# Problem 3 - Man-O-War

Problem for exam preparation for the [Programming Fundamentals Course @SoftUni](https://softuni.bg/courses/programming-fundamentals-csharp-java-js-python).

Submit your solutions in the SoftUni judge system at <https://judge.softuni.org/Contests/Practice/Index/1773#2>.

*The pirates encounter a huge Man-O-War at sea.*

Create a program that **tracks** the **battle** and either chooses a **winner** or prints a **stalemate**. On the **first line,** you will receive the **status** of the **pirate ship**, which is a **string** representing **integer sections** separated by **">"**. On **the second line,** you will receive the **same** type of status, but for the **warship**:

**"{section1}>{section2}>{section3}… {sectionn}"**

On the **third line,** you will receive the **maximum health capacity** a section of the ship can reach.

The following lines represent commands **until** **"Retire"**:

* **"Fire {index} {damage}"** - the pirate ship **attacks** the warship with the **given damage** at that section. Check if the **index is valid** and if not, **skip** the command. If the section **breaks** (health <= 0) the warship **sinks**, print the following and **stop** the program: **"You won! The enemy ship has sunken."**
* **"Defend {startIndex} {endIndex} {damage}"** - the warship **attacks** the pirate ship with the **given damage** at that **range** (**indexes are inclusive)**. Check if both **indexes are valid** and if not, **skip** the command. If the section **breaks** (health <= 0) the pirate ship **sinks**, print the following and **stop** the program:

**"You lost! The pirate ship has sunken."**

* **"Repair {index} {health}"** - the crew **repairs** a section of the **pirate ship** with the **given health**. Check if the **index is valid** and if not, **skip** the command. The health of the section **cannot** exceed the **maximum health capacity**.
* **"Status"** - prints the **count** of all sections of the **pirate ship** that need repair soon, which are all sections that are **lower than 20%** of the **maximum** **health capacity**. Print the following:

**"{count} sections need repair."**

In the end, if a **stalemate** occurs, print the **status** of **both** ships, which is the **sum** of their individual sections, in the following format:

**"Pirate ship status: {pirateShipSum}**

**Warship status: {warshipSum}"**

## Input

* On the **1st line,** you are going to receive the **status** of the **pirate ship** (**integers** separated by **'>'**)
* On the **2nd line,** you are going to receive the **status** of the **warship**
* On the **3rd line,** you will receive the **maximum health** a section of a ship can reach.
* On the following **lines**, until **"Retire"**, you will be receiving commands.

## Output

* Print the output in the **format** **described** **above**.

## Constraints

* The **section numbers** will be integers in the range [**1**….**1000**]
* The **indexes** will be integers [**-200**….**200**]
* The **damage** will be an integer in the range [**1**….**1000**]
* The **health** will be an integer in the range [**1**….**1000**]

## Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | |
| 12>13>11>20>66  12>22>33>44>55>32>18  70  Fire 2 11  Fire 8 100  Defend 3 6 11  Defend 0 3 5  Repair 1 33  Status  Retire | 2 sections need repair.  Pirate ship status: 135  Warship status: 205 | |
| **Comments** | | |
| First, we receive the command "**Fire 2 11**", and damage the warship at section index 2, which is currently 33, and after reduction, the status of the warship is the following:  **12 22 22 44 55 32 18**  The **second** and **third** commands have **invalid indexes**, so we skip them.  The **fourth** command, **"Defend 0 3 5"** damages **4 sections** of the pirate ship with **5,** which results in the following states:  **7 8 6 15 66**  The **fifth** command, **"Repair 1 33"** repairs the pirate ship section and adds **33 health** to the current **8,** which results in **41**  Only **2 sections** of the pirate ship (**7** and **6**) need repair soon.  In the end, there is a **stalemate,** so we print both ship statuses (**sum** of all sections). | | |
| **Input** | | **Output** |
| 2>3>4>5>2  6>7>8>9>10>11  20  Status  Fire 2 3  Defend 0 4 11  Repair 3 18  Retire | 3 sections need repair.  You lost! The pirate ship has sunken. | |

## JS Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| (["12>13>11>20>66",  "12>22>33>44>55>32>18",  "70",  "Fire 2 11",  "Fire 8 100",  "Defend 3 6 11",  "Defend 0 3 5",  "Repair 1 33",  "Status",  "Retire"]) | 2 sections need repair.  Pirate ship status: 135  Warship status: 205 |
| **Comments** | |
| First, we receive the command "**Fire 2 11**", and damage the warship at section index 2, which is currently 33, and after reduction, the status of the warship is the following:  **12 22 22 44 55 32 18**  The **second** and **third** commands have **invalid indexes**, so we skip them.  The **fourth** command, **"Defend 0 3 5"** damages **4 sections** of the pirate ship with **5,** which results in the following states:  **7 8 6 15 66**  The **fifth** command, **"Repair 1 33"** repairs the pirate ship section and adds **33 health** to the current **8,** which results in **41**  Only **2 sections** of the pirate ship (**7** and **6**) need repair soon.  In the end, there is a **stalemate,** so we print both ship statuses (**sum** of all sections). | |
| **Input** | **Output** |
| (["2>3>4>5>2",  "6>7>8>9>10>11",  "20",  "Status",  "Fire 2 3",  "Defend 0 4 11",  "Repair 3 18",  "Retire"]) | 3 sections need repair.  You lost! The pirate ship has sunken. |