**Exercises: Associative Arrays**

Problems for exercise for the ["PHP Fundamentals" course @ SoftUni](https://softuni.bg/trainings/2344/php-fundamentals-may-2019).

You can check your solutions in [Judge](https://judge.softuni.bg/Contests/1220/).

* **Gold Miner**

You are given a sequence of strings, each on a new line. Every odd line on the console is representing the type of the gold (e.g. Yellow, Rose, White, and so on), and its karats. Your task is to collect the resources while you receive **"stop"** аnd print each on a new line.

**Print the resources and their quantities in format:**

**"{type} –> {karats}K"**

The karats inputs will be in the range [1 … 24 000]

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| White  18  Yellow  24  Green  10  stop | White -> 18K  Yellow -> 24K  Green -> 10K |
| Rose  14  White  24  Yellow  10  Rose  10  stop | Rose -> 24K  White -> 24K  Yellow -> 10K |

* **Student Academy**

Write a program, which keeps information about **students** and **their grades**.

You will receive **n pair of rows**. First you will receive the **student's name**, after that you will receive **his grade**. **Check if student already exists, and if not, add him**. Keep track of all grades for each student.

When you finish reading data, keep students with **average grade higher or equal to 4.50**. Order filtered students by **average grade in descending**.

**Print the students and their average grade in format:**

**"{name} –> {averageGrade}"**

**Format** the average grade to the **2nd decimal place**.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  John  5.5  John  4.5  Alice  6  Alice  3  George  5 | John -> 5.00  George -> 5.00  Alice -> 4.50 |
| 5  Amanda  3.5  Amanda  4  Rob  5.5  Christian  5  Robert  6 | Robert -> 6.00  Rob -> 5.50  Christian -> 5.00 |

* **Courses**

Write a program, which keeps information about **courses**. Each course has a name and registered students.

You will receive **course** **name** and **student** **name,** until you receive the command "**end**". **Check if such course already exists, and if not, add the course.** Register the user into the course. When you do receive the command "**end**", print the courses with their **names** and **total registered users**, ordered by the **count of registered users** in **descending** order. For each contest print registered users **ordered by name in ascending order**.

**Input**

* Until you receive "**end**", the input come in the format: "**{courseName} : {studentName}**".

**Output**

* Print information about **each** **course**, following the format:   
  **"{courseName}: {registeredStudents}"**
* Print information about each student, following the format:  
  **"-- {studentName}"**

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| Programming Fundamentals : John Smith  Programming Fundamentals : Linda Johnson  JS Core : Will Wilson  Java Advanced : Harrison White  end | Programming Fundamentals: 2  -- John Smith  -- Linda Johnson  JS Core: 1  -- Will Wilson  Java Advanced: 1  -- Harrison White |
| Algorithms : Jay Moore  Programming Basics : Martin Taylor  Python Fundamentals : John Anderson  Python Fundamentals : Andrew Robinson  Algorithms : Bob Jackson  Python Fundamentals : Clark Lewis  end | Python Fundamentals: 3  -- Andrew Robinson  -- Clark Lewis  -- John Anderson  Algorithms: 2  -- Bob Jackson  -- Jay Moore  Programming Basics: 1  -- Martin Taylor |