# 04. Agents

After the agents had joined the agency, it is time to get their missions. Your task is to write a program that store every agent, missions that he has assigned for and rating for every mission. First you will receive receive information about the agents, missions and rating until you get the word **"registration".** This is called "the registration phase" and it may contains two type of information – **mission (String) with rating (floating point number)** or **agent name (String).**

* If it starts with **"#",** it's a mission. Missions are in the following format: **"#{missionName}:{missionRating}".**
* If **last 3 characters** starts with **"0",** then it's an agent.

When the registration phase is over, the operation phase starts. In this phase you will get commands that you have to implement to your program. Format and commands are shown below:

**{command}->{commandArguments}**, until the word **"operate". CommandArguments** will vary, depend on command. Commands that you can receive in this phase are:

* If command is **"assign" – line format will be: assign->{agentName}->{missionName}.** Your task is to assign a mission with its rating to a given agent. **(One agent cannot assign a mission twice, but one mission can be assigned to many agents.) ONLY REGISTERED MISSION CAN BE ASSIGNED.**
* If command is **"abort" – line format will be: abort->{missionName}.** Remove a mission from every agent that is assigned to it **(one or many).**
* If command is **"change" – line format will be: change->{agentName}->{agentName}.** Swap the missions with their ratings, of the two given agents **(they are always be valid)**.

**After all the commands are executed, if some agent has no missions, he has to be deleted.**

At the end you have to print the info for the agents with the **missions,** in **descending order by their total missions rating** in following format: **"Agent: {agentName} - Total Rating: {totalRating}"**, and then for each agent print their missions and ratings in **descending order by mission rating** in format : **"- {missionName} -> {missionRating}"** . If you have equal sum of ratings, **keep the order of the assignment**. See the examples.

## Input

* Strings in format format: **"#{missionName}:{missionRating}",** or "**{agentName}"** until the **"registration"** command.
* Strings in format **"{command}->{commandArguments}** **",** until the "**operate"** command.

## Output

* Print all agents and their missions ordered as mentioned above in format:

Agent: {agentName} - Total Rating: {totalRating}

- {missionName} -> {missionRating}

**…**

## Constraints

* the strings may contain any ASCII character except from **(:, -, >)**
* the floating point numbers will be in range **[0 - 10000]**

## Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| #BombField:10.45  Alpha005  Delta008  #Hostages:9.65  #Spying:8.35  Zulu011  registration  assign->Delta008->#BombField  assign->Alpha005->#Hostages  assign->Zulu011->#Spying  operate | Agent: Delta008 - Total Rating: 10.45  - #BombField -> 10.45  Agent: Alpha005 - Total Rating: 9.65  - #Hostages -> 9.65  Agent: Zulu011 - Total Rating: 8.35  - #Spying -> 8.35 |
| #BombField:10.45  Alpha005  Delta008  #Hostages:9.65  #Spying:8.35  Zulu011  registration  assign->Alpha005->#Hostages  assign->Zulu011->#Spying  assign->Delta008->#BombField  abort->#Spying  assign->Delta008->#Spying  assign->Alpha005->#Spying  operate | Agent: Delta008 - Total Rating: 18.80  - #BombField -> 10.45  - #Spying -> 8.35  Agent: Alpha005 - Total Rating: 18.00  - #Hostages -> 9.65  - #Spying -> 8.35 |
| #Guarding:13.54  #Undercover:18.85  #Spying:8.35  #BombField:10.45  Bond007  K001  registration  assign->Bond007->#Undercover  assign->Bond007->#BombField  assign->K001->#Guarding  assign->K001->#Spying  abort->#Guarding  change->Bond007->K001  operate | Agent: K001 - Total Rating: 29.30  - #Undercover -> 18.85  - #BombField -> 10.45  Agent: Bond007 - Total Rating: 8.35  - #Spying -> 8.35 |