# Mid Exam Preparation – 19 June 2024

## Guinea Pig

**Link:** [**https://judge.softuni.org/Contests/Practice/Index/2031#0**](https://judge.softuni.org/Contests/Practice/Index/2031#0)

*Merry has a guinea pig named Puppy, that she loves very much. Every month she goes to the nearest pet store and buys him everything he needs – food, hay, and cover.*

On the **first three lines**, you will receive **the quantity of food**, **hay**, and **cover**, which Merry buys for a **month (30 days)**. On the **fourth line**, you will receive the **guinea pig's weight**.

**Every day** Puppy eats **300 gr of food**. **Every** **second** day Merry **first feeds the pet**, then gives it a **certain amount of hay** **equal to** **5%** of the rest of the **food**. On **every** **third** day, Merry puts Puppy **cover** with **a quantity of** **1/3** of its **weight**.

**Calculate** whether the quantity of **food, hay, and cover**, will be enough for a **month**.

**If Merry runs out of food, hay, or cover, stop the program!**

### Input

* **On the first line – quantity food in kilograms** - afloating-point number in the range **[0.0 – 10000.0].**
* **On the second line – quantity hay in kilograms** - afloating-point number in the range **[0.0 – 10000.0].**
* **On the third line – quantity cover in kilograms** - afloating-point number in the range **[0.0 – 10000.0].**
* **On the fourth line – guinea's weight in kilograms** - afloating-point number in the range **[0.0 – 10000.0].**

### Output

* If the food, the hay, and the cover are enough, print:
  + **"Everything is fine! Puppy is happy! Food: {excessFood}, Hay: {excessHay}, Cover: {excessCover}."**
* If one of the things is not enough, print:
  + **"Merry must go to the pet store!"**

**The output values must be formatted to the second decimal place!**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 10  5  5.2  1 | Everything is fine! Puppy is happy! Food: 1.00, Hay: 1.10, Cover: 1.87. |
| You receive food – **10000**, hay – **5000**, cover – **5200**, weight – **1000** (in grams).  On the first day, Merry gives Puppy 300gr food – 9700gr food left.  On the second day, the food left is **9400gr**, so the needed hay is **9400 \* 5% = 470**,and thehay left is **4530.**  On the third day, the cover left is **4866.67,** and the food left is **9100**,and so on.  On the last day, Merry has: food – 1.00, hay – 1.10, and cover – 1.87. | |
| 1  1.5  3  1.5 | Merry must go to the pet store! |
| 9  5  5.2  1 | Merry must go to the pet store! |

## Shopping List

**Link:** [**https://judge.softuni.bg/Contests/Practice/Index/2031#1**](https://judge.softuni.bg/Contests/Practice/Index/2031#1)

*It’s the end of the week and it is time for you to go shopping, so you need to create a shopping list first.*

### Input

You will receive an **initial list** with groceries separated by **"!"**.

After that you will be receiving **4 types** of commands, until you receive **"Go Shopping!"**

* **Urgent {item} -** **add** the item at the **start** of the list. If the item **already exists,** skip this command.
* **Unnecessary {item} - remove** the item with the given name, only **if it exists** in the list. Otherwise skip this command.
* **Correct {oldItem} {newItem} –** if the item with the given **old name** exists, **change** its name with the **new** one. If it **doesn't exist**, skip this command.
* **Rearrange {item} -** if the grocery exists in the list, **remove** it from its **current position** and **add** it at the **end** of the list.

### Constraints

* There won`t be any duplicate items in the initial list

### Output

Print the **list** with all the groceries, joined by **", ".**

* **"{firstGrocery}, {secondGrocery}, …{nthGrocery}"**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Tomatoes!Potatoes!Bread  Unnecessary Milk  Urgent Tomatoes  Go Shopping! | Tomatoes, Potatoes, Bread |
| **Input** | **Output** |
| Milk!Pepper!Salt!Water!Banana  Urgent Salt  Unnecessary Grapes  Correct Pepper Onion  Rearrange Grapes  Correct Tomatoes Potatoes  Go Shopping! | Milk, Onion, Salt, Water, Banana |

## Inventory

**Link:** [**https://judge.softuni.org/Contests/Practice/Index/2028#2**](https://judge.softuni.org/Contests/Practice/Index/2028#2)

You will receive a journal with some collecting items, separated with a comma and a space (**", "**). After that, until receiving "Craft!" you will be receiving different commands split by **" - "**:

* "Collect - {item}" - you should add the given item to your inventory. If the item already **exists**, you should **skip** this line.
* "Drop - {item}" - you should remove the item from your inventory **if it exists**.
* "Combine Items - {old\_item}:{new\_item}" - you should check if the **old item exists**. If so, **add** the new item **after** the old one. Otherwise, **ignore** the command.
* "Renew – {item}" – if the given item exists, you should change its position and **put it last** in your inventory.

### Output

After receiving "Craft!" print the items in your inventory, separated by **", "**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Iron, Wood, Sword  Collect - Gold  Drop - Wood  Craft! | Iron, Sword, Gold |
| Iron, Sword  Drop - Bronze  Combine Items - Sword:Bow  Renew - Iron  Craft! | Sword, Bow, Iron |