Problem 4 – Game of Epicness

Welcome to the Game of Epicness where different kingdoms are fighting for the grant price of a bucket of Bitcoins… EPIC! In this amazing game there are many kingdoms with many generals and every general have their own army. To decide who is the winner for this totally amazing price they fight battles between them. But they are not so awesome at math, so they need you to help them record their battle results.

Write a JavaScript program that **determines** the **winner** from **all battles**. You will receive **two** arguments:

The **first** argument is an **array of kingdoms** with **generals** and their **army** in the form of an **object** with format:

```
{ kingdom: String, general: String, army: Number }
```

Every **general** has his own **army** that fights for a certain **kingdom**. Note that, every **kingdom's name** is **unique,** and every general's **name** is **unique** in **this kingdom**. If the **general** already **exists in** this **kingdom add** the **army** to his current one. After you go through all the kingdoms with their generals with armies and store the information about them, it's time to start the battles.

The **second** argument is **matrix of strings** showing which **kingdom's generals** are **fighting** in this format:

[

["{AttackingKingdom} ", "{AttackingGeneral}", "{DefendingKingdom} ", "{DefendingGeneral}"],

…

]

The **first two elements** are the **names** of the **attacking general from** certain **kingdom** and the **second two** are the **names** of the **defending general from** certain **kingdom**. **Compare** the two general's **armies to determine** who **wins** and who **losses** based on who have the **larger army wins.** The **winner's army increases** with **10%** and the **loser's army decreases** with **10%.** Keep in mind to **round** them **down** if there is any excess **army after the battle.** If there is a **draw**, **do not do anything**. **Keep track** of the **wins** and **losses** for every general's battle.

Note that, **generals** from the **same kingdom cannot attack each other**.

After you finish with all battles you need to **find** which **kingdom wins** the game. To decide that, **first order them** by all their **general's wins (descending)** then by their **losses (ascending),** and finally by the **kingdom's name** in **ascending alphabetical** order.

Input

You will receive **two arguments –** an **array of objects** with properties and a **matrix of strings** as shown above.

Output

Print on the **console** the winning kingdom and **sort** the generals by their **armies in descending** order, **formatted** as seen in the examples.

Constraints

- The **number** of **elements** in the **first input argument** will be in range **[1..100] inclusive**
- The **number** of **elements** in the **second input argument** will be in range **[0..100] inclusive**
- General's **army** will be always an **integer** in range **[0..1,000,000] inclusive**
- There **will** be **no invalid input**
- There **will** be **no matching number** of **armies** in the **output**

Examples

Input

`[ { kingdom: "Maiden Way", general: "Merek", army: 5000 },`
`  { kingdom: "Stonegate", general: "Ulric", army: 4900 },`
`  { kingdom: "Stonegate", general: "Doran", army: 70000 },`
`  { kingdom: "YorkenShire", general: "Quinn", army: 0 },`
`  { kingdom: "YorkenShire", general: "Quinn", army: 2000 },`
`  { kingdom: "Maiden Way", general: "Berinon", army: 100000 } ],`
`[ ["YorkenShire", "Quinn", "Stonegate", "Ulric"],`
`  ["Stonegate", "Ulric", "Stonegate", "Doran"],`
`  ["Stonegate", "Doran", "Maiden Way", "Merek"],`
`  ["Stonegate", "Ulric", "Maiden Way", "Merek"],`
`  ["Maiden Way", "Berinon", "Stonegate", "Ulric"] ]`

Output

`Winner: Stonegate`
`/\general: Doran`
`---army: 77000`
`---wins: 1`
`---losses: 0`
`/\general: Ulric`
`---army: 5336`
`---wins: 2`
`---losses: 1`

Explanation
After you successfully store the kingdoms information, the first battle's result is victory for the defender Ulric and a loss for the attacker Quinn. Second battle is ignored because the generals are from the same kingdom. Third battle is a victory for Doran and a loss for Merek. Fourth battle is a win for Ulric and a loss for Merek. Fifth battle is a victory for Berinon and a defeat for Ulric. All winners increase their armies with 10% for each win and all losers decrease their armies with 10% for each loss. The result from the battles are – Stonegate: 3 wins and 1 loss; Maiden Way: 1 win and 2 losses; YorkenShire: 0 wins and 1 loss. Making Stonegate the winner of the games because they have the most wins from kingdoms.

Input

`[ { kingdom: "Stonegate", general: "Ulric", army: 5000 },`
`  { kingdom: "YorkenShire", general: "Quinn", army: 5000 },`
`  { kingdom: "Maiden Way", general: "Berinon", army: 1000 } ],`
`[ ["YorkenShire", "Quinn", "Stonegate", "Ulric"],`
`  ["Maiden Way", "Berinon", "YorkenShire", "Quinn"] ]`

Output

`Winner: YorkenShire`
`/\general: Quinn`
`---army: 5500`
`---wins: 1`
`---losses: 0`

Explanation
The first battle between Quinn and Ulric is a draw because they have even armies because of that it is not recorded and their armies size does not change. The second battle is a win for Quinn and a loss for Berinon making YorkenShire the winner of the game with 1 win and 0 losses.

Input

`[ { kingdom: "Maiden Way", general: "Merek", army: 5000 },`
`  { kingdom: "Stonegate", general: "Ulric", army: 4900 },`
`  { kingdom: "Stonegate", general: "Doran", army: 70000 },`
`  { kingdom: "YorkenShire", general: "Quinn", army: 0 },`
`  { kingdom: "YorkenShire", general: "Quinn", army: 2000 } ],`
`[ ["YorkenShire", "Quinn", "Stonegate", "Doran"],`
`  ["Stonegate", "Ulric", "Maiden Way", "Merek"] ]`

Output

`Winner: Maiden Way`
`/\general: Merek`
`---army: 5500`
`---wins: 1`
`---losses: 0`