decisions for you. Because this sounds quite scary, it has been proposed to give the machines a bit of human touch. This feature will make them pretend to not always know the answer to a question, but to express uncertainty from time to time.

Of course, secretly, the machines will still follow precise rules – those of three-valued logic invented by Jan Łukasiewicz. It adds to the two values F (false) and T (true) of Boolean logic a third value U (uncertain) and extends the logic operators as shown in the first three of the following tables:

$\wedge$	F	U	T
F	F	F	F
U	F	U	U
T	F	U	T

AND

$\vee$	F	U	T
F	F	U	T
U	U	U	T
T	T	T	T

OR

$\rightarrow$	F	U	T
F	T	T	T
U	U	T	T
T	F	U	T

IMPLIES

$\equiv$	F	U	T
F	T	F	F
U	F	T	F
T	F	F	T

EQUALS

For example, F $\vee$ T $=$ T as in Boolean logic, T $\wedge$ U $=$ U and F $\rightarrow$ U $=$ T. Some functions cannot be expressed using the three operators AND, OR and IMPLIES. For example, $f(x, y) = x \rightarrow x$ is the constant T function, but there is no expression $g(x, y)$ in terms of x, y, AND, OR and IMPLIES which is constant F.

Let us add the operator EQUALS shown in the right-most of the above tables. It captures equality of its two arguments by returning T if they are the same and F otherwise. Your task is to determine whether a given function $g(x, y)$ can be expressed in terms of x, y, AND, OR, IMPLIES and EQUALS. This is important so the machines know their own limits.

Input

The first line of the input contains an integer n ($1 \leq n \leq 20\,000$), which is the number of functions you have to consider. This is followed by n function descriptions.

Each function description consists of four lines. The first of these lines is empty. The remaining three lines describe a function g as a table of its values $g(x, y)$. The table has three rows and three columns, corresponding to the values F, U, T of x and y, respectively. Its layout is like that of the tables shown above.

Output

For each function description in the same order as in the input, display one line containing either `definable` or `undefinable`. Display `definable` if the given function $g(x, y)$ can be expressed in terms of x, y, AND, OR, IMPLIES and EQUALS. Display `undefinable` if $g(x, y)$ cannot be expressed this way.

Sample Input 1	Sample Output 1

<pre>3
 	
F F F
F U U
F U T
 	
T T T
U U T
F U T
 	
T T T
U U U
F F F</pre>
	<pre>definable
	definable
	undefinable</pre>

Problem G
Graphics Design
Time limit: 1 second

Source: Pexels

You are in a graphics design course. You and the rest of your class need to finish your final projects. The instructor has several items (cameras, camcorders and super computers) that the students will use to finish their projects. Each project consists of several subprojects that must be completed in order. These subprojects may or may not be different for each student. Each subproject may need a camera, a camcorder and/or a super computer (8 possibilities). The instructor has given each subproject a number, its priority.

A subproject is eligible to be started if all items that are needed for that subproject are available and the student has fully completed all of their subprojects before this one. Out of all eligible subprojects, the one with the highest priority will be started first. This student will borrow the items needed to complete the subproject and then return them after the subproject is finished.

If, after a subproject has started and the items have been borrowed, there still remain eligible subprojects, the eligible subproject with the highest priority is started. This means that it is possible that several subprojects are started at the same time (see Sample Input 1). It is also possible that several students are waiting for an item to be returned, and so there are no eligible subprojects at a specific moment in time (see Sample Input 2). Note that the order of each student's subprojects is fixed. That is, they must complete their first subproject first, then their second subproject, and so on, even if the priorities of their subprojects are not in decreasing order (see Sample Input 3).

If a student starts a subproject at time x and it takes t time units to complete, then the subproject is considered finished at time $x + t$ and the student must return the borrowed items. This means that the items borrowed for that subproject are available to be borrowed again at time $x + t$ and that the student is able to start their next subproject at time $x + t$ (subject to availability of the items and the subproject's priority, as specified above). Each camera, camcorder and super computer can only be used by one student at a time. Students must do the subprojects in order, even if they have the items needed for a later subproject but not the current one.

For each student, you are to compute the time that they finish their last subproject.

Input

The first line of input contains a single integer n ($1 \le n \le 1\,000$), which is the number of students in the class. The second line contains 3 integers a ($1 \le a \le 1\,000$), which is the number of cameras available, b ($1 \le b \le 1\,000$), which is the number of camcorders available, and c ($1 \le c \le 1\,000$), which is the number of super computers available. The third line contains n integers $d_1, \ldots, d_n$ ($1 \le d_i \le 250$), which are the number of subprojects that each student needs to complete for their project.

This is followed by d_i lines for each student i in the class, one line for each subproject in the order they must be completed by that student. Each line for a subproject consists of an integer t ($1 \le t \le 1\,000\,000$), which is the amount of time it takes to complete the subproject, an integer p ($1 \le p \le 1\,000\,000$), which is the priority of the subproject, and between zero and three distinct strings. Each of these strings will be one of `Camera`, `Camcorder` or `Computer`.

All priorities will be distinct.

Output

For each student, display the time that they finish their last subproject, in the same order that the students are given in the input.

Sample Input 1

```
3
1 1 1
1 1 1
4 1 Camera
4 2 Camcorder
4 3 Computer
```

Sample Output 1

```
4 4 4
```

Sample Input 2

```
3
1 1 1
1 1 1
3 3 Computer
4 2 Computer
5 1 Camera Computer
```

Sample Output 2

```
3 7 12
```

Sample Input 3

```
2
1 1 1
2 1
1 1 Computer
1 3 Computer
1 2 Computer
```

Sample Output 3

```
3 1
```

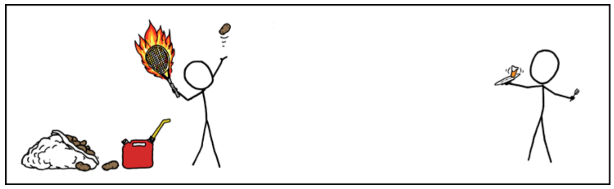
Sample Input 4

```
3
2 2 2
2 1 3
2 3 Camera
5 1 Camera Camcorder
3 2 Camcorder Computer
1 6 Camera Camcorder Computer
1 5 Camera Camcorder Computer
1 4 Camera Camcorder Computer
```

Sample Output 4

```
8 3 3
```

Problem H
Hilbert's Hash Browns
Time limit: 1 second



Source: xkcd.com, number 421, titled "Making Hash Browns"

You may have heard of Hilbert's Hotel. It has infinitely many rooms, and can cater for infinitely many guests. Each room has a number so even if it seems full, room for an extra guest can be found by asking every guest to move from room i to room $i + 1$, thus freeing up room 0. Unfortunately, the restaurant attached to the hotel, Hilbert's Hash Browns, is not infinite, although the number of tables is very large.

To make matters worse, the waiter is very lazy. Rather than keep a record of which tables are available, he just assigns people to a table using a simple formula: when someone arrives at the restaurant, the waiter asks them their hotel room number. He raises this number to the power of p, and adds q. Since this gives very big numbers and there are only n tables (numbered 0 to $n - 1$), the waiter takes the remainder when dividing by n, and directs the customer to that table. Of course, sometimes the table is already occupied, in which case the customer goes away hungry.

The waiter uses a different value for p and q every day and he has noticed that some days a table never gets used no matter how many patrons turn up. For example if $n = 3$, $p = 2$ and $q = 1$, then table number 0 will never be used because there is no number x such that $x^2 + 1 \equiv 0 \mod 3$.

The waiter is lazy but he would like to appear competent. Can you help him determine the maximum number of tables that can be assigned for given values of p, q and n?

Input

The input consists of a single line containing three integers p ($1 \le p < 2^{31}$), q ($0 \le q < 2^{31}$) and n ($2 \le n < 2^{31}$).

Output

Display the maximum number of tables that could be used.

Sample Input 1	Sample Output 1
`2 3 5`	`3`

Sample Input 2	Sample Output 2
`4 1 15`	`4`

This page is intentionally left (almost) blank.

Problem I
Intuidiff II
Time limit: 1 second

You may recall *Intuidiff*, the alternative to `diff` that we asked you to help develop at the Divisionals contest. Intuidiff gives an intuitive way to track changes in two files: the original document and the modified document.

We are now onto the next stage of development of Intuidiff. After some preprocessing, the original document has been broken up into several non-overlapping substrings, and each of these has been assigned a different colour (for example, see the 'Before' paragraph in Figure I.1). Then, in the modified document, the substrings are coloured using the same colours as those in the original document (for example, see the 'After' paragraph in Figure I.1). This allows us to see how large substrings have moved in the document. Substrings with the same colour may appear multiple times in the 'After' section, but only once in the 'Before' section. For example, the substring "et dolore magna aliqua." appears twice in the modified document of Figure I.1.

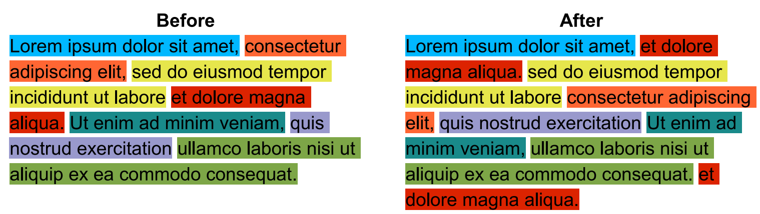


Figure I.1: A full colouring from the Intuidiff program.

While pretty, this full colouring might be overwhelming for some users. Also, it is distracting if every character is highlighted in the modified document. Attention should be focused only on the changes.

Therefore, for the next stage of development of Intuidiff, we must select which substrings to highlight. A full colouring of the document has already been decided on. We must select substrings from the full colouring in such a way that the non-highlighted characters are in the same order as in the original document.

The characters that are *not* highlighted must be in the same order as in the original document. The selection of substrings is done in such a way that the number of non-highlighted characters in the modified document is maximised (for example, see Figure I.2).

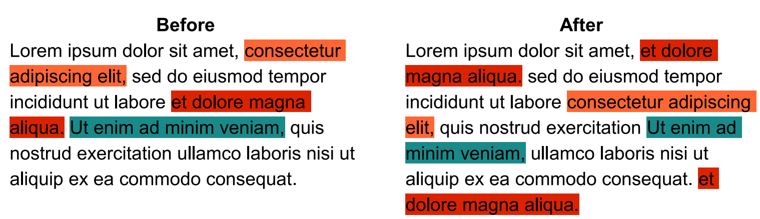


Figure I.2: A desired colouring from the Intuidiff program.

Input

The first line of input contains a single integer n ($1 \leq n \leq 100\,000$), which is the number of coloured substrings in the full colouring in the modified document. The next n lines describe the coloured substrings in the modified document. Each of these lines contain two integers a and b ($0 \leq a \leq b \leq 10^9$), which is the substring of the original document running from the character at index a to the character at index b, inclusive.

No two substrings will partially overlap. That is, if two substrings share any common indices, then the substrings will be identical.

Output

Display the number of characters that are not highlighted by Intuidiff in the modified document.

Notes

Sample Input 1 is the ‘After’ paragraph from Figure I.1. The 154 non-highlighted characters are shown in Figure I.2.

Sample Input 1	Sample Output 1
8 0 27 99 122 56 98 28 55 148 174 123 147 175 230 99 122	154

Sample Input 2	Sample Output 2
5 1 1 10 11 5 7 3 4 10 11	6

Problem J
Just Terraffic!
Time limit: 1 second

The local council is recording traffic flow using a pressure pad laid across the road. The pressure pad tracks whenever the wheels on an axle of a vehicle cross the pressure pad. The only vehicles using the road are cars with two axles. Each vehicle may or may not have a single-axle trailer attached to it. When a car crosses the pressure pad, two times are recorded: one when the front wheels cross and another when the rear wheels cross. If the car is towing a trailer an additional time is recorded when the trailer wheels cross. Given a sequence of times from the recorder, the council wishes to know how many cars without trailers crossed the pad and how many cars with trailers crossed it.

Obviously, there is some ambiguity. For example, a sequence of 6 recordings could be three cars without trailers or two cars with trailers. To reduce such ambiguity, we will make the following two assumptions:

1. Any two successive times with a difference less than or equal to 1000 ms must belong to the same vehicle.
2. Any two successive times with a difference greater than or equal to 2000 ms must be from different vehicles.

Given a sequence of times, determine the number of cars with and without a trailer.

Input

The first line of the input contains a single integer n ($1 \leq n \leq 300\,000$), which is the number of times the pressure pad was triggered. The second line contains n distinct integers $t_1, \ldots, t_n$ ($0 \leq t_i < 2^{30}$) in increasing order, the times that the pressure pad was triggered. The times are in milliseconds.

Output

Display the number of cars with and without trailers. If the number of cars of each type can be uniquely determined, then display two lines of the form

```
Cars without trailers: X
Cars with trailers: Y
```

If there is no interpretation of the times that is consistent with the assumptions, then display `Impossible`. If there are multiple interpretations of the times that give different numbers of cars with and without trailers, then display `Ambiguous`.

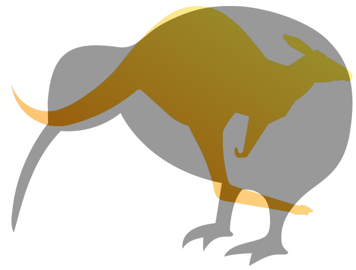
Sample Input 1	Sample Output 1
7 10 200 5000 6100 7200 8300 9400	Cars without trailers: 2 Cars with trailers: 1

Sample Input 2	Sample Output 2
6 0 1100 2200 3300 4400 5500	Ambiguous

Sample Input 3	Sample Output 3
4 0 1000 2000 3001	Impossible

This page is intentionally left (almost) blank.

Problem K
Kiwis vs Kangaroos
Time limit: 1 second



Source: Pixabay

As everyone in this room knows, there is an ongoing feud between the national animals of Australia and New Zealand: the kangaroos and the kiwis! Centuries ago, kangaroos and kiwis were great friends and played happily with one another all day and all night. Over time, they both became too tired to play 24 hours per day, so they had to decide what time of day they should play: the day time (preferred by the kangaroos) or the night time (preferred by the kiwis). This disagreement started the feud that has lasted ever since.

The king of the kangaroos and the queen of the kiwis are now at a secret meeting under the Tasman Sea in hopes of stopping this feud. After hours of negotiations, they decide that the only fair way to settle the feud is to get a neutral third party to decide. And that third party is you!

You have come up with the following algorithm to determine the winner:

1. The kangaroo king and kiwi queen must agree upon a secret phrase.
2. Each animal is given a *key*: `KANGAROO` for the kangaroos and `KIWIBIRD` for the kiwis.
3. For each letter in the secret phrase, count the number of times that letter appears in the animal's key. Uppercase and lowercase should be treated as the same.
4. The total score for each animal is the sum of these counts.
5. The animal with the higher total score is the winner.

For example, if the secret phrase is "Australia", then the kangaroos' score would be 7 and the kiwis' score would be 4. In this case, the kangaroos would win the feud.

	A	u	s	t	r	a	l	i	a	Total
`KANGAROO`	2	0	0	0	1	2	0	0	2	7
`KIWIBIRD`	0	0	0	0	1	0	0	3	0	4

Given the secret phrase, who wins the feud?

Input

The input will consist of text on one line, the secret phrase. The secret phrase will be non-empty and contain only uppercase letters and lowercase letters. The secret phrase will contain at most 100 characters.

Output

If there is a winner of the feud, display the winner's name: either `Kangaroos` or `Kiwis`. If there is a tie, display `Feud continues` instead.

Sample Input 1	Sample Output 1
`BattlestarGalactica`	`Kangaroos`

Sample Input 2	Sample Output 2
`RiddlesInTheDark`	`Kiwis`

Sample Input 3	Sample Output 3
`Hexadecimal`	`Feud continues`

This page is intentionally left (almost) blank.