In the strong room of ABC bank there are N vaults in a row. The amount of money inside each vault displayed on the door. You can empty any number of vaults as long as you do not empty more than 2 out of any 5 adjacent vaults. For example, of the vaults 1, 2, 3, 4, 5, if you empty 4 and 5, then you can not empty any of the vaults 6, 7 or 8 but you can empty 9th vault. If you attempt to break more than 2 of any 5 adjacent vaults, an alarm sounds and the sentry, a sharp shooter will kill you instantly with his laser gun!  
  
The output is the maximum amount of money that can be emptied without sounding the alarm

**Input**   
  
The first line contains an integer N which is the number of vaults. The next line has a sequence of positive integers of length N, giving the amount of cash in the vaults in order  
  
**Output**   
  
The maximum amount of money that can be looted without sounding the alarm.  
  
**Constraints**  
  
N≤50, Amount in each vault ≤50000

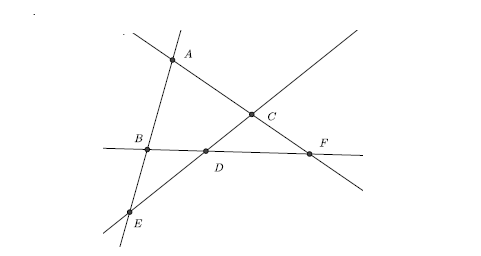
**Example 1**  
  
Input:   
9   
1000, 2000, 1000, 5000, 9000, 5000, 3000, 4000, 1000

Output:   
15000

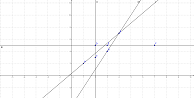
Explanation:   
One possible set of vaults to be looted to get the maximum possible are vaults 4, 5, 9 are looted, giving a total loot of (5000+9000+1000)=15000. Hence the output is 15000.

**Example 2**  
  
Input:   
10   
5000,7000,3000,5000,9000,7000,6000,4000,7000,5000  
  
Output:   
28000  
  
Explanation:   
One possible set of vaults to be looted are (2, 5, 9, 10) Note that no set of five consecutive vaults has more than two vaults looted. The total looted is (7000 + 9000 + 7000 + 5000=28000). The out put is hence 28000.

**Problem : Special Triangles**

When we have 4 lines in the plane, the maximum number of triangles formed by them is 4 as shown below: Of the four triangles ABF, CDF, BDE, ACE, triangles BDE and CDF are somewhat "special". They do not have another of the given lines go through their interiors.  
  
  
  
Given n lines in the plane, the objective is to count the number of special triangles formed by them. You may assume that no three lines will be concurrent (go through the same point). Each line will be specified by two points (x and y coordinates of two points on the line).  
  
Note that three lines may not form a triangle, let alone a special triangle, if two of them are parallel lines.

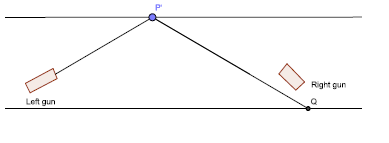
**Input**  
  
The first line will have a single integer, N, representing the number of lines in the input.  
  
The next N lines will have four comma separated numbers representing x1, y1, x2, y2, (the x and y coordinates of two points on the line.)  
  
**Output**  
  
The output is a single line giving the number of special triangles formed.   
  
**Constraints**  
  
N≤20  
  
Each of the x and y coordinates of the points defining any line lies between -10 and 10.

**Example 1**  
  
Input   
4   
1,2,4,7   
2,3,2,5  
3,5,7,5  
4,7,3,4  
  
Output   
2  
  
Explanation   
  
  
Line I is defined by the points (1,2) (A) and (4,7) (B). Similarly, line II is defined by C (2,3) and D (2,5), line III by E (3,5) and F(7,5) and line IV by B (4,7) and G (3,4).  
  
Of the four triangles formed by the four lines, the ones by (I,II,III) and (I,III,IV) are special.  
  
(I,II,IV) has III going through the interior, while (II,III,IV) has I going through the interior.  
  
As only two triangles are special, the output is 2.

**Example 2**  
  
Input   
4   
2,2,3,2   
2,3,3,3   
2,4,3,4   
2,5,3,5  
  
Output   
0

Explanation   
As all lines are parallel to the x axis, there are no triangles, and hence no special triangles. The output is 0.

**Problem : Colliding Cannons**

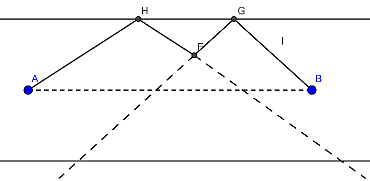
We have seen in many mythological movies, the arrows shot, by the opponents collide midair and one devour the other.   
   
You wanted to simulate a similar situation for the video game you are designing. In the game, the opponents are in a tunnel and have a gun each. They can shoot bullets in any direction (within limits). The roof and floor of the tunnel are perfect surfaces and any object hitting them are bounced off according to the law of reflection (angle of incidence equals the angle of reflection), with unchanged speed. For simplicity, we can assume that the tunnel is a two dimensional horizontal strip. Of course, this being the mythological world, gravity does not exist, and the bullets travel in straight lines at constant spped until being reflected (or they collide).   
  
The two guns are positioned at half the height (h) of the tunnel, at a distance D apart.  
  
The two guns fire simultaneously. The trajectories of the bullets (if extended) will meet at a maximum of one point. They are said to collide if their trajectories meet, and the two bullets arrive at that point within 0.5 seconds of each other.  
  
The shooting angle varies from -85 degrees to 85 degrees from the horizontal. For the left gun, the angles are measured anti-clockwise, and for the right gun they are measured clockwise. Hence, with a positive angle for both, the left gun shoots up and to the right, and the right gun shoots up and to the left.  
  
Write a program to decide whether the bullets shot will collide or not. If they do, determine where will they collide. The coordinate axes for reference have origin at the midpoint of the line joining the guns, X axis along the line joining the guns. Hence, the left gun's coordinates are (-D/2,0) and the right gun's coordinates are (D/2,0). The equations for the top of the tunnel is y=h/2, and of the bottom of the tunnel is y=-h/2.

**Input**   
  
The input has two lines.  
  
The first line has two comma separated positive integers, h and D (the height of the tunnel and the distance between them). The unit of distance measurement is myth units.  
  
The second line has four comma separated numbers (with up to two decimals each) giving LA, LS, RA and RS respectively, where LA, LS denote the angle (in degrees) and speed of firing (in myth units per second) of the left gun and RA, RS denote the angle (in degrees) and speed of firing (in myth units per second) of the right gun.

**Output**   
  
If the two collide (the trajectories meet, and they arrive at the meeting point within 0.5 seconds of each other), the output is a comma separated string of the word Yes and the coordinates of the colliding point.  
  
Yes, x, y  
  
Here, x and y are the coordinates of the collision point, and each must be rounded to two decimal places If they do not collide, the output is the word No.  
  
No

**Constraints**   
  
-85 ≤ LA, RA ≤85   
0 < LS, RS <1000   
0 < h, D < 10000

**Example 1**  
  
Input:   
500,2000   
30,90,40,70  
  
Output:   
Yes,46.20,171.01  
  
Explanation:   
The distance between the guns is 500 myth units, and the height of the tunnel is 2000 myth units. The angle of firing of the left gun is 30 degrees and the speed of the bullet from the let gun is 90 myth units per second. The angle of the right gun is 40 degrees, and the speed of the bullet from the right gun is 70 myth units per second.  
  
The coordinates of the intersection point of the trajectories is (to two decimal places) (46.20,171.01). The time for the left bullet to reach this point is (to two places) 3.80 seconds, and for the right bullet (to two seconds) is 3.80 seconds. As this is within 0.5 seconds of each other, the collision is assumed to have taken place. Hence the output is Yes,46.20,171.01

**Example 2**  
  
Input:   
500,2000   
80,70,85,70  
  
Output:   
Yes,84.19,104.74  
  
Explanation:   
The distance between the guns is 500 myth units and the height of the tunnel is 2000 myth units. The left gun angle is 80 degrees, and the left gun speed is 70 myth units per second. The right gun angle is 85 degrees and the speed is 70 myth units per second also.   
   
The guns shoot, and reflect off the ceiling (at H and G respectively), and the trajectories meet at F. The coordinates of F are (84.19, 104.74). The time taken for the left bullet is 27.49 seconds, and the time for the second bullet is 27.17 seconds. As they arrive within 0.5 seconds of each other, this is considered a collision. Hence the output is Yes,84.19,104.74

**Example 3**  
  
Input   
500,2000   
30,170,50,160  
  
Output   
No  
  
Explanation   
The two trajectories meet at (86.82,194.47), but the left bullet takes 2.29 seconds, and the right bullet takes 1.59 seconds. Hence, they do not pass within 0.5 seconds of each other, and there is no collision.

**Problem : Longest Ski subsequence**

Given a sequence a1, a2, ..... an, of positive integers, a subsequence b1, b2, ...bm of it is said to be a ski sequence if there is a number k<m so that b1 < b2 ...< bk, bk>bk+1...bm. Thus, bk is the maximum number, and must not be in either end (there must an increasing part up to k and a decreasing part after k). For example, in the sequence 1, 2, 10, 3, 7 , 4, two ski sequences are 1,10,3 and 1,2,7,4. The longest ski subsequence is 1,2,10,7,4  
  
The Input consists of N sets of two positive integers each. From these sets, a derived sequence is formed using the following rule  
  
**Rule for forming derived sequence**   
  
If both the integers given are odd, the maximum of this is taken in the derived sequence. If one of them is odd and one of them is even, the even number is taken in the derived sequence. If both are even, the minimum of the two numbers will be taken in to the derived sequence.  
  
The objective is to find the length of the longest Ski subsequence of the derived sequence.

**Input**   
  
The first line of the input has a positive integer N which is the number of sets of integers in the input. Each of the next N lines consists of two (not necessarily distinct) comma separated positive integers.  
  
**Output**  
  
The output is the length of the longest Ski subsequence of the derived sequence. If no ski subsequence exists, the output should be the word Impossible.  
  
**Constraints**  
  
N<=50   
Integers in sets<=10000

**Example 1**  
  
Input:   
5   
1089,3234   
6740,2803   
9243,2638   
4865,4355   
5993,8946  
  
Output:   
3  
  
Explanation:   
The derived sequence is 3324, 6740, 2638, 4865, 8946. The longest ski subsequence is 3324, 6740, 2638, which has 3 elements. Hence the output is 3

**Example 2**  
  
Input:   
5   
5333,4267   
4813,443   
1113,518   
429,437   
119,136  
  
Output:   
Impossible  
  
Explanation:   
The derived sequence is 5333, 4813, 518, 437, 136. As this is a purely descending sequence, we cannot have a ski sequence (as there is no increasing subsequence that is part of it.

**Problem : Prime numbers spelt with prime number of letters**

If you like numbers, you may have been fascinated by prime numbers. These are numbers that have no divisors other than 1 and themselves. If we consider the primes 2 and 3, and write them in words, we write TWO and THREE. Both have a prime number of letters in their spelling. Not all prime numbers have this property.   
  
Write a program to count the number of primes between a given pair of integers (including the given integers if they are primes) that have a prime number of characters when written in words. The blanks are not counted when we write the numbers in words. For example, ONE HUNDRED AND THREE has only 18 characters.

**Input**  
  
One line containing two integers separated by space giving N1 and N2  
  
**Output**   
  
One integer M giving the number of primes P such that N1 <= P <= N2 that are such that when P is written in words, it has a prime number of letters.  
  
**Constraint**   
  
N2 <= 99999

**Example 1**  
  
Input:   
1 10  
  
Output:   
3  
  
Explanation:   
The primes between 1 and 10 and 2, 3, 5 and 7. Of these, 5 written in words is FIVE and has a non prime number of letters and others have prime number of letters (viz TWO, THREE and SEVEN).

**Example 2**   
Input:   
1100 1130  
  
Output:   
1  
  
Explanation:   
The primes between 1100 and 1130 are 1103, 1109, 1117, 1123 and 1129. When these are written in words, we get   
ONE THOUSAND ONE HUNDRED AND THREE   
ONE THOUSAND ONE HUNDRED AND NINE   
ONE THOUSAND ONE HUNDRED AND SEVENTEEN   
ONE THOUSAND ONE HUNDRED AND TWENTY THREE   
ONE THOUSAND ONE HUNDRED AND TWENTY NINE  
  
The count of characters in the above are 29, 28, 33, 35 and 34   
Of these only for 1103 the count of characters is prime.

**Problem : Digital Time 12**

The objective is to form the maximum possible time in the HH:MM:SS format using any six of nine given single digits (not necessarily distinct)

Given a set of nine single (not necessarily distinct) digits, say 0, 0, 1, 3, 4, 6, 7, 8, 9, it is possible to form many distinct times in a 12 hour time format HH:MM:SS, such as 10:36:40 or 01:39:46 by using each of the digits only once. The objective is to find the maximum possible valid time (00:00:01 to 12:00:00) that can be formed using some six of the nine digits exactly once. In this case, it is 10:49:38.

**Input**  
  
A line consisting of a sequence of 9 (not necessarily distinct) single digits (any of 0-9) separated by commas. The sequence will be non-decreasing

**Output**  
  
The maximum possible time in a 12 hour clock (00:00:01 to 12:00:00) in a HH:MM:SS form that can be formed by using some six of the nine given digits (in any order) precisely once each. If no combination of any six digits will form a valid time, the output should be the word - Impossible

**Example 1**  
  
Input:  
  
0,0,1,1,3,5,6,7,7

Output:  
  
11:57:37  
  
The maximum valid time in a 12 hour clock that can be formed using some six of the 9 digits precisely once is 11:57:37

**Example 2**

Input:  
  
3,3,3,3,3,3,3,3,3

Output:  
  
Impossible  
  
No set of six digits from the input may be used to form a valid time.