# Lab: Dictionaries, Lambda and LINQ

You can check your solutions in [Judge](https://alpha.judge.softuni.org/contests/associative-arrays-lab/1212)

# Associative Arrays

## Count Real Numbers

Read a **list of integers** and **print them in ascending order,** along with their **number of occurrences**.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 8 2 2 8 2 | 2 -> 3  8 -> 2 | 1 5 1 3 | 1 -> 2  3 -> 1  5 -> 1 | -2 0 0 2 | -2 -> 1  0 -> 2  2 -> 1 |

### Hints

Read an array of doubles:



Use SortedDictionary<double, int> named counts.



Pass through each of the numbers and increase their count - counts[num], if num exists in the dictionary, or assign counts[num] = 1, if the number does not exist in the dictionary. We are assigning it that value, because it is its first occurrence. The count represents the occurrences.



Pass through all of the numbers in the dictionary and print the number num and its count of occurrences.



## Odd Occurrences

Create a program that extracts all elements from a given sequence of words that are present in it an **odd number of times** (case-insensitive).

* Words are given on a single line, space-separated.
* Print the result elements in lowercase, in their order of appearance.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Java C# PHP PHP JAVA C java | java c# c |
| 3 5 5 hi pi HO Hi 5 ho 3 hi pi | 5 hi |
| a a A SQL xx a xx a A a XX c | a sql xx c |

### Hints

Read a line from the console and split it by a space.



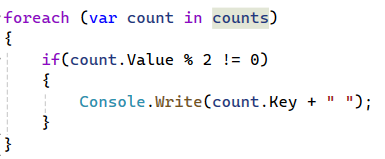
Use a **dictionary** (string 🡪 int) to count the occurrences of each word.



Pass through each of the elements in the array and count each word.



Pass through the dictionary and print words that occur at odd times.



## Word Synonyms

Create a program, which keeps a dictionary with synonyms. The **key** of the dictionary will be the **word**. The **value** will be a **list of all the synonyms of that word**. You will be given a number **n – the count of the words**. After each word, you will be given a synonym, so the count of lines you have to read from the console is **2 \* n. You will be receiving** a **word** and a **synonym** each on a separate line like this:

* {**word**}
* {**synonym**}

If you get the same word twice, just add the new synonym to the list.

Print the words in the following format:

**"{word} - {synonym1, synonym2, …, synonymN}"**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  cute  adorable  cute  charming  smart  clever | cute - adorable, charming  smart - clever |
| 2  task  problem  task  assignment | task – problem, assignment |

### Hints

* Use a **Dictionary (string** 🡪 **List<string>)** to keep all of the synonyms.

****

* **Read n \* 2 lines**
* **Add the word in the dictionary, if it is not present**



* **Add the synonym as a value to the given word**

****

* **Print each word with the synonyms in the required format**

# LINQ

## Word Filter

Read an array of strings and take only words, whose length is an even number. Print each word on a new line.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| kiwi orange banana apple | kiwi  orange  banana |
| pizza cake pasta chips | cake |

* Read an array of strings
* Filter those, whose length is even



* Print each word on a new line