# Problem 3 – First Album

Orgolt’s Final Revenge are finally ready to record their first album. The problem is that they have more songs than the 4 they have the funds to record. So you should browse through the messy diary of the band members and find the first 4 songs or less that are valid.

You will be **receiving input lines** which may contain **any ASCII character**. Your task is to find **the songs that will be included in the debut album of the band**.

A **Valid Song** consists of three elements:

* Name
* Lyrics
* Length

The **Name** may consist of **any word character, whitespace characters and dashes('-'), but should neither start nor end with whitespace character or dash** ('**-**'). The song name should be between square brackets(**[ ])**.

The **Lyrics** are between double quotes (") and may contain **any word character and whitespace characters.**

The **Length** is valid when found in one of the following formats:

* In minutes – {minutes}:{seconds}m -> example: 0:44m; 10:05m;
* In seconds- {seconds}s -> example: 35s; 123s;

For a song to be valid, all three elements of a song should be present, but **their order may vary.**

Your program must end either when **4 valid songs are found** or when you receive the command "Rock on!". You must print **all valid songs** you've found, each on a **new line in the following format:**

{name} -> {length} => {lyrics}

## Input

* The input will come in several input lines, which may contain any ASCII character.
* The input ends when you receive the command "Rock on!".

## Output

* As output you must print the valid songs you’ve found, each on a new line.
* The format is: "**{song} -> {length} => {lyrics}"**
* The length should consist of two digits for the minutes and two for the seconds, separated by ':' -> **"{minutes}:{seconds}".**

## Constraints

* The input may contain **any ASCII character**.
* A song name and lyrics contain minimum **3 characters**.
* A song length will be between 1 second and 1 hour;
* There will be **no more** than **100 input lines**.
* Allowed working **time** / **memory**: **100ms** / **16MB**.

## Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| "Love me or hate me"jh#kefjhs#2fass$jfaahiu[Love and Hate]222snfaiufaui213544b  [Faith restored]"Orgolts Final Revenge"faniufsa8218217@2a0o99sRock'n'Roll  Guns'n'Tulips[SweetChildO'Nine]1:22maffsa########  22:14mfafa@@##@#@%^&\*\*\*\*\*\*[Sweet girl]"OFR"asffsasafsfa  Rock on! | Love and Hate -> 03:42 => Love me or hate me  Faith restored -> 01:39 => Orgolts Final Revenge  Sweet girl -> 22:14 => OFR |
| [SoftUniSong]"Flower Jokes are cool"student@softuni.bg66shttps://judge.softuni.bg/Contests/988  [ PESHO ]"IS GOSH@"200s  Rock on! | SoftUniSong -> 01:06 => Flower Jokes are cool |