# Mid Exam Preparation – 20 June 2022

## Computer Store

**Link:** [**https://judge.softuni.org/Contests/Practice/Index/2517#0**](https://judge.softuni.org/Contests/Practice/Index/2517#0)

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! |

## MuOnline

**Link:** [**https://judge.softuni.org/Contests/Practice/Index/2028#1**](https://judge.softuni.org/Contests/Practice/Index/2028#1)

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 |

1. **Moving Target**

**Link:** [**https://judge.softuni.org/Contests/PrSactice/Index/2305#2**](https://judge.softuni.org/Contests/PrSactice/Index/2305#2)

You are at the shooting gallery again, and you need a program that helps you keep track of moving targets. On the first line, you will receive a **sequence of targets with their integer values**, split by a **single space**. Then, you will start receiving **commands for manipulating the targets** until the **"End"** command. The commands are the following:

* **"Shoot {index} {power}"**
  + Shoot the target at the index **if it exists** by **reducing** its **value** by the **given** **power** (**integer value**).
  + Remove the target **if it is shot**. A target is considered **shot** when **its value reaches 0**.
* **"Add {index} {value}"**
  + Insert a target with the received value at the received **index if it exists**.
  + If not, print: **"Invalid placement!"**
* **"Strike {index} {radius}"**
  + **Remove** the target at the given **index** and **the ones before and after it** depending on the **radius**.
  + If **any of the indices** in the range is **invalid**, print: **"Strike missed!"** and **skip** this command.

**Example:** **"Strike 2 2"**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | {radius} | {radius} | {strikeIndex} | {radius} | {radius} |  |  |

* **"End"**
  + **Print** the sequence with targets in the following format and **end** **the program**:

**"{target1}|{target2}…|{targetn}"**

### Input / Constraints

* On the **first line,** you will receive **the sequence of targets** – **integer values [1-10000]**.
* On the **following lines,** until the **"End"** will be receiving the command described above – **strings**.
* There will never be a case when the **"Strike"** command would empty the whole sequence.

### Output

* Print the appropriate message in case of any command if necessary.
* In the end, print the sequence of targets in the format described above.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 52 74 23 44 96 110  Shoot 5 10  Shoot 1 80  Strike 2 1  Add 22 3  End | Invalid placement!  52|100 | The first command is "**Shoot**", so we reduce the target on **index** **5**, which is valid, with the given **power** – **10**.  Then we receive the same command, but we need to reduce the target on the 1st index, with power 80. The value of this target is 74, so it is considered shot, and we **remove** it.  Then we receive the "**Strike**" command on the 2nd index, and we need to check if the range with radius 1 is valid:  **52 23 44 96 100**  And it is, so we **remove** the targets.  At last, we receive the "**Add**" command, but the index is **invalid**, so we print the appropriate **message**, and in the end, we have the following result:  **52|100** |
| 1 2 3 4 5  Strike 0 1  End | Strike missed!  1|2|3|4|5 |  |