Exercise: Objects and Classes

Problems for exercises and homework for the "Technology Fundamentals" course @ SoftUni.

You can check your solutions in Judge.

1. Advertisement Message

Write a program that **generate random fake advertisement message** to extol some product. The messages must consist of 4 parts: **laudatory phrase** + **event** + **author** + **city**. Use the following predefined parts:

- **Phrases** – {"Excellent product.", "Such a great product.", "I always use that product.", "Best product of its category.", "Exceptional product.", "I can't live without this product."}
- **Events** – {"Now I feel good.", "I have succeeded with this product.", "Makes miracles. I am happy of the results!", "I cannot believe but now I feel awesome.", "Try it yourself, I am very satisfied.", "I feel great!"}
- **Authors** – {"Diana", "Petya", "Stella", "Elena", "Katya", "Iva", "Annie", "Eva"}
- **Cities** – {"Burgas", "Sofia", "Plovdiv", "Varna", "Ruse"}

The format of the output message is: **`{phrase} {event} {author} – {city}`**.

As an input, you take the **number of messages** to be generated. Print each random message at a separate line.

Examples

Input	Output
3	Such a great product. Now I feel good. Elena – Ruse Excelent product. Makes miracles. I am happy of the results! Katya – Varna Best product of its category. That makes miracles. Eva - Sofia

2. Articles

Create an article class with the following properties:

- **Title** – a string
- **Content** – a string
- **Author** – a string

The class should have a constructor and the following methods:

- **`Edit (new content)`** – change the old content with the new one
- **`ChangeAuthor (new author)`** – change the author
- **`Rename (new title)`** – change the title of the article
- override **ToString** – print the article in the following format: **`"{title} - {content}:{author}"`**

Write a program that reads an article in the following format **`"{title}, {content}, {author}"`**. On the next line, you will get a number **n**. On the next **n lines,** you will get one of the following commands: **`"Edit: {new content}"`**; **`"ChangeAuthor: {new author}"`**; **`"Rename: {new title}"`**. At the end, print the final article.

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

Example

Input	Output

`some title, some content, some author`
`3`
`Edit: better content`
`ChangeAuthor:  better author`
`Rename: better title`
	`better title - better content: better author`

3. Opinion Poll

Using the Person class, write a program that reads from the console **N** lines of personal information and then prints all people whose **age** is **more than 30** years, **sorted in alphabetical order**.

Note: you can use **stream()** to filter people.

Examples

Input	Output

`3`
`Pesho 12`
`Stamat 31`
`Ivan 48`
	`Ivan - 48`
	`Stamat – 31`

`5`
`Nikolai 33`
`Yordan 88`
`Tosho 22`
`Lyubo 44`
`Stanislav 11`
	`Lyubo - 44`
	`Nikolai - 33`
	`Yordan – 88`

4. Articles 2.0

Change the program, so you can store a **list of articles**. You will not need the methods any more (**except the ToString method**). On the **first line**, you will get a number **n**. On the **next n lines**, you will get some **articles in the same format** as the previous task (**"`{title}, {content}, {author}`"**). Finally, you will get one of the **three inputs**: **"title"**, **"content"**, **"author"**. You need to **order the articles** alphabetically based on the command and **print them sorted by the given criteria**.

Example

Input	Output

`2`
`Science, planets, Bill`
`Article, content, Johnny`
`title`
	`Article - content: Johnny`
	`Science - planets: Bill`

`3`
`title1, C, author1`
`title2, B, author2`
	`title3 – A: author3`
	`title2 – B: author2`
	`title1 – C: author1`

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

title3, A, author3 content	

5. Students

Write a program that receives an **n count of students** and **orders them by grade** (in **descending**). Each student should have **First name** (string), **Last name** (string) and **grade** (floating-point number).

Input

- First line will be a number **n**
- Next n lines you will get a student info in the format **"{first name} {second name} {grade}"**

Output

- Print each student in the following format **"{first name} {second name}: {grade}"**

Example

Input	Output
4 Lakia Eason 3.90 Prince Messing 5.49 Akiko Segers 4.85 Rocco Erben 6.00	Rocco Erben: 6.00 Prince Messing: 5.49 Akiko Segers: 4.85 Lakia Eason: 3.90

6. Vehicle Catalogue

You have to make a catalogue for vehicles. You will receive two types of vehicle – **car** or **truck**.

Until you receive the command "**End**" you will receive **lines** of **input** in the format:

```
{typeOfVehicle} {model} {color} {horsepower}
```

After the "**End**" command, you will start receiving **models** of **vehicles**. Print for every received vehicle its **data** in the format:

```
Type: {typeOfVehicle}
Model: {modelOfVehicle}
Color: {colorOfVehicle}
Horsepower: {horsepowerOfVehicle}
```

When you receive the command "**Close the Catalogue**", stop receiving input and print the **average horsepower** for the **cars** and for the **trucks** in the format:

```
"{typeOfVehicles} have average horsepower of {averageHorsepower}."
```

The **average horsepower** is calculated by **dividing** the **sum** of **horsepower** for **all** vehicles of the type by the **total count** of **vehicles** from the **same type**.

Format the answer to the **2nd decimal point**.

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

Constraints

- The type of vehicle will always be **car** or **truck**.
- You will not receive the **same model twice**.
- The received horsepower will be integer in the interval **[1…1000]**
- You will receive at most **50** vehicles.
- **Single** whitespace will be used for **separator**.

Examples

Input	Output

`truck Man red 200`
`truck Mercedes blue 300`
`car Ford green 120`
`car Ferrari red 550`
`car Lamborghini orange 570`
`End`
`Ferrari`
`Ford`
`Man`
`Close the Catalogue`
	`Type: Car`
	`Model: Ferrari`
	`Color: red`
	`Horsepower: 550`
	`Type: Car`
	`Model: Ford`
	`Color: green`
	`Horsepower: 120`
	`Type: Truck`
	`Model: Man`
	`Color: red`
	`Horsepower: 200`
	`Cars have average horsepower of: 413.33.`
	`Trucks have average horsepower of: 250.00.`

7. Order by Age

You will receive an **unknown** number of lines. On each line, you will receive array with **3** elements. **The first** element will be string and represents the name of the person. **The second** element will be a **string** and will represent the **ID** of the person. **The last** element will be an **integer** and represents the **age** of the person.

When you receive the command "**End**", stop taking input and print **all the people**, **ordered** by **age**.

Examples

Input	Output

`Georgi 123456 20`
`Pesho 78911 15`
`Stefan 524244 10`
`End`
	`Stefan with ID: 524244 is 10 years old.`
	`Pesho with ID: 78911 is 15 years old.`
	`Georgi with ID: 123456 is 20 years old.`

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.