Function Name: `lovePropagation`

Inputs:

1. (*double*) A 1 x 2 vector of initial values.
2. (*double*) A maximum value.

Outputs:

1. (*double*) A vector containing numbers less than or equal to the maximum value.

Topics: (*while loops*), (*iterative concatenation*)

Background:

It's Valentine's week! Love can only grow, but your psychology professor says it can only grow so far. You wish to simulate the growth of love based on your professor's theory and have decided to use MATLAB for the purpose.

Function Description:

Write a function that takes in a 1 x 2 vector of initial values, and outputs a vector where each value is the sum of the preceding two values. The first two values are given by the input. This means that the 3rd entry should be the sum of the 1st and the 2nd entries, 5th entry should be the sum of the 3rd and 4th entries, and so on. Stop concatenating values when the value you concatenate is larger than the second input. The output vector must NOT contain a value that is greater than the second input to your function.

Example:

```
init = [2, 6]
limit = 94
prop = lovePropagation(init, limit)

prop → [2 6 8 14 22 36 58 94]
```

Notes:

- The input vector will only contain positive values.

Function Name: `mostAdmired`

Inputs:
1. (*double*) A 1XN vector of values where each value only appears once
2. (*double*) A 1XM vector of the same values where each value may appear multiple times

Outputs:
1. (*char*) A sentence describing who has the most secret admirers

Topics: (*for loops*), (*masking*)

Background:

It's finally February 14th, and everyone in your CS 1371 recitation is giving and (hopefully) receiving Valentine's Day cards. For ultimate secrecy, all of the cards are labeled with students' GTID numbers rather than their names. Being the nosey student you are, you decide to snoop around and figure out who received the most letters.

Function Description:

You are given a vector of values where each value is unique. Each value represents a different GTID, or a different student. Additionally, you are given a longer vector that contains the values from the first vector. However, in the second vector, the values may appear any number of times. Each of these values in the second vector represents a letter received by the student with the matching value from the first vector.

Iterate through the first vector, each time searching the second vector for the corresponding value. Count the number of times that the value appears in the second vector. Keep track of the value that appears the most in the second vector as well as how many times it appears. Finally, output a string with the format below with the final numbers.

```
'Student <value> has <number of occurrences> secret admirers!'
```

Example:

```
students = [ 3 2 7 5 9 ]
letters = [ 7 9 3 3 7 2 9 7 3 7 ]
sentence = mostAdmired ( students, cards )
sentence → 'Student 7 has 4 secret admirers!'
```

Notes:
- There will never be a tie.
- The greatest number of letters received will always be more than 1.

Hints:
- There are two ways to count the number of times that a value appears in a vector. One is to sum the vector (the mask); the other is to use the mask to index out the corresponding values and then take the length of that new vector.

Function Name: `flowerRun`

Inputs:

1. (*double*) A 1xN vector of house numbers
2. (*double*) A 1xM vector of grocery stores

Outputs:

1. (*double*) A 1xN vector of grocery stores

Topics: (*for loops*), (*array math/functions*)

Background:

You are having a poker night with your friends who live on your street when someone mentions that tomorrow is Valentine's Day. You, having a significant other and no gift picked out, begin to panic. You and your friends decide to use MATLAB to find which grocery store is closest to each of your houses so you can run to buy flowers and chocolates first thing in the morning.

Function Description:

Given two vectors of addresses, one which represents the location of houses and another which represents the location of stores. Return a vector of the same size as the houses vector wherein each element is replaced with the address of the store closest to it. The distance between two addresses is simply the absolute value of the difference between them.

Example:

```
houses = [10   1   6  15   8]
stores = [ 5  20   2]
[closest] = flowerRun(houses, stores)
closest → [5   2   5  20   5]
```

Notes:

- The stores and houses are in no particular order and are NOT guaranteed to be the same length.
- There will never be two stores equidistant from a house.

Function Name: `loveLetter`

Inputs:

1. (*char*) The encrypted message
2. (*char*) A vector of letter-number pairs

Outputs:

1. (*char*) The decrypted message

Topics: (*for loops*), (*non-unit step vector creation*)

Background:

Valentine's Day is here and as you leave for class you find a note taped outside your door. It appears to be a message, but some of the letters have been replaced with numbers, making it nearly impossible to read. Luckily, the back of the note contains a string of letters and numbers that you think may contain the key to decoding the message.

Function Description:

You will be given an encoded message where some of the letters are all replaced with a certain numerical characters (e.g. `'0'`, `'1'`, `'2'`...). For example, all the a's might be replaced with `'2'`, all of the `r`'s might be replaced with '6', etc. You will also be given another string that contains the missing letters, followed by their corresponding digit they were replaced with, (e.g. `'a2r6'`). Using the letter-number pairs, replace the numbers in the message with their corresponding letter to complete the message.

Example:

```
encrypted = 'Thi2 i2 6n 8x6mpl8 m8226g8.';
pairs = 's2a6e8';
decrypted = loveLetter(encrypted, pairs);
decrypted → 'This is an example message.';
```

Notes:

- The numbers will always be a single digit, e.g. a letter will never be replaced by `'10'`.

Function Name: `badBreakup`

Inputs:

1. (*char*) A vector of items and their value

Outputs:

1. (*char*) The most valuable item

Topics: (*while loops*), (*string tokenization*)

Background:

You met on Tech Green, and it was the start of a beautiful relationship…..until you found out that they are currently in 4 other relationships. *The nerve!!* After the break up you realize that you gave them some presents and decide to take the most valuable item back and liquidate it (because that's exactly what they did to your heart).

Function Description:

Given a comma separated list of an item followed by a number indicating its value (e.g. `'ferrari,200000,wallet,80,snack,10'`) identify the item that has the highest value and return its name as a character vector.

Example:

```
list = 'airpods,160,mayonnaise,4,yeezys,300,snuggie,15';
mostVal = badBreakup(list);
mostVal → 'yeezys'
```

Notes:

- The list can have any number of items.
- There will never be a negative value of an item.
- There will never be ties for the maximum value.