# Problem 2 – Snowmen

The Snowmen Fray has been on since the dawn of time. Professor Olaf is currently studying the ancient war methods, and needs you to write a program that simulates a battle between snowmen.

You will receive a **sequence** of **integers**, **separated** by **spaces** – the **snowmen**,   
which are **indexed** from **0** – **sequence.length – 1**.

You must **traverse the elements**, from the **first** till the **last**. Each element is an **attacker index**, and its **integer value** is its **target index**. If the **integer value** is **greater than** the **length** of the **sequence**, **divide** it **modulo** (**%**) by the length of the sequence. **Example**: **target** = **12**, **length** = **7**, **target** = **12 % 7** - > **5**.

When you extract **attacker index** and the **target index**, you must calculate the **absolute value** of the **difference** between them.

* If the **difference** is an **even number**, the **attacker wins**.
* if the **difference** is an **odd number**, the **target wins**.
* If the **attacker** and the **target** are **equal** (they are **the same indexes**), that means that snowman **suicides**.

If an element **loses** a battle or **suicides**, it **should NOT be able** to **attack**. It **can** still be a **target** though.

When you finish traversing the whole sequence, you must **remove all elements** that have **lost** or **suicided**, and then you must **start over**. The process must be **repeated** until there is **1** **snowman left** in the **sequence**.

For each **attacker** and **target**, you must print “{attacker} x {target} -> {winner} wins”.   
The **attacker** and the **target** are **indexes**, and the winner is the **index** of the **winner**.

In that case you must print “{attacker} performed harakiri”.

## Input

* As input you will receive a **single input line** containing the **sequence** of **integers**, **separated** by **spaces**.

## Output

* As output you must print each action between attacker and target, in the formats specified above.

## Constraints

* The **elements** of the **sequence** will be **between 0** and **100**.
* The **integers** in the **sequence** will be in **range [0, 1000]**.
* Allowed working **time** / **memory**: **100ms** / **16MB**.

## Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comment** |
| 4 3 2 1 0 | 0 x 4 -> 0 wins  1 x 3 -> 1 wins  2 performed harakiri  0 performed harakiri | First: Attacker – 0, Target – 4. Difference = 4 (even) Attacker wins, Target lost. Second: Attacker – 1, Target – 3. Difference = 2 (even) Attacker wins, Target lost. Third: Attacker – 2, Target – 2. Equal, Attacker suicided. All other elements have lost. We remove them from the sequence. Sequence – 4 3. Length = 2. First: Attacker – 0, Target – 4 (% 2) = 0. Equal, Attacker suicided.  There is 1 element left, we stop the program. |
| 25 31 6 9 2 4 7 | 0 x 4 -> 0 wins  1 x 3 -> 1 wins  2 x 6 -> 2 wins  5 x 4 -> 4 wins  0 x 1 -> 1 wins  1 performed harakiri |  |