Problem J1: The Cell Sell

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 Session 1

```
Program Output:  Number of daytime minutes?
User Input:      251
Program Output:  Number of evening minutes?
User Input:      10
Program Output:  Number of weekend minutes?
User Input:      60
Program Output:  Plan A costs 51.25
                 Plan B costs 18.95
                 Plan B is cheapest.
```

Sample Session 2

```
Program Output:  Number of daytime minutes?
User Input:      162
Program Output:  Number of evening minutes?
User Input:      61
Program Output:  Number of weekend minutes?
User Input:      66
Program Output:  Plan A costs 37.85
                 Plan B costs 37.85
                 Plan A and B are the same price.
```

Note: If you wish, you may output the prices in cents instead of dollars. For example, for Session 1, Plan A would cost 5125 cents and Plan B would cost 1895 cents.

Problem J2: RSA Numbers

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...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 Session 1

Program Output: `Enter lower limit of range`
User Input: `10`
Program Output: `Enter upper limit of range`
User Input: `12`
Program Output: `The number of RSA numbers between 10 and 12 is 1`

Sample Session 2

Program Output: `Enter lower limit of range`
User Input: `11`
Program Output: `Enter upper limit of range`
User Input: `15`
Program Output: `The number of RSA numbers between 11 and 15 is 2`

Problem J3: Returning Home

Jane's family has just moved to a new city and today is her first day of school. She has a list of instructions for walking from her home to the school. Each instruction describes a turn she must make. For example, the list

```
R
QUEEN
R
FOURTH
R
SCHOOL
```

means that she must turn right onto Queen Street, then turn right onto Fourth Street, then finally turn right into the school. Your task is to write a computer program which will create instructions for walking in the opposite direction: from her school to her home. The input and output for your program will be formatted like the samples below. You may assume that Jane's list contains at least two but at most five instructions, and you may assume that each line contains at most 10 characters, all of them capital letters. The last instruction will always be a turn into the "SCHOOL".

Sample Input 1

```
R
QUEEN
R
FOURTH
R
SCHOOL
```

Sample Output for Sample Input 1

```
Turn LEFT onto FOURTH street.
Turn LEFT onto QUEEN street.
Turn LEFT into your HOME.
```

Sample Input 2

```
L
MAIN
R
SCHOOL
```

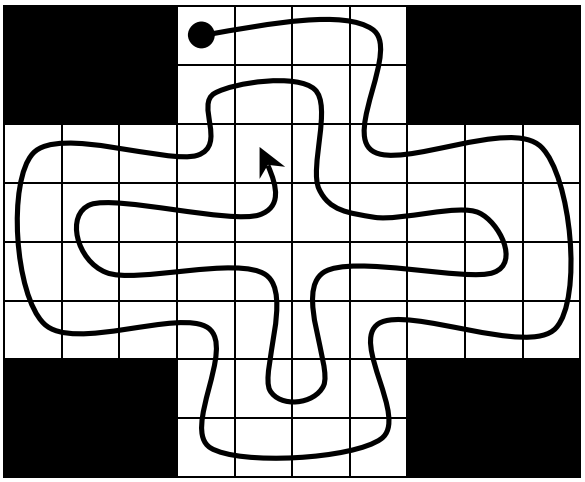
Sample Output for Sample Input 2

```
Turn LEFT onto MAIN street.
Turn RIGHT into your HOME.
```

Problem J4: Cross Spiral

In an old house there is a room that is shaped like a cross. You can think of a cross as being an outlining rectangle with four smaller rectangles cut out of the corners. The floor of the room is completely covered with square tiles. Bridget is walking around the room, stepping from one tile to the next, spiraling towards the centre. Bridget always walks clockwise and stays as close to the edge of the room as possible without stepping on any tile twice. It is possible that she may be trapped and unable to move to an adjacent tile before reaching all the tiles in the room.

Assume that the upper left corner of the outlining rectangle is position (1, 1), that is column 1 and row 1. The walk always starts at column *X* and row 1, where *X* is the leftmost column of row 1 in the cross. The tile at position (*X*, 1) cannot be revisited during the walk, however the first step is counted when Bridget moves to an adjacent tile.



Write a program that calculates Bridget's final column and row position in the room after the walk. The program must accept input for the dimensions of the cross: i.e. the width and height of the outlining rectangle, the width and height of the "cut out" rectangles, and the number of steps the person will take. In the diagram, the inputs for the dimensions of the cross are:
10 8 3 2
In other words, the outlining rectangle is 10 units wide and 8 units high. The "cut out" rectangles are 3 units wide and 2 units high.

The maximum width and height of the outlining rectangle will be 20 x 20. The minimum width of the cross will be 1. The vertical and horizontal parts of the cross are not necessarily the same width.

Sample Input	Sample Output	Description of test case
10 8 3 2 15	7 7	Sample room above, taking 15 steps, and ending up in column 7 and row 7
8 7 2 2 27	5 2	8 columns by 7 rows, with 2-by-2 squares removed in corner. After 27 steps, we end up at column 5, row 2.

8 7 2 2 40	7 4	Same square as above, but after 40 steps, we are at column 7, row 4.
6 6 1 2 1	3 1	A 6 by 6 square, with the corners being pieces 1 column wide and 2 rows high. After one step, we end at column 3, row 1.

Problem J5: Bananas

The term “code monkey” is sometimes used to refer to a programmer who doesn't know much about programming. This is unfair to monkeys, because contrary to popular belief, monkeys are quite smart. They have just been misunderstood. This may be because monkeys do not speak English, but only monkey language. Your job is to help humans and monkeys understand each other by writing a monkey language dictionary. For each word that is typed in, your program must determine whether it is a valid monkey language word.

Unlike in English, spelling in monkey language is very simple. Every word in monkey language satisfies the following rules, and every word satisfying the following rules is a monkey language word.

1. A monkey language word is a special type of word called an *A-word*, which may be optionally followed by the letter N and a monkey language word.
2. An *A-word* is either only the single letter A, or the letter B followed by a monkey language word followed by the letter S.

Here are some examples:

The word “A”' is a monkey language word because it is an A-word.

The word “ANA” is a monkey language word because it is the A-word “A” followed by the letter N and the monkey language word “A”.

The word “ANANA” is a monkey language word because it is the A-word “A” followed by the letter N and the monkey language word “ANA”.

The word “BANANAS” is a monkey language word because it is an A-word, since it is the letter B followed by the monkey language word “ANANA” followed by the letter S.

Write a program which accepts words, one word on each line, and for each word prints YES if the word is a monkey language word, and NO if the word is not a monkey language word. The input word “X” indicates the program should terminate, and there is no output for word “X” (even though it is not a monkey word).

Sample Input

```
A
ANA
ANANA
BANANAS
BANANA
X
```

Sample Output for Sample Input

```
YES
YES
YES
YES
NO
```