CCC '18 J2 - Occupy parking

Time limit: 3.0s **Memory limit:** 256M

Canadian Computing Competition: 2018 Stage 1, Junior #2

You supervise a small parking lot which has N parking spaces.

Yesterday, you recorded which parking spaces were occupied by cars and which were empty.

Today, you recorded the same information.

How many of the parking spaces were occupied both yesterday and today?

Input Specification

The first line of input contains the integer N $(1 \leq N \leq 100)$. The second and third lines of input contain N characters each. The second line of input records the information about yesterday's parking spaces, and the third line of input records the information about today's parking spaces. Each of these $2N$ characters will either be `C` to indicate an occupied space or `.` to indicate it was an empty parking space.

Output Specification

Output the number of parking spaces which were occupied yesterday and today.

Sample Input 1

```
5
CC..C
.CC..
```

Sample Output 1

```
1
```

Explanation for Sample Output 1

Only the second parking space from the left was occupied yesterday and today.

Sample Input 2

```
7
CCCCCCC
C.C.C.C
```

Sample Output 2

```
4
```

Explanation for Sample Output 2

The first, third, fifth, and seventh parking spaces were occupied yesterday and today.

CCC '21 J2 - Silent Auction

Time limit: 3.0s **Memory limit:** 1G

Canadian Computing Competition: 2021 Stage 1, Junior #2

A charity is having a silent auction where people place bids on a prize without knowing anyone else's bid. Each bid includes a person's name and the amount of their bid. After the silent auction is over, the winner is the person who has placed the highest bid. If there is a tie, the person whose bid was placed first wins. Your job is to determine the winner of the silent auction.

Input Specification

The first line of input contains a positive integer N, where $1 \leq N \leq 100$, representing the number of bids collected at the silent auction. Each of the next N pairs of lines contains a person's name on one line, and the amount of their bid, in dollars, on the next line. Each bid is a positive integer less than $2\,000$. The order of the input is the order in which bids were placed.

Output Specification

Output the name of the person who has won the silent auction.

Sample Input 1

```
3
Ahmed
300
Suzanne
500
Ivona
450
```

Output for Sample Input 1

```
Suzanne
```

Explanation of Output for Sample Input 1

The highest bid placed was 500 and it was placed by Suzanne. Suzanne wins the silent auction.

Sample Input 2

```
2
Ijeoma
20
Goor
20
```

Output for Sample Input 2

```
Ijeoma
```

Explanation of Output for Sample Input 2

The highest bid placed was 20 and it was placed by both Ijeoma and Goor. Since Ijeoma's bid was placed first, Ijeoma wins the silent auction.

CCC '15 J2 - Happy or Sad

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2015 Stage 1, Junior #2

We often include emoticons in our text messages to indicate how we are feeling. The three consecutive characters `:-)` indicate a happy face and the three consecutive characters `:-(` indicate a sad face. Write a program to determine the overall mood of a message.

Input Specification

There will be one line of input that contains between 1 and 255 characters.

Output Specification

The output is determined by the following rules:

- If the input line does not contain any happy or sad emoticons, output `none`.
- Otherwise, if the input line contains an equal number of happy and sad emoticons, output `unsure`.
- Otherwise, if the input line contains more happy than sad emoticons, output `happy`.
- Otherwise, if the input line contains more sad than happy emoticons, output `sad`.

Sample Input 1

```
How are you :-) doing :-( today :-)?
```

Output for Sample Input 1

```
happy
```

Sample Input 2

```
:)
```

Output for Sample Input 2

```
none
```

Sample Input 3

```
This :-(is str :-(:-a(nge te:-)xt.
```

Output for Sample Input 3

```
sad
```

CCC '07 J2 - I Speak TXTMSG

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2007 Stage 1, Junior #2

Text messaging using a cell phone is popular among teenagers. The messages can appear peculiar because short forms and symbols are used to abbreviate messages and hence reduce typing.

For example, `LOL` means "laughing out loud" and `:-)` is called an emoticon which looks like a happy face (on its side) and it indicates chuckling. This is all quite a mystery to some adults.

Write a program that will continually input a short form and output the translation for an adult using the following translation table:

Short Form	Translation
`CU`	see you
`:-)`	I'm happy
`:-(`	I'm unhappy
`;-)`	wink
`:-P`	stick out my tongue
`(~.~)`	sleepy
`TA`	totally awesome
`CCC`	Canadian Computing Competition
`CUZ`	because
`TY`	thank-you
`YW`	you're welcome
`TTYL`	talk to you later

Input Specification

The user will enter text to be translated one line at a time. When the short form `TTYL` is entered, the program ends. Users may enter text that is found in the translation table, or they may enter other words. All entered text will be symbols or uppercase letters. There will be no spaces and no quotation marks.

Output Specification

The program will output text immediately after each line of input. If the input is one of the phrases in the translation table, the output will be the translation; if the input does not appear in the table, the output will be the original word. The translation of the last short form entered `TTYL` should be output.

Sample Input

```
CCC
:-)
SQL
TTYL
```

Sample Output

```
Canadian Computing Competition
I'm happy
SQL
talk to you later
```

CCC '02 J2 - AmeriCanadian

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2002 Stage 1, Junior #2

Americans spell differently from Canadians. Americans write `neighbor` and `color` while Canadians write `neighbour` and `colour`. Write a program to help Americans translate to Canadian.

Your program should interact with the user in the following way. The user should type a word (not to exceed 64 letters) and if the word appears to use American spelling, the program should echo the Canadian spelling for the same word. If the word does not appear to use American spelling, it should be output without change. When the user types `quit!` the program should terminate.

The rules for detecting American spelling are quite naive: If the word has more than four letters and has a suffix consisting of a consonant followed by `or`, you may assume it is an American spelling, and that the equivalent Canadian spelling replaces the `or` by `our`. Note: you should treat the letter `y` as a vowel.

Sample Input

```
color
for
taylor
quit!
```

Sample Output

```
colour
for
taylour
```

CCC '09 J2 - Old Fishin' Hole

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2009 Stage 1, Junior #2

Fishing habitat and fish species are a resource that must be carefully managed to ensure that they will be there for the future. Accordingly, fishing limits have been established for a particular river based on the population of each species. Specifically, *points* are associated with the fish caught and the total points you catch must be less than or equal to the points allowed for that river.

As an example, suppose each brown trout counts as 2 points, each northern pike counts as 5 points and each yellow pickerel counts as 2 points, and the total points allowed must be less than or equal to 12. One acceptable catch could consist of 3 brown trout and 1 northern pike, but, other combinations would also be allowed.

Your job is to write a program to input the points allocated for a river, and find how many different ways an angler who catches *at least* one fish can stay within his/her limit.

Input Specification

You will be given 4 integers, one per line, representing trout points, pike points, pickerel points, and total points allowed in that order.

You can assume that each integer will be greater than 0 and less than or equal to 100.

Output Specification

For each different combination of fish caught, output the combination of brown trout, northern pike, and yellow pickerel in that order. The combinations may be listed in any order. The last line of output should display the total number of unique ways to catch fish within the established limit.

Sample Input

```
1
2
3
2
```

Sample Output

```
1 Brown Trout, 0 Northern Pike, 0 Yellow Pickerel
2 Brown Trout, 0 Northern Pike, 0 Yellow Pickerel
0 Brown Trout, 1 Northern Pike, 0 Yellow Pickerel
Number of ways to catch fish: 3
```

CCC '00 J2 - 9966

Time limit: 1.0s **Memory limit:** 256M

Canadian Computing Competition: 2000 Stage 1, Junior #2

The digits 0, 1, and 8 look much the same if rotated 180 degrees on the page (turned upside down). Also, the digit 6 looks much like a 9, and vice versa, when rotated 180 degrees on the page. A multi-digit number may also look like itself when rotated on the page; for example 9966 and $10\,801$ do, but 999 and 1234 do not.

You are to write a program to count how many numbers from a given interval look like themselves when rotated 180 degrees on the page. For example, in the interval $[1 \ldots 100]$ there are six: 1, 8, 11, 69, 88, and 96.

Your program should take as input two integers, m and n, which define the interval to be checked, $1 \leq m \leq n \leq 32\,000$. The output from your program is the number of rotatable numbers in the interval.

You may assume that all input is valid.

Sample Input

```
1
100
```

Sample Output

```
6
```

CCC '06 J2 - Roll the Dice

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2006 Stage 1, Junior #2

Diana is playing a game with two dice. One die has m sides labelled $1, 2, 3, \ldots, m$.

The other die has n sides labelled $1, 2, 3, \ldots, n$.

Write a program to determine how many ways can she roll the dice to get the sum 10.

For example, when the first die has 6 sides and the second die has 8 sides, there are 5 ways to get the sum 10:

- 2 + 8 = 10
- 3 + 7 = 10
- 4 + 6 = 10
- 5 + 5 = 10
- 6 + 4 = 10

Input

The input is given as two integers. First, the user will enter in the number m $(1 \leq m \leq 1000)$.

Second, the user will enter the number n $(1 \leq n \leq 1000)$.

Output

The program prints out the number of ways 10 may be rolled on these two dice. Note that in the output, the word `way` should be used if there is only one way to achieve the sum of 10; otherwise, the word `ways` should be used in the output. That is, if there is only one way to get the sum 10, the output should be:

```
There is 1 way to get the sum 10.
```

Sample Input 1

```
6
8
```

Sample Output 1

```
There are 5 ways to get the sum 10.
```

Sample Input 2

```
12
4
```

Sample Output 2

```
There are 4 ways to get the sum 10.
```

CCC '05 J2 - RSA Numbers

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2005 Stage 1, Junior #2

When a credit card number is sent through the Internet it must be protected so that other people cannot see it. Many web browsers use a protection based on "RSA Numbers."

A number is an RSA number if it has exactly four divisors. In other words, there are exactly four numbers that divide into it evenly. For example, 10 is an RSA number because it has exactly four divisors $(1, 2, 5, 10)$. 12 is not an RSA number because it has too many divisors $(1, 2, 3, 4, 6, 12)$. 11 is not an RSA number either. There is only one RSA number in the range $10 \ldots 12$.

Write a program that inputs a range of numbers and then counts how many numbers from that range are RSA numbers. You may assume that the numbers in the range are less than 1000.

Sample Input 1

```
10
12
```

Sample Output 1

```
The number of RSA numbers between 10 and 12 is 1
```

Sample Input 2

```
11
15
```

Sample Output 2

```
The number of RSA numbers between 11 and 15 is 2
```

CCC '08 J2 - Do the Shuffle

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2008 Stage 1, Junior #2

Those tiny music machines that play your digital music are really computers that keep track of and play music files. The *CCC music player* (C^3MP) is currently in development and will be hitting the stores soon! In this problem, you have to simulate a C^3MP.

The C^3MP music player will hold 5 songs in memory, whose titles will always be "A", "B", "C", "D" and "E". The C^3MP also keeps track of a *playlist*, which is an ordering of all the songs. The C^3MP has 4 buttons that the user will press to rearrange the playlist and play the songs.

Initially, the C^3MP playlist is "A, B, C, D, E". The 4 control buttons do the following:

- Button 1: move the first song of the playlist to the end of the playlist.
  For example: "A, B, C, D, E" will change to "B, C, D, E, A".

- Button 2: move the last song of the playlist to the start of the playlist.
  For example, "A, B, C, D, E" will change to "E, A, B, C, D".

- Button 3: swap the first two songs of the playlist.
  For example, "A, B, C, D, E" will change to "B, A, C, D, E".

- Button 4: stop rearranging songs and output the playlist.

You need to write a program to simulate a CCC music player. Your program should repeatedly ask for two positive integers b and n. Here b represents the button number that the user wants to press, $1 \leq b \leq 4$, and n represents the number of times that the user wants to press button b. You can assume that n always satisfies $1 \leq n \leq 10$.

The input will always finish with the pair of inputs ($b = 4$, $n = 1$) when this happens, you should print the order of songs in the current playlist and your program should end. You can assume that the user will only ever press button 4 once.

Sample Input

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

Output for Sample Input

```
B C D A E
```

Explanation

1. (initial playlist is "A, B, C, D, E")
2. ($b = 2$, $n = 1$ so "A, B, C, D, E" changed to "E, A, B, C, D")
3. ($b = 3$, $n = 1$, so "E, A, B, C, D" changed to "A, E, B, C, D")
4. ($b = 2$, $n = 3$, so "A, E, B, C, D" changed to "B, C, D, A, E")
5. ($b = 4$, $n = 1$) When this happens, you should output the playlist.

CCC '05 J1 - The Cell Sell

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2005 Stage 1, Junior #1

Moe Bull has a cell phone and after a month of use is trying to decide which price plan is the best for his usage pattern. He has two options, each plan has different costs for daytime minutes, evening minutes and weekend minutes.

Plan	Costs		
	daytime	evening	weekend
A	100 free minutes then 25 cents per minute	15 cents per minute	20 cents per minute
B	250 free minutes then 45 cents per minute	35 cents per minute	25 cents per minute

Write a program that will input the number of each type of minutes and output the cheapest plan for this usage pattern, using the format shown below. The input will be in the order of daytime minutes, evening minutes and weekend minutes. In the case that the two plans are the same price, output both plans.

Sample Input 1

```
251
10
60
```

Sample Output 1

```
Plan A costs 51.25
Plan B costs 18.95
Plan B is cheapest.
```

Sample Input 2

```
162
61
66
```

Sample Output 2

```
Plan A costs 37.85
Plan B costs 37.85
Plan A and B are the same price.
```

CCC '03 J2 - Picture Perfect

Time limit: 2.0s **Memory limit:** 256M

Canadian Computing Competition: 2003 Stage 1, Junior #2

Roy has a stack of student yearbook photos. He wants to lay the pictures on a flat surface edge-to-edge to form a filled rectangle with minimum perimeter. All photos must be fully visible. Each picture is a square with dimensions 1 unit by 1 unit.

For example, he would place 12 photos in the following configuration, where each photo is indicated with an `X`.

```
XXXX
XXXX
XXXX
```

Of course, he could orient them in the other direction, such as

```
XXX
XXX
XXX
XXX
```

which would have the same perimeter, 14 units.

Your program should repeatedly read a positive integer C, the number of pictures to be laid out. For each input, it should print the smallest possible perimeter for a filled rectangle that is formed by laying all the pictures edge-to-edge. Also print the dimensions of this rectangle.

You may assume that there are less than $65\,000$ photos. An input value of $C = 0$ indicates that the program should terminate.

Sample Input

```
100
15
195
0
```

Sample Output

```
Minimum perimeter is 40 with dimensions 10 x 10
Minimum perimeter is 16 with dimensions 3 x 5
Minimum perimeter is 56 with dimensions 13 x 15
```