# String Encoder Software

The text “The quick and brown fox jumps over the Lazy Dog” contains all 26 alphabets in English.

We can generate a code for any letter in the following manner:

The code is a combination of two digits. The first digit represents the index of the word (first occurrence), which contains the letter and the second digit represents the index of that letter in the found word.

For example, the letter “m” is found in the word “jumps” which is at index 5 in the sentence, and in that word, “m” is found at index 2. Hence the code is 52. Similarly, the letter “v” is found in the word “over” which is at index 6 in the sentence and in that word, “v” is at index 1. Hence the code for the letter “v” is 61.

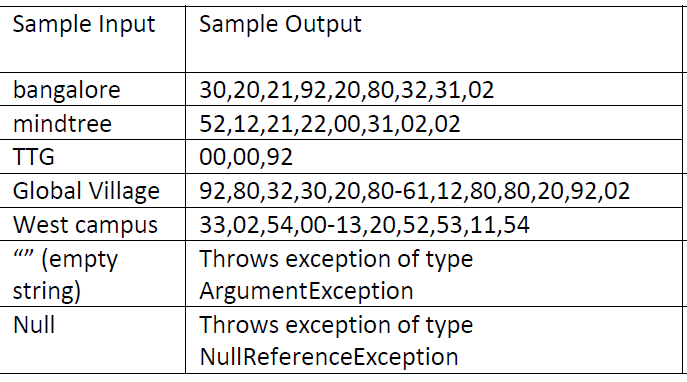
Write a program that accepts a String and returns the encoded String of the input.

**Input**

Input is a case-insensitive String containing one or more words delimited by space.

**Output**

The output is a String containing codes for each letter separated by comma. The space in the input is replaced by a hyphen in the output.



**Assumptions/Hints:**

1. Input contains only alphabets (lowercase or uppercase) and/or spaces (no special characters and digits).

# Unique and Duplicate Words

Create a function that takes a string as an argument. The function should separate the duplicate words and unique words of the string and return. The function should have the following rules:

* If a string is passed as an argument, it should return the unique and duplicate values as 2 Lists.
* If an empty string is passed as an argument, it should throw a Custom exception called EmptyStringArgument.
* If a null is passed as an argument, it must throw NullReferenceException.

The Program should handle the exceptions and properly display the results.

|  |  |
| --- | --- |
| Sample Input | Expected Output |
| "Do not call me, let me call you" | Unique: List<string> { "do", "let", "not", "you" } Duplicate: List<string> { "call", "me" } |
| "I had coffee at a coffee shop yesterday during coffee break" | Unique: List<string> { "a", "at", "break", "during", "had", "I", "shop", "yesterday" }  duplicate : List<string> { "coffee" } |
| “” | ExpectedException(typeof(EmptyStringException) |
| Null | ExpectedException(typeof(NullReferenceException) |