Problem 1 – Computer Store

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

Write a program that **prints you a receipt** for your new computer. You will receive the **parts' prices (without tax)** until you receive what type of customer this is – **special** or **regular**. Once you receive the type of customer you should print the receipt.

The **taxes are 20%** of each part's price you receive.

If the customer is **special**, he has a 10% discount on the total price with taxes.

If a given price is not a positive number, you should print **"Invalid price!"** on the console and continue with the next price.

If the total price is equal to zero, you should print **"Invalid order!"** on the console.

Input

- You will receive numbers representing **prices (without tax)** until command **"special"** or **"regular":**

Output

- The receipt should be in the following format:

"Congratulations you've just bought a new computer!

Price without taxes: {total price without taxes}$

Taxes: {total amount of taxes}$

Total price: {total price with taxes}$"

Note: All prices should be displayed to the second digit after the decimal point! The discount is applied only on the total price. Discount is only applicable to the final price!

Examples

Input	Output
1050 200 450 2 18.50 16.86 special	Congratulations you've just bought a new computer! Price without taxes: 1737.36$ Taxes: 347.47$ ---------- Total price: 1876.35$
Comment	
1050 – valid price, total **1050** **200** – valid price, total **1250** ... **16.86** – valid price, total **1737.36** We receive **special**	

Price is positive number, so it is valid order Price without taxes is **1737.36** Taxes: **20**% from **1737.36** = **347.47** Final price = **1737.36** + **347.47** = **2084.83** Additional **10**% discount for special customers **2084.83** – **10**% = **1876.35**

Input	Output
1023 15 -20 -5.50 450 20 17.66 19.30 regular	Invalid price! Invalid price! Congratulations you've just bought a new computer! Price without taxes: 1544.96$ Taxes: 308.99$ ---------- Total price: 1853.95$
regular	Invalid order!

Problem 2 – MuOnline

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

You have **initial health 100** and **initial bitcoins 0**. You will be given **a string representing the dungeon's rooms**. Each room is separated with **'|'** (vertical bar): **"room1|room2|room3…"**

Each room contains **a command** and a **number**, separated by space. The command can be:

- **"potion"**
  - You are healed with the number in the second part. But your health **cannot exceed** your **initial health (100)**.
  - First print: **"You healed for {amount} hp."**
  - After that, print your current health: **"Current health: {health} hp."**
- **"chest"**
  - You've found some bitcoins, the number in the second part.
  - Print: **"You found {amount} bitcoins."**
- In **any other case,** you are **facing a monster**, which you will **fight**. The **second part of the room** contains the **attack** of the monster. You should remove the monster's attack from your health.
  - If you are not dead (health <= 0), you've slain the monster, and you should print: **"You slayed {monster}."**
  - If you've died, print **"You died! Killed by {monster}."** and your quest is over. Print the best room you've manage to reach: **"Best room: {room}"**

If you managed to **go through all the rooms** in the dungeon, print on the **following three lines**:

"You've made it!"

"Bitcoins: {bitcoins}"

"Health: {health}"

Input / Constraints

You receive a **string** representing the dungeon's rooms, separated with '**|**' (vertical bar): **"room1|room2|room3…"**.

Output

Print the corresponding messages described above.

Examples

Input	Output
rat 10|bat 20|potion 10|rat 10|chest 100|boss 70|chest 1000	You slayed rat. You slayed bat. You healed for 10 hp. Current health: 80 hp. You slayed rat. You found 100 bitcoins. You died! Killed by boss. Best room: 6
Input	**Output**
cat 10|potion 30|orc 10|chest 10|snake 25|chest 110	You slayed cat. You healed for 10 hp. Current health: 100 hp. You slayed orc. You found 10 bitcoins. You slayed snake. You found 110 bitcoins. You've made it! Bitcoins: 120 Health: 65

Problem 3 – Hearth Delivery

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

Valentine's day is coming, and Cupid has minimal time to spread some love across the neighborhood. Help him with his mission!

You will receive a **string** with **even integers,** separated by a **"@"** - this is our neighborhood. After that, a series of **Jump** commands will follow until you receive **"Love!"**. Every house in the neighborhood needs a certain number of **hearts** delivered by Cupid so it can celebrate Valentine's day. The integers in the neighborhood indicate those needed hearts.

Cupid starts at the position of the **first house** (index **0**) and must jump by a **given length.** The jump commands will be in this format: **"Jump {length}"**.

Every time he jumps from one house to another, the needed hearts for the visited house are **decreased by 2**:

- If the needed hearts for a certain house become **equal to 0**, print on the console **"Place {house_index} has Valentine's day."**
- If **Cupid** jumps to a house where the needed hearts are **already 0,** print on the console **"Place {house_index} already had Valentine's day."**
- Keep in mind that **Cupid** can have a **larger jump length** than the **size of the neighborhood,** and if he does jump **outside** of it, he should **start** from the **first house** again (index 0)

For example, we are given this neighborhood: 6@6@6. Cupid is at the start and jumps with a length of 2. He will end up at index 2 and decrease the needed hearts by 2: [6, 6, 4]. Next, he jumps again with a length of 2 and goes outside the neighborhood, so he goes back to the first house (index 0) and again decreases the needed hearts there: [4, 6, 4].

Input

- On the first line, you will receive a **string** with **even integers** separated by **"@"** – the neighborhood and the number of hearts for each house.
- On the next lines, until **"Love!"** is received, you will be getting jump commands in this format: **"Jump {length}"**.

Output

In the end, print **Cupid's last position** and whether his mission was successful or not:

- **"Cupid's last position was {last_position_index}."**
- If **each house** has had Valentine's day, print:
  - **"Mission was successful."**
- If **not,** print the **count** of all houses that **didn't** celebrate Valentine's Day:
  - **"Cupid has failed {houseCount} places."**

Constraints

- The **neighborhood's** size will be in the range [**1…20**]
- Each **house** will need an **even number** of hearts in the range [**2…10**]
- Each **jump length** will be an integer in the range [**1…20**]

Examples

Input	Output	Comments

10@10@10@2 Jump 1 Jump 2 Love!	Place 3 has Valentine's day. Cupid's last position was 3. Cupid has failed 3 places.	Jump 1 ->> [10, 8, 10, 2] Jump 2 ->> [10, 8, 10, 0] so we print "Place 3 has Valentine's day." The following command is "Love!" so we print Cupid's last position and the outcome of his mission.
2@4@2 Jump 2 Jump 2 Jump 8 Jump 3 Jump 1 Love!	Place 2 has Valentine's day. Place 0 has Valentine's day. Place 0 already had Valentine's day. Place 0 already had Valentine's day. Cupid's last position was 1. Cupid has failed 1 places.	