# Lab: Dictionaries

Problems for in-class lab for the [Python Fundamentals Course @SoftUni](https://softuni.bg/trainings/3368/python-fundamentals-may-2021).

Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1736>.

## Bakery

*This is your first task at your new job. You were tasked to create a list of the stock in a bakery, and you really don't want to fail at you first day at work.*

You will receive a single line containing some **food** (keys) and **quantities** (values). They will be separated by a **single space** (the **first** element is the **key**, the **second** – the **value** and so on). Create a **dictionary** with all the keys and values and **print** it on the console

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| bread 10 butter 4 sugar 9 jam 12 | {'bread': 10, 'butter': 4, 'sugar': 9, 'jam': 12} |

### Hint

Let us start with **reading the input** and **creating an empty dictionary**:



* Note that there are **2 ways** of creating a dictionary (with the **curly braces** or using the **dict()** method).

Since we know that we will get **key-value pairs**, we can use a **for-loop** with **step 2.** We take the **key** and the **value** and **add** them to the dictionary and after the **loop** **is over,** we **print** it:



* Note that the **value** must be an **integer** (since it is a **quantity**).

RS solution:

ingredients = input().split()  
keys = []  
values = []  
dictionary\_ingredients = {}  
  
for i in range(len(ingredients)):  
 if i % 2 == 0:  
 keys.append(ingredients[i])  
 else:  
 values.append(int(ingredients[i]))  
  
dictionary\_ingredients = dict(zip(keys, values))  
print(dictionary\_ingredients)

## Stock

*After you have successfully completed your first task, your boss decides to give you another one right away. Now not only you have to keep track of the stock, but also you should answer customers if you have some products in stock or not.*

You will be given **key-value** pairs of **products** and **quantities** (on a single line **separated by space**). On the next line you will be given products to **search** for. Check for each product, you have **2 possibilities**:

* If you **have the product**, print **"We have {quantity} of {product} left"**
* **Otherwise**, print **"Sorry, we don't have {product}"**

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| cheese 10 bread 5 ham 10 chocolate 3  jam cheese ham tomatoes | Sorry, we don't have jam  We have 10 of cheese left  We have 10 of ham left  Sorry, we don't have tomatoes |

### Hint

We repeat the steps from the previous task for reading the products and adding them to the dictionary



Next, we **read** the products we have to **search** for and **check** for each of them



stock\_items = input().split()  
warehouse = {}  
  
for i in range(0, len(stock\_items), 2):  
 key = stock\_items[i]  
 value = stock\_items[i+1]  
 warehouse[key] = int(value)  
  
searched\_products = input().split()  
  
for x in searched\_products:  
 if x in warehouse:  
 print(f"We have {warehouse[x]} of {x} left")  
 else:  
 print(f"Sorry, we don't have {x}")

## Statistics

*You seem to be doing great at your first job. You have now successfully completed the first 2 of your tasks and your boss decides to give you as your next task something more challenging. You have to accept all the new products coming in the bakery and finally gather some statistics.*

You will be receiving **key-value** pairs on separate lines separated by **": "** until you receive the command **"statistics"**. Sometimes you may receive a product **more than once**. In that case you have to **add** the **new quantity** to the existing one. When you receive the **"statistics"** command, print the following:

**"Products in stock:**

**- {product1}: {quantity1}**

**- {product2}: {quantity2}**

**…**

**Total Products: {count\_all\_products}**

**Total Quantity: {sum\_all\_quantities}"**

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| bread: 4  cheese: 2  ham: 1  bread: 1  statistics | Products in stock:  - bread: 5  - cheese: 2  - ham: 1  Total Products: 3  Total Quantity: 8 |

### Hint

Let us start by **creating** **the dictionary** and creating a **while loop**



Now, let us get the **product** and the **quantity**



Then, we want to create a **check if** the product is **not already** in the dictionary and set its **value** to **0**



* That way we make sure that the **product will exist** before we add the **quantity**

Then we **add up the quantity**



And finally, we **print** the result



* For the total products we get the **length of the keys**
* For the total quantity we **sum the values**

**\*** Another way of implementing lines 14 and 15 would be by using **dictionary comprehension**, which will be covered in the advanced course



data = input()  
  
products = {}  
  
while not data == "statistics":  
 product\_name, product\_quantity = data.split(": ")  
 if product\_name not in products:  
 products[product\_name] = int(product\_quantity)  
 else:  
 products[product\_name] += int(product\_quantity)  
 data = input()  
  
print("Products in stock:")  
  
for key, value in products.items():  
 print(f"- {key}: {value}")  
  
print(f"Total Products: {len(products)}")  
print(f"Total Quantity: {sum(products.values())}")

## Odd Occurrences

Write 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**, and create a **dictionary**



Create a **loop** and check for **each word** (**lower case**) if it is **in the dictionary**, and if it is not, add it



Then create another loop using the **items()** method and check if the **occurrence** of the current word is **odd**



words\_sequence = input().split()  
dictionary = {}  
  
for x in words\_sequence:  
 word\_lower = x.lower()  
 if word\_lower not in dictionary:  
 dictionary[word\_lower] = 0  
 dictionary[word\_lower] += 1  
  
for (key, value) in dictionary.items():  
 if value % 2 != 0:  
 print(key, end=" ")

## Word Synonyms

Write a program, which keeps a dictionary with synonyms. The **key** to 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 should 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 |

### Hint

We start by reading the number **n** and creating the **dictionary with synonyms**



Then we create a **for loop** to read the **word-synonym pairs**



We check if the **word is not** in the dictionary and in that case we set its **value to an empty list** (since one word can have **multiple synonyms**) and we **append the new synonym** to that list



Finally, we **print** the result



n = int(input())  
  
synonym\_dict = {}  
  
for \_ in range(n):  
 word = input()  
 synonym = input()  
  
 if word not in synonym\_dict:  
 synonym\_dict[word] = []  
 synonym\_dict[word].append(synonym)  
  
for key, value in synonym\_dict.items():  
 print(f"{key} - {', '.join(value)}")

## Students

You will be receiving names of students, their ID and a course of programming they have taken in the format **"{name}:{ID}:{course}"**. On the last line you will receive a name of a course in snake case lowercase letters. You should print only the information of the students taken the corresponding course in the format: **"{name} - {ID}"** on separate lines.

***Note: each student's ID will always be unique***

|  |  |
| --- | --- |
| **Input** | **Output** |
| Peter:123:programming basics  John:5622:fundamentals  Maya:89:fundamentals  Lilly:633:fundamentals  fundamentals | John - 5622  Maya - 89  Lilly - 633 |
| Alex:6:programming basics  Maria:7:programming basics  Kaloyan:9:advanced  Todor:10:fundamentals  programming\_basics | Alex - 6  Maria - 7 |

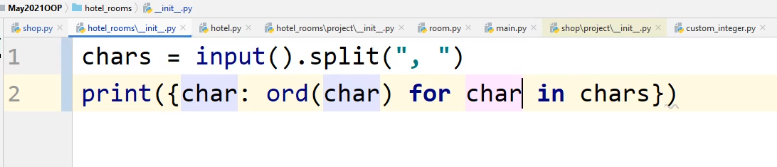
data = input()  
courses = {}  
  
while ":" in data:  
 # student\_name, id, course\_name = data.split(":")  
 # if course\_name not in courses:  
 # courses[course\_name] = {}  
 # courses[course\_name][id] = student\_name  
 # data = input()  
  
 student\_name, id, course\_name = data.split(":")  
 if course\_name not in courses:  
 courses[course\_name] = {id: student\_name}  
 else:  
 courses[course\_name].update({id: student\_name})  
  
 data = input()  
  
searched\_course = data  
searched\_course\_name\_as\_list = searched\_course.split("\_")  
searched\_course = ' '.join(searched\_course\_name\_as\_list)  
  
for course\_name in courses:  
 if course\_name == searched\_course:  
 for id, name in courses[course\_name].items():  
 print(f"{name} - {id}")

## ASCII Values

Write program that receives a **list of characters** separated by **", "** and creates a dictionary with each **character** as a **key** and its **ASCII** value as a **value**. Try solving that problem using **comprehensions**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| a, b, c, a | {'a': 97, 'b': 98, 'c': 99} |
| d, c, m, h | {'d': 100, 'c': 99, 'm': 109, 'h': 104} |



chars = input().split(", ")  
char\_to\_ascii\_dictionary = {char: ord(char) for char in chars}  
print(char\_to\_ascii\_dictionary)