# C++ Fundamentals: Second Exam

The following tasks should be submitted to the SoftUni Judge system, which will be open starting Sunday, 28 January 2018, 09:00 (in the morning) and will close the same day at 15:00. Submit your solutions here: https://judge.softuni.bg/Contests/Compete/Index/923.

For this exam, the code for each task should be a single C++ file, the contents of which you copy-paste into the Judge system.

Please be mindful of the strict input and output requirements for each task, as well as any additional requirements on running time, used memory, etc., as the tasks are evaluated automatically and not following the requirements strictly may result in your program's output being evaluated as incorrect, even if the program's logic is mostly correct.

You can use C++03 and C++11 features in your code.

Unless explicitly stated, any integer input fits into **`int`** and any floating-point input can be stored in **`double`**. On the Judge system, a C++ **`int`** is a **`32-bit`** signed integer and a C++ **`double`** is a **`64-bit`** IEEE754 floating point number.

NOTE: the tasks here are NOT ordered by difficulty level.

# Task 3 – Entities (Second-Exam-Task-3-Entities)

Relational Databases store data in "tables", where each row represents an entity, while the columns represent the properties/fields in an entity. All entities in a table are of the same type/schema, i.e. they all have the same fields. Of course, each row usually has different values in each of its fields, but sometimes some rows have the same values in some of the columns (e.g. in a table of people, with columns name, age, and height, two people could have the same age, or same height, even the same name).

One common operation on tables in databases is counting similarities between entities. For example, you could search for the most-common value in a column. Let's say we have the following (*totally fictional and not at all related to any real people or events*) table of exam scores:

| username | task1 | task2 | task3 | task4 | total |
|---|---|---|---|---|---|
| JanVidenov | 100 | 100 | 100 | 100 | 400 |
| MartinFazanov | 100 | 100 | 0 | 100 | 300 |
| IvoRand | 100 | 100 | 100 | 0 | 300 |
| DevilA | 100 | 100 | 50 | 0 | 250 |
| Specificdude | 100 | 100 | 10 | 20 | 230 |
| 2001ASpaceOdyssey | 100 | 100 | 20 | 0 | 220 |
| Hodor | 100 | 100 | 20 | 0 | 220 |
| ghost4e | 100 | 100 | 0 | 0 | 200 |
| moon_finder | 100 | 100 | 0 | 0 | 200 |
| AsparuhBumbarov | 100 | 100 | 0 | 0 | 200 |
| EvgeniLegnev | 100 | 100 | 0 | 0 | 200 |
| kamustna | 100 | 100 | 0 | 0 | 200 |
| TodorJivkov | 100 | 100 | 0 | 0 | 200 |
| stukcho | 100 | 100 | 0 | 0 | 200 |
| VeselaKoleda | 100 | 100 | 0 | 0 | 200 |
| misho_shamara97 | 100 | 100 | 0 | 0 | 200 |
| saturnp | 100 | 100 | 0 | 0 | 200 |
| dj_emily | 100 | 100 | 0 | 0 | 200 |

All entities in this table have the fields **`username`**, **`task1`**, **`task2`**, **`task3`**, **`task4`**, and **`total`** (in that order). If we want to count the most-common value for the **`total`** field/column, the answer would be **`200`** – it appears **`11`** times, whereas all other values appear less times than that (**`220`** appears twice, **`230`** appears once, **`250`** appears once, **`300`** appears twice, **`400`** appears once). If we ask for the most-common value in **`task4`**, we'd get **`0`** (and if we don't consider **`0`** as a value, the most common would be **`100`**).

Your task is to write a program, which, **given the fields of entities in a table**, and then given the **values for each entity**, answers a question about the **most-common value** for **one of the fields** in the table, and counts **how many times** it appears.

## Input

On the first line of the standard input the program will receive the names of the fields (columns) of the entities in the table, separated by single spaces (i.e. the "header" of the table).

On each of the following rows – until a line containing the single string **end** is encountered – the program will receive the values for the fields of one entity, separated by spaces, ordered in the same way the fields are ordered on the first line of input (i.e. the first value on the line is the value of the first field, the second value is the value of the second field, etc.).

After the line containing the string **end**, the program will receive one more line, containing a single string – the name of the field (column) for which the most-common value must be found.

## Output

A single line, containing the most-common value for the field (mentioned in the input), followed by a single space, followed by a positive integer number – the number of occurrences of the most common value.

## Restrictions

The most-common value will occur at least 1 more time than the next most-common (i.e. the input will be such that there is only 1 most-common value for the field we are searching).

The names of the fields will only contain lowercase English letters (**a-z**). The names of the fields will be no more than **20** symbols each.

The values of the fields will only contain English letters (**a-z**, **A-Z**) and the digits from **0** to **9** (inclusive). The values of the fields will be no more than **10** symbols each.

There will be no more than **10** fields per entity.

There will be no more than **200** entities (rows) in the input.

The total running time of your program should be no more than **0.1s**

The total memory allowed for use by your program is **16MB**

## Example I/O

| Example Input | Expected Output | Explanation |
|---|---|---|
| name age height<br>Joro 25 182<br>Oroj 25 182<br>youngerJoro 18 182<br>end<br>age | 25 2 | We are asked about the age column. We have the value 25 twice and the value 18 once. The most-common value is 25 and it appears 2 times |
| name age place weight<br>Joro 25 unknown 87<br>Ina 25 Sofia 53<br>Dog 4 dogland 13<br>Cat 25 catland 5<br>Canary 25 Sofia 1<br>end<br>place | Sofia 2 | We are asked about the place column. We have unknown 1 time, Sofia 2 times, dogland 1 time and catland 1 time. Sofia has the maximum number of occurrences, so the answer is Sofia with a count of 2 |