# Exercises: Unit Testing Dictionaries

Test your tasks in the Judge system:

<https://alpha.judge.softuni.org/contests/dictionaries-lambda-and-linq-unit-testing-exercise/4474>

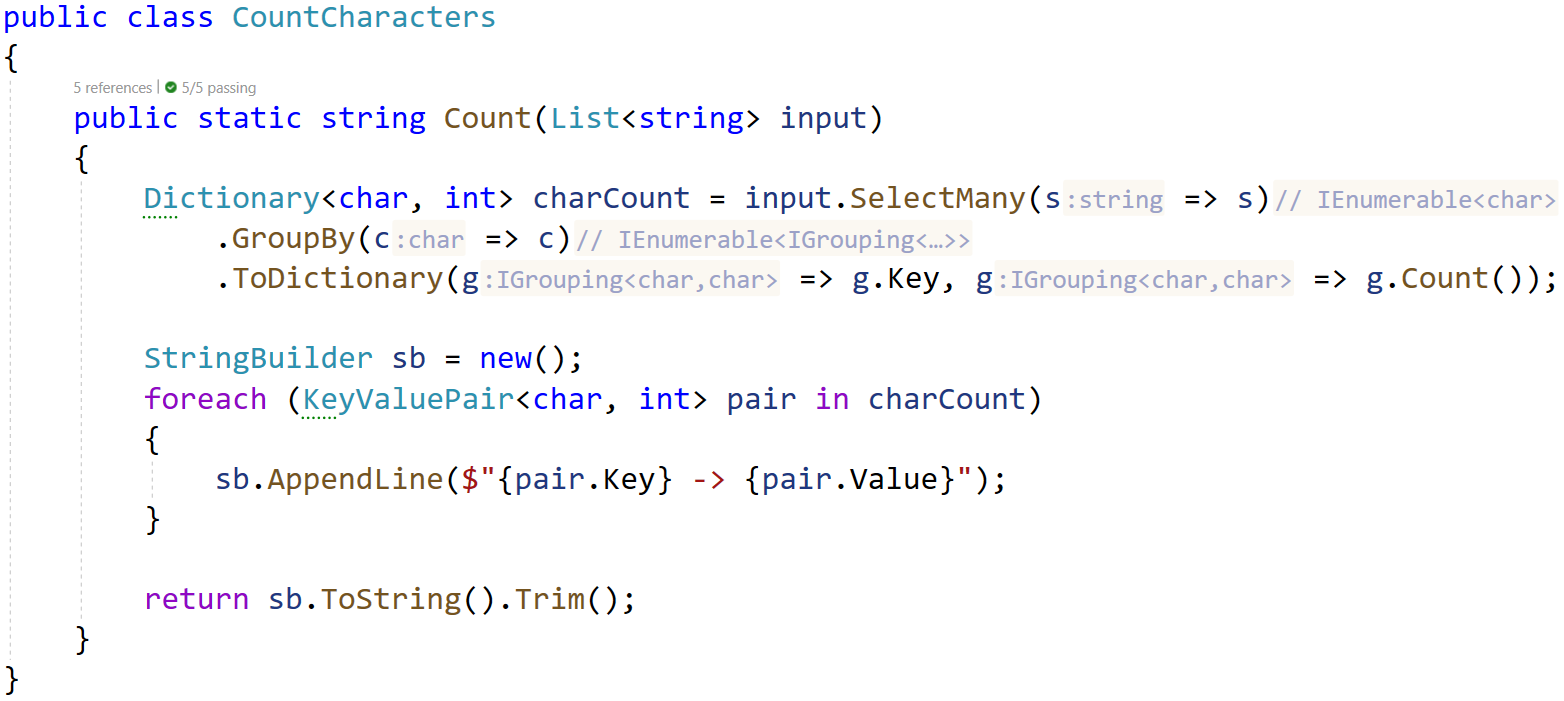
# Unit Test: Count Characters

Look at the **provided skeleton** and examine the CountCharacters.cs class that you will test:

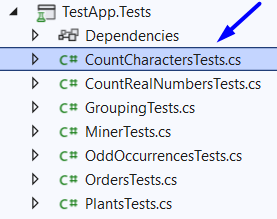
A screenshot of a computer

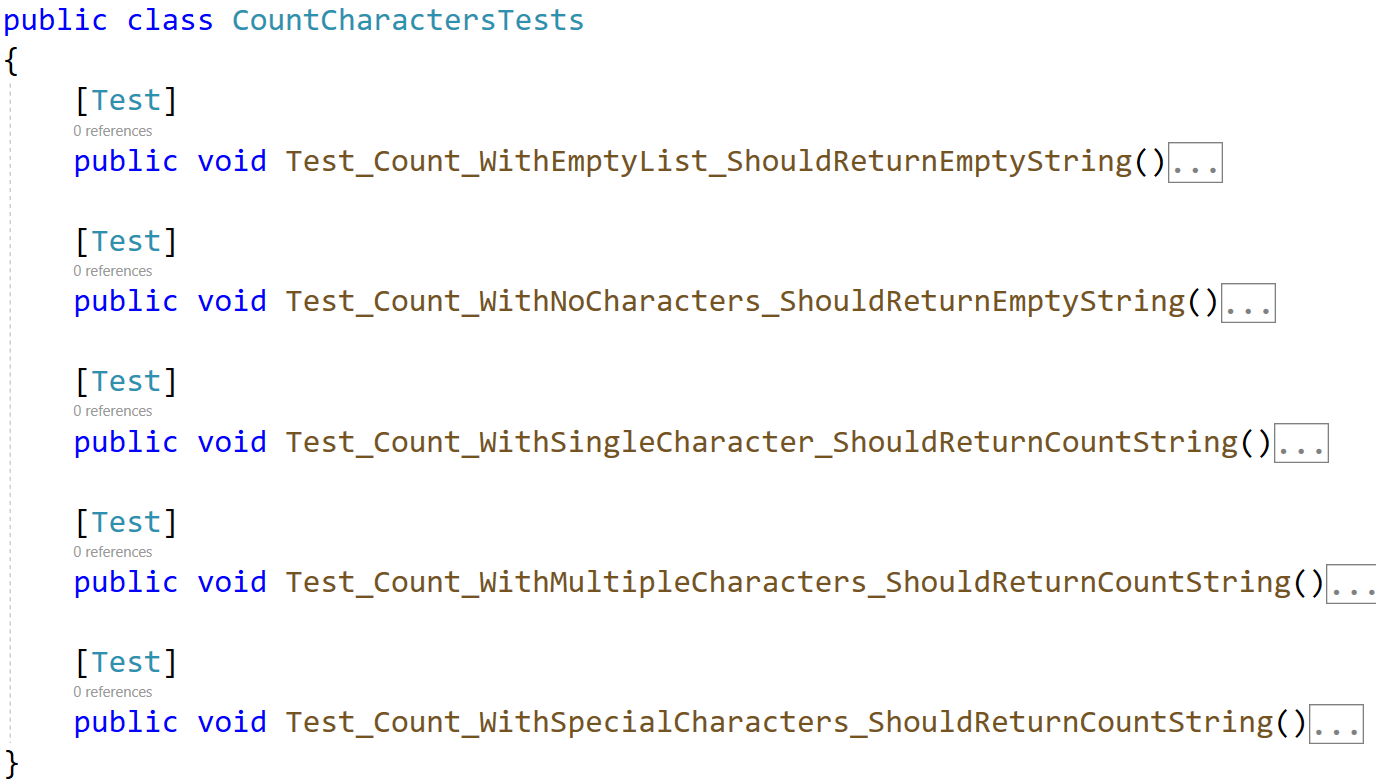
Description automatically generated

The method takes in a **list of strings**, and collects the **number of times a character has appeared** and returns a string representing that information:



Then, look at the tests inside the CountCharactersTests.cs class:





The first test if **finished** so you have a **reference**, **one** is finished **partially**, the rest of the tests are **empty,** and your task is to finish them. The tests should run when you're finished:

A screenshot of a computer

Description automatically generated

# Unit Test: Count Real Numbers

Test a given method which takes in an **array of integers** and **counts** how many **times** each **number** wasseen.

The method is found in the CountRealNumbers.cs file:

A screenshot of a computer code

Description automatically generated

You are given a **test** **file** CountRealNumbresTests.cs which contains **5 tests**. **One** of them has been **finished partially**, and **four** are **empty** for you to finish:

A screenshot of a computer program

Description automatically generated

When you are ready make sure your **tests run:**

A screenshot of a computer program

Description automatically generated

# Unit Test: Grouping

Test a given method which takes in a **list of integers** and **groups** them by **even** and **odd** numbers.

The method is found in the Grouping.cs file:

A computer code with many text

Description automatically generated with medium confidence

You are given a **test** **file** GroupingTests.cs which contains **5 tests**. **One** of them has been **finished partially**, and **four** are **empty** for you to finish:

A screenshot of a computer

Description automatically generated

When you are ready make sure your **tests run:**

A screenshot of a computer

Description automatically generated

# Unit Test: Odd Occurrences

Test a given method which takes in an **array of strings** andfindswhich **words appear** an **odd number** oftimes.

The method is found in the OddOccurrences.cs file:

A computer code with text

Description automatically generated with medium confidence

You are given a **test** **file** OddOccurencesTests.cs which contains **5 tests**. **One** of them has been **finished partially**, and **four** are **empty** for you to finish:

A screenshot of a computer

Description automatically generated

When you are ready make sure your **tests run:**

A screenshot of a computer

Description automatically generated

# Unit Test: Miner

Test a given method which takes in **N number of strings** in the form of:

"{mineral} {quantity}"

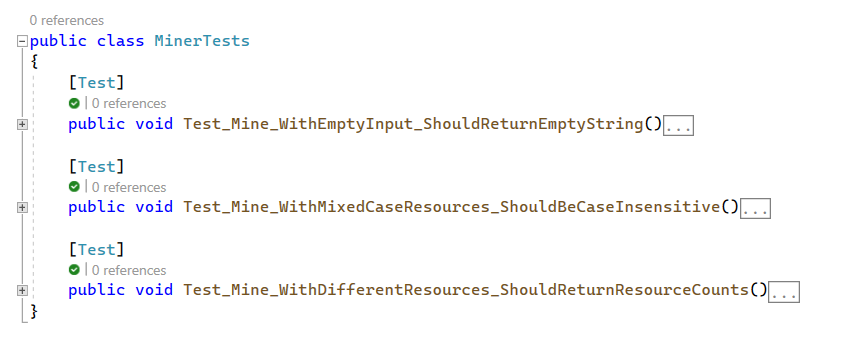
Then it counts the **total quantity of a given mineral** and returns a string showing that.

The method is found in the Miner.cs file:

A computer code with text

Description automatically generated with medium confidence

You are given a **test** **file** MinerTests.cs which contains **4 tests**. **One** of them has been **finished partially**, and **three** are **empty** for you to finish:



When you are ready make sure your **tests run:**

A screenshot of a computer

Description automatically generated

# Unit Test: Orders

Test a given method which takes in **N number of strings** in the form of:

"{product} {price} {quantity}"

It saves **each** **product** and **quantity** and **updates** the **price** each time it **changes**, and finally **calculates** the **total price** for each **product**.

The method is found in the Orders.cs file:

A screenshot of a computer code

Description automatically generated

A computer code with text

Description automatically generated

You are given a **test** **file** OrdersTests.cs which contains **4 tests**. **One** of them has been **finished partially**, and **three** are **empty** for you to finish:

A screenshot of a computer program

Description automatically generated

When you are ready make sure your **tests run:**

A screenshot of a computer

Description automatically generated

# Unit Test: Plants

Test a given method which takes in an **array of strings** which saves and groups plants based on their number of letters, the shortest named plants will grow the **fastest**.

The method is found in the Plants.cs file:

A screen shot of a computer code

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

You are given a **test** **file** PlantsTests.cs which contains **4 tests**. **One** of them has been **finished partially**, and **three** are **empty** for you to finish:

A screenshot of a computer screen

Description automatically generated

When you are ready make sure your **tests run:**

A screenshot of a computer program

Description automatically generated

At the end make sure all tests pass:

A screenshot of a computer

Description automatically generated