# Problem 4 – Population Aggregation

Write a program that receives as input information about **country, city and** its **population** and prints an **aggregated statistic**. There are **2 types of input** lines

* **{Country}**\**{city}**\**{population}**
* **{city}**\**{Country}**\**{population}**

The **country name always starts with a capital letter** and the **city name always starts with a lowercase letter**. The country name and the city name are both polluted with **prohibited symbols (@, #, $, & and digits from 0 to 9)**. Your task is to **clear all prohibited symbols** and **print aggregated data** about the **all the** **countries ordered alphabetically** in format:

**{Country} -> {number of cities}**

And **top 3 cities with biggest population** ordered in **descending** **order** **by population** in format:

**{city} -> {population}**

In case of **repeating towns**, count the last seen population for each town (ignore the others).

**Count all towns** in each country. In case of repeating towns for certain country, **count the duplicated towns**.

### Input

* The input data should be read from the console.
* It consists of a variable number of lines and ends when the command "**stop**" is received.
* The input data will always be valid and in the format described. There is no need to check it explicitly.

### Output

* The output should be printed on the console.
* Print the aggregated data for each country and city in the described format.

### Constraints

* The name of the city, country and the population will be separated from each other by **a back slash ('\')**.
* The **number of input lines** will be in the range [2 … 50].
* The **population count** of each city will be an integer in the range [0 … 263 − 1].
* Allowed working time for your program: 0.1 seconds. Allowed memory: 16 MB.

### Examples

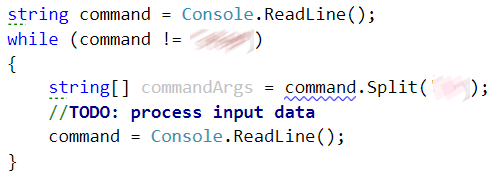
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| Bulgaria\sofia\123000  burgas\Bulgaria\4456576  stop | Bulgaria -> 2  burgas -> 4456576  sofia -> 123000 | Bulgaria\sofia\100  sofia\Bulgaria\200  stop | Bulgaria -> 2  sofia -> 200 |

|  |  |
| --- | --- |
| **Input** | **Output** |
| G$er&m@an@y\berlin\1234333  pa$r###is\F&r&a&n&c&e\30000000  Bulg@aria\varn@a@#$#\32145535  Bulgaria\pom$#or$ie\3131231  l$#ond$32on\U$#434565K43\98686644  ham$#bu4300r43g\Ger$man2@y\1324  stop | Bulgaria -> 2  France -> 1  Germany -> 2  UK -> 1  london -> 98686644  varna -> 32145535  paris -> 30000000 |

# Solution

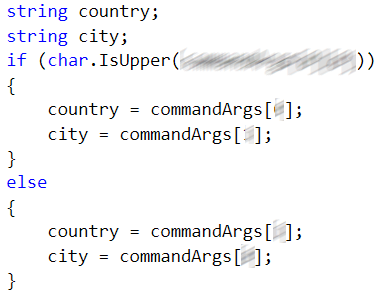
## Read Input

Firstly we need to create a loop that will read commands from the console until it receives command “stop”. Every line is split by back slash.



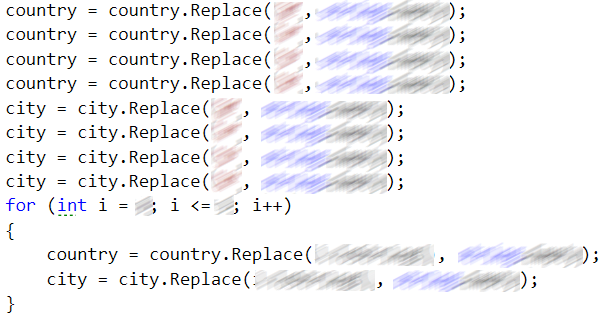
## Get Country And City From Input

Now we should decide which element of the commandArgs array is the country and which the city is. To decide we must check if the first letter of the first argument is in upper case. If it is the first argument is the country and the second is the city. Otherwise the first element is the city and the second element is the country



## Remove Prohibited Symbols

We could easily do that by replacing the prohibited symbols with an empty string.

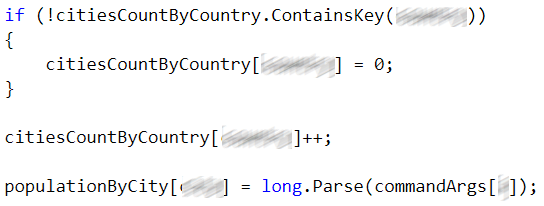


## Add Information to The Appropriate Data Structure

We now need to define 2 data structures that will keep the information we need. First data structure will keep information about countries and cities count ordered alphabetically. The second data structure will keep information about city and its population. We can use **SortedDictionary<string, int>** and **Dictionary<string, long>**

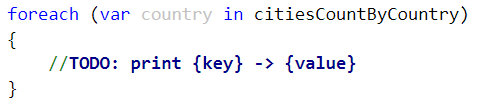
**c4**

Let’s add the data to the appropriate dictionaries. To the first dictionary if the country does not exists in it we add it and set its counter to 1 otherwise we increment the value with 1. And on the second structure we just add the city as key and its population as value.



## Print Countries

Just print all the data that is in the sorted dictionary



## Print Biggest 3 Cities by Population

Now we should sort the dictionary containing cities name by their population in descending order and take first 3 elements and print their information

