Exercise: Objects and Classes

Problems for exercise and homework for the "C# Fundamentals" course @ SoftUni You can check your solutions in Judge

1. Advertisement Message

Create a program that generates random fake advertisement messages to advertise a product. The messages must consist of 4 parts: 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 the following: "{phrase} {event} {author} - {city}."

You will receive 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. Excellent product. Makes miracles. I am happy of the results! Katya -
	Varna. Best product of its category. That makes miracles. Eva - Sofia.

2. Articles

Create a **class Article** with the following properties:

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

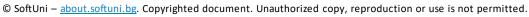
The class should have a constructor and the following methods:

- 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 the **ToString** method print the article in the following format: "{title} - {content}: {author}"

Create a program that reads an article in the following format "{title}, {content}, {author}". On the next line, you will receive a number n, representing the number of commands, which will follow after it. On the next n lines, you will be receiving the following commands:

- "Edit: {new content}"
- "ChangeAuthor: {new author}"



















"Rename: {new title}"

In the end, print the final state of the article.

Example

Input	Output
some title, some content, some author	better title - better content: better
3	author
Edit: better content	
ChangeAuthor: better author	
Rename: better title	
Fight club, love story, Martin Scorsese	Fight club - underground fight club that
2	evolves into much more: Chuck Palahniuk
Edit: underground fight club that	
evolves into much more	
ChangeAuthor: Chuck Palahniuk	

3. Articles 2.0

Change the program from the previous problem in such a way, that you will be able to store a list of articles. You will not need to use the previous methods anymore (except the "ToString()"). On the first line, you will receive the number of articles. On the next lines, you will receive the articles in the same format as in the previous problem: "{title}, {content}, {author}". Print the articles.

Example

Input	Output
2	Science - planets: Bill
Science, planets, Bill	Article - content: Johnny
Article, content, Johnny	
3	title1 - C: author1
title1, C, author1	title2 - B: author2
title2, B, author2	title3 - A: author3
title3, A, author3	

4. Students

Create a program that sorts some students by their grade in descending order. Each student should have:

- First name (string)
- Last name (string)
- **Grade** (a floating-point number)

Input

- On the first line, you will receive a number **n** the **count of all students.**
- On the next **n** lines, you will be receiving information about these students in the following format: "{first name} {second name} {grade}".

















Output

Print out the information about each student in the following format: "{first name} {second name}: {grade}".

Example

Input	Output
4	Rocco Erben: 6.00
Lakia Eason 3.90	Prince Messing: 5.49
Prince Messing 5.49	Akiko Segers: 4.85
Akiko Segers 4.85	Lakia Eason: 3.90
Rocco Erben 6.00	
3	Li Xiao: 5.74
Mary Elizabeth 4.22	Liz Smith: 4.87
Li Xiao 5.74	Mary Elizabeth: 4.22
Liz Smith 4.87	

5. Teamwork Projects

It's time for the teamwork projects and you are responsible for gathering the teams. First, you will receive an integer - the count of the teams you will have to register. You will be given a user and a team, separated with "-". The user is the **creator** of **the team**. For every newly created team you should **print** a message:

"Team {teamName} has been created by {user}!".

Next, you will receive a user with a team, separated with "->", which means that the user wants to join that team. Upon receiving the command: "end of assignment", you should print every team, ordered by the count of its members (descending) and then by name (ascending). For each team, you have to print its members sorted by name (ascending). However, there are several rules:

- If a user tries to **create** a team more than once, a message should be displayed:
 - "Team {teamName} was already created!"
- A creator of a team **cannot create** another team the following message should be thrown:
 - "{user} cannot create another team!"
- If a user tries to **join** a non-existent team, a message should be displayed:
 - "Team {teamName} does not exist!"
- A member of a team cannot join another team the following message should be thrown:
 - "Member {user} cannot join team {team Name}!"
- In the end, teams with zero members (with only a creator) should disband and you have to print them ordered by name in ascending order.
- Every valid team should be printed ordered by name (ascending) in the following format:
 - "{teamName} - {creator} -- {member}..."

Examples

|--|

















2 Team PowerPuffsCoders has been created Tony created a team, by John! which he attempted to John-PowerPuffsCoders join later and this action Team Tony is the best has been created Tony-Tony is the best resulted in throwing a by Tony! Peter->PowerPuffsCoders certain message. Since Member Tony cannot join team Tony is the best! Tony->Tony is the best nobody else tried to join PowerPuffsCoders his team, the team had end of assignment John to disband. -- Peter Teams to disband: Tony is the best 3 Team CloneClub has been created by Note that when a user Tanya! joins a team, you should Tanya-CloneClub first check if the team Team CloneClub was already created! Helena-CloneClub exists and then check if Team SoftUni has been created by Tedy! Tedy-SoftUni the user is already in a Team softUni does not exist! George->softUni team: Team Leda does not exist! George->SoftUni SoftUni Tatyana->Leda Tanya has created - Tedy CloneClub, then she John->SoftUni tried to join a non--- George Cossima->CloneClub existent team and the -- John end of assignment concrete message was CloneClub displayed. - Tanya -- Cossima Teams to disband:

6. Vehicle Catalogue

Until you receive the "End" command, you will be receiving lines of input in the following format:

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

When you receive the "End" command, you will start receiving information about some vehicles.

For every vehicle, print out the information about it in the following **format**:

"Type: {typeOfVehicle} Model: {modelOfVehicle} Color: {colorOfVehicle}

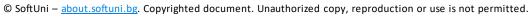
Horsepower: {horsepowerOfVehicle}"

When you receive the "Close the Catalogue" command, print out the average horsepower of the cars and the average horsepower of the trucks in the format:

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

The average horsepower is calculated by dividing the sum of the horsepower of all vehicles of the given type by the total count of all vehicles from that type. Format the answer to the second digit after the decimal point.

















Constraints

- The type of vehicle will always be either a car or a truck.
- You will not receive the same model twice.
- The received horsepower will be an integer in the range [1...1000].
- You will receive at most **50** vehicles.
- The separator will always be single **whitespace**.

Examples

Input	Output
truck Man red 200	Type: Car
truck Mercedes blue 300	Model: Ferrari
car Ford green 120	Color: red
car Ferrari red 550	Horsepower: 550
car Lamborghini orange 570	Type: Car
End	Model: Ford
Ferrari	Color: green
Ford	Horsepower: 120
Man	Type: Truck
Close the Catalogue	Model: Man
	Color: red
	Horsepower: 200
	Cars have average horsepower of: 413.33.
	Trucks have average horsepower of: 250.00.
truck Volvo blue 220	Type: Car
truck Man red 350	Model: Tesla
car Tesla silver 450	Color: silver
car Nio red 650	Horsepower: 450
truck Mack white 430	Type: Car
car Koenigsegg orange 750	Model: Nio
End	Color: red
Tesla	Horsepower: 650
Nio	Type: Truck
Man	Model: Man
Mack	Color: red
Close the Catalogue	Horsepower: 350
	Type: Truck
	Model: Mack
	Color: white
	Horsepower: 430
	Cars have average horsepower of: 616.67.
	Trucks have average horsepower of: 333.33.



















7. Order by Age

You will receive an unknown number of lines. On each line you will receive an array with 3 elements:

- The first element is a string the name of the person
- The second element a string the ID of the person
- The third element is an integer the age of the person

If you get a person whose ID you have already received before, update the name and age for that ID with that of the new person. When you receive the command "End", print all of the people, ordered by age.

Examples

Input	Output
George 123456 20	Stephen with ID: 524244 is 10 years old.
Peter 78911 15	Peter with ID: 78911 is 15 years old.
Stephen 524244 10	George with ID: 123456 is 20 years old.
End	
Lewis 123456 20	Robert with ID: 523444 is 11 years old.
James 78911 15	Jennifer with ID: 345244 is 13 years old.
Robert 523444 11	James with ID: 78911 is 15 years old.
Jennifer 345244 13	Lewis with ID: 123456 is 20 years old.
Mary 52424678 22	Mary with ID: 52424678 is 22 years old.
Patricia 567343 54	Patricia with ID: 567343 is 54 years old.
End	















