Question - 1
Last and Second-Last

Given a string, create a new string made up of its last two letters, reversed and separated by a space.

Example
Given the word '*bat*', return '*t a*'.

Function Description
Complete the function *lastLetters* in the editor below.

lastLetters has the following parameter(s):
string word: a string to process

Returns:
string: a string of two space-separated characters

Constraint

- *2 ≤* length of *word ≤ 100*

▼ Input Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The line contains a string, *word*.

▼ Sample Case 0

Sample Input

```
STDIN     Function
-----     -----
APPLE  →  word = 'APPLE'
```

Sample Output

```
E L
```

Explanation
The last letter in '*APPLE*' is *E* and the second-to-last letter is *L*, so return *E L*.

Question - 2
FizzBuzz

Given a number *n*, for each integer *i* in the range from *1* to *n* inclusive, print one value per line as follows:

- If *i* is a multiple of both *3* and *5*, print *FizzBuzz*.
- If *i* is a multiple of *3* (but not *5*), print *Fizz*.
- If *i* is a multiple of *5* (but not *3*), print *Buzz*.
- If *i* is not a multiple of *3* or *5*, print the value of *i*.

Function Description
Complete the function *fizzBuzz* in the editor below.

fizzBuzz has the following parameter(s):
  int n: upper limit of values to test (inclusive)
Returns: NONE
Prints:
  The function must print the appropriate response for each value *i* in the set *{1, 2, ... n}* in ascending order, each on a separate line.

Constraints

- $0 < n < 2 \times 10^5$

▼ Input Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The single integer *n*, the limit of the range to test: [1, 2, ...n].

▼ Sample Case 0

Sample Input

```
STDIN    Function
-----    --------
15    →  n = 15
```

Sample Output

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
```

Explanation
The numbers *3*, *6*, *9*, and *12* are multiples of *3* (but not *5*), so print *Fizz* on those lines.
The numbers *5* and *10* are multiples of *5* (but not *3*), so print *Buzz* on those lines.
The number *15* is a multiple of both *3* and *5*, so print *FizzBuzz* on that line.
None of the other values is a multiple of either *3* or *5*, so print the value of *i* on those lines.

Question - 3
Count Duplicate Elements

Given an integer array, *numbers,* count the number of elements that occur more than once.

Example
numbers = [1, 3, 3, 4, 4, 4]

There are two non-unique elements: *3* and *4.*

Function Description
Complete the function *countDuplicate* in the editor below.

countDuplicate has the following parameter(s):
int numbers[n]: an array of integers

Returns:
int: an integer that denotes the number of non-unique values in the *numbers* array

Constraints

- $3 \le n \le 1000$
- $1 \le numbers[i] \le 1000, 0 \le i < n$

▼ Input Format Format for Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer *n*, the size of the *numbers* array.
Each of the next *n* lines contains an integer, *numbers[i],* where $0 \le i < n$.

▼ Sample Case 0

Sample Input

```
STDIN     Function
-----     -----
8      →  numbers[] size n = 8
1      →  numbers = [1, 3, 1, 4, 5, 6, 3, 2]
3
1
4
5
6
3
2
```

Sample Output

```
2
```

Explanation
The values *1* and *3* occur more than once, therefore the answer is *2.*

▼ Sample Case 1

Sample Input

```
STDIN     Function
-----     -----
6      →  numbers[] size n = 6
1      →  numbers = [1, 1, 2, 2, 2, 3]
1
2
2
2
3
```

Sample Output

```
2
```

Explanation

The values *1* and *2* occur more than once, therefore the answer is *2*.