Problem 1 – Password Reset

A problem for exam preparation for the "C# Fundamentals" course @ SoftUni
Submit your solutions in the SoftUni Judge system here

Yet again, you have forgotten your password. Naturally, it's not the first time this has happened. Actually, you got so tired of it that you decided to help yourself with an intelligent solution.

Write a password reset program that performs a series of commands upon a predefined string. First, you will receive a string, and afterward, until the command **"Done"** is given, you will be receiving strings with commands split by a single space. The commands will be the following:

- **"TakeOdd"**
  - Takes only the characters at **odd indices** and **concatenates** them to obtain the **new raw password** and then **prints** it.
- **"Cut {index} {length}"**
  - Gets the substring with the **given length** starting from the **given index** from the password and removes its **first occurrence**, then **prints** the password on the console.
  - The given index and the length will **always** be **valid**.
- **"Substitute {substring} {substitute}"**
  - If the raw password contains the given substring, replaces all of its occurrences with the substitute text given and prints the result.
  - If it doesn't, prints **"Nothing to replace!"**.

Input

- You will be receiving strings until the **"Done"** command is given.

Output

- After the **"Done"** command is received, print:
  - **"Your password is: {password}"**

Constraints

- The indexes from the "**Cut {index} {length}**" command will always be valid.

Examples

Input	Output
Siiceercaroetavm!:?:ahsott.:i:nstupmomceqr TakeOdd Cut 15 3 Substitute :: - Substitute | ^ Done	icecream::hot::summer icecream::hot::mer icecream-hot-mer Nothing to replace! Your password is: icecream-hot-mer

Comments
TakeOdd → We only take the chars at odd indices **1, 3, 5 etc.**
Siiceercaroetavm!:?:ahsott.:i:nstupmomceqr → **icecream::hot::summer**
Cut 15 3 → We cut a substring starting at index 15 with length 3, then remove it from the raw password:
icecream::hot::summer → sum
Substitute :: → We replace "::" with "-":
icecream::hot::summer → icream-hot-summer
Substitute | ^ → **"|"** is not found anywhere in the raw password, so we print **"Nothing to replace!"**
Finally, after receiving the **"Done"** command, we print the resulting password in the proper format.

Input	Output
up8rgoyg3r1atmlmpiunagt!-irs7!1fgulnnnqy	programming!is!funny
TakeOdd	programming!is!fun
Cut 18 2	programming***is***fun
Substitute ! ***	Nothing to replace!
Substitute ? .!.	Your password is: programming***is***fun
Done	

Problem 2 – Mirror Words

A problem for exam preparation for the "C# Fundamentals" course @ SoftUni
Submit your solutions in the SoftUni Judge system here

The SoftUni Spelling Bee competition is here. But it`s not like any other Spelling Bee competition out there. It`s different and a lot more fun! You, of course, are a participant, and you are eager to show the competition that you are the best, so go ahead, learn the rules and win!

On the first line of the input, you will be given a **text string**. To win the competition, you have to find all hidden **word pairs**, read them, and mark the ones that are **mirror images** of each other.

First of all, you have to **extract the hidden word pairs**. Hidden word pairs are:

- Surrounded by **"@"** or **"#"** (only one of the two) in the following pattern **#wordOne##wordTwo#** or **@wordOne@@wordTwo@**
- At least **3 characters long each** (**without the surrounding symbols**)
- Made up of **letters only**

If the second word, **spelled backward,** is the **same as the first word and vice versa** (**casing matters**!), they are a **match,** and you have to store them somewhere. **Examples** of mirror words:

#Part##traP# @leveL@@Level@ #sAw##wAs#

- If you don`t find any valid pairs, print: **`"No word pairs found!"`**
- If you find valid pairs print their count: **`"{valid pairs count} word pairs found!"`**
- If there are no mirror words, print: **`"No mirror words!"`**
- If there are mirror words print:

  `"The mirror words are:`

  `{wordOne} <=> {wordtwo}, {wordOne} <=> {wordtwo}, … {wordOne} <=> {wordtwo}"`

Input / Constraints

- You will receive a string.

Output

- Print the proper output messages in the proper cases as described in the problem description.
- If there are pairs of mirror words, print them in the end, each pair separated by **`", "`**.
- Each pair of mirror word must be printed with **`" <=> "`** between the words.

Examples

<table>
<tr><th colspan="2">Input</th></tr>
<tr><td colspan="2">@mix#tix3dj#poOl##loOp#wl@@bong&song%4very$long@thong#Part##traP##@@leveL@@Level@##car#rac##tu@pack@@ckap@#rr#sAw##wAs#r#@w1r</td></tr>
<tr><th>Output</th><th>Comments</th></tr>
<tr><td>5 word pairs found!
The mirror words are:
Part <=> traP, leveL <=> Level, sAw <=> wAs</td><td>There are 5 green and yellow pairs that meet all requirements and thus are valid.
#poOl##loOp# is valid and looks very much like a mirror words pair, but it isn't because the casings don't match.
#car#rac# "rac" spelled backward is "car", but this is not a valid pair because there is only one "#" between the words.
@pack@@ckap@ is also valid, but "ckap" backward is "pakc" which is not the same as "pack", so they are not mirror words.</td></tr>
<tr><th colspan="2">Input</th></tr>
<tr><td colspan="2">#po0l##l0op# @bAc##cAB@ @LM@ML@ #xxxXxx##xxxXxx# @aba@@ababa@</td></tr>
<tr><th>Output</th><th>Comments</th></tr>
</table>

`2 word pairs found!`
`No mirror words!`
	"xxxXxx" backward is not the same as "xxxXxx"
	@aba@@ababa@ is a valid pair, but the word lengths are different – these are definitely not mirror words

Input	
`#lol#lol# @#God@@doG@# #abC@@Cba# @Xyu@#uyX#`	
Output	**Comments**

`No word pairs found!`
`No mirror words!`	

Problem 3 – P!rates

A problem for exam preparation for the ["C# Fundamentals" course @ SoftUni]
Submit your solutions in the SoftUni Judge system [here]

Anno 1681. The Caribbean. The golden age of piracy. You are a well-known pirate captain by the name of Jack Daniels. Together with your comrades Jim (Beam) and Johnny (Walker), you have been roaming the seas, looking for gold and treasure… and the occasional killing, of course. Go ahead, target some wealthy settlements and show them the pirate's way!

Until the **"Sail"** command is given, you will be receiving:

- You and your crew have targeted **cities**, with their **population** and **gold**, separated by **"||"**.
- If you receive a city that has already been received, you have to increase the population and gold with the given values.

After the **"Sail"** command, you will start receiving lines of text representing events until the **"End"** command is given.

Events will be in the following format:

- **"Plunder=>{town}=>{people}=>{gold}"**
  - You have successfully attacked and plundered the town, killing the given number of people and stealing the respective amount of gold.
  - For every town you attack print this message: **"{town} plundered! {gold} gold stolen, {people} citizens killed."**
  - If any of those two values (population or gold) **reaches zero**, the town is disbanded.
    - You need to **remove it** from your collection of targeted cities and print the following message: **"{town} has been wiped off the map!"**
  - There will be no case of receiving more people or gold than there is in the city.
- **"Prosper=>{town}=>{gold}"**
  - There has been dramatic economic growth in the given city, **increasing its treasury** by the given amount of gold.
  - The gold amount **can be a negative number, so be careful.** If a negative amount of gold is given, print: **"Gold added cannot be a negative number!" and ignore the command.**

- If the given gold is a valid amount, increase the town's gold reserves by the respective amount and print the following message:

"{gold added} gold added to the city treasury. {town} now has {total gold} gold."

Input

- On the first lines, until the **"Sail"** command, you will be receiving strings representing the cities with their gold and population, separated by **"||"**
- On the following lines, until the **"End"** command, you will be receiving strings representing the actions described above, separated by **"=>"**

Output

- After receiving the **"End"** command, if there are any existing settlements on your list of targets, you need to print all of them, in the following format:

```
"Ahoy, Captain! There are {count} wealthy settlements to go to:
{town1} -> Population: {people} citizens, Gold: {gold} kg
{town2} -> Population: {people} citizens, Gold: {gold} kg
…
{town…n} -> Population: {people} citizens, Gold: {gold} kg"
```

- If there are no settlements left to plunder, print:

"Ahoy, Captain! All targets have been plundered and destroyed!"

Constraints

- The initial population and gold of the settlements will be valid 32-bit integers, never negative, or exceed the respective limits.
- The town names in the events will always be valid towns that should be on your list.

Examples

Input	Output
Tortuga||345000||1250 Santo Domingo||240000||630 Havana||410000||1100 Sail Plunder=>Tortuga=>75000=>380 Prosper=>Santo Domingo=>180 End	Tortuga plundered! 380 gold stolen, 75000 citizens killed. 180 gold added to the city treasury. Santo Domingo now has 810 gold. Ahoy, Captain! There are 3 wealthy settlements to go to: Tortuga -> Population: 270000 citizens, Gold: 870 kg Santo Domingo -> Population: 240000 citizens, Gold: 810 kg Havana -> Population: 410000 citizens, Gold: 1100 kg

Input	Output
Nassau||95000||1000 San Juan||930000||1250 Campeche||270000||690 Port Royal||320000||1000 Port Royal||100000||2000 Sail Prosper=>Port Royal=>-200 Plunder=>Nassau=>94000=>750 Plunder=>Nassau=>1000=>150 Plunder=>Campeche=>150000=>690 End	Gold added cannot be a negative number! Nassau plundered! 750 gold stolen, 94000 citizens killed. Nassau plundered! 150 gold stolen, 1000 citizens killed. Nassau has been wiped off the map! Campeche plundered! 690 gold stolen, 150000 citizens killed. Campeche has been wiped off the map! Ahoy, Captain! There are 2 wealthy settlements to go to: San Juan -> Population: 930000 citizens, Gold: 1250 kg Port Royal -> Population: 420000 citizens, Gold: 3000 kg