## Problem 1 – Cafeteria

*You are the owner of a cozy cafe in a charming little town. Your team consists of baristas, and you need to manage their work schedules and customer orders. Each barista has a name, shift hours, and coffee-making skills..*

On the first line of the standard input, you will receive an integer n – the number of baristas on your team. On the next n lines, the baristas' details will follow with their names, shift, and a comma-separated list of coffee types they can prepare in the following format:

"{barista name} {shift} {coffee type 1,coffee type 2,...}"

Shift represent baristas shifts(it can be day or night), and the list of coffee types indicates the beverages they can make.

After you have formed your team, you will receive different commands, each on a new line, separated by " / ", until the "Closed" command is given. There are three actions that the baristas can perform:

"Prepare / {barista name} / {shift} / {coffee type}"

* If the barista is available during the shift and has the necessary skills, they can prepare the coffee order. Print this message:

"{barista name} has prepared a {coffee type} for you!"

* If the barista is not available during their shift or cannot make the requested coffee type, **print**:

"{barista name} is not available to prepare a {coffee type}."

"Change Shift / {barista name} / {new shift}"

* The barista updates their shift for either the day or night shift.
* **Print** the following message:

"{barista name} has updated his shift to: {new shift}"

"Learn / {barista name} / {new coffee type}"

* If the barista can prepare that coffee type, **print**:

"{barista name} knows how to make {new coffee type}."

* Else the barista adds a new coffee type to their list of beverages they can prepare.
* **Print** the following message:

"{barista name} has learned a new coffee type: {new coffee type}."

### Input

* On the first line of the standard input, you will receive an integer **n**
* On the following n lines, the baristas themselves will follow with their shift, and the list of coffee types they can prepare, separated by a space and comma-separated, as specified.
* You will be receiving different **commands**, each on a new line, separated by " / ", until the "Closed" command is given

### Output

Every order should print its own template sentence. After that, print all baristas, along with their updated shifts, and the list of coffee types they can prepare, in the following format:

**"Barista: {barista name}, Shift: {shift}, Drinks: {drink type 1, drink type 2, ...}"**

### Constraints

* The **names** of the baristas will **always** be **unique**.
* All given **commands** will be **valid**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| [  '3',  'Alice day Espresso,Cappuccino',  'Bob night Latte,Mocha',  'Carol day Americano,Mocha',  'Prepare / Alice / day / Espresso',  'Change Shift / Bob / night',  'Learn / Carol / Latte',  'Learn / Bob / Latte',  'Prepare / Bob / night / Latte',  'Closed']; | Alice has prepared a Espresso for you!  Bob has updated his shift to: night  Carol has learned a new coffee type: Latte.  Bob knows how to make Latte.  Bob has prepared a Latte for you!  Barista: Alice, Shift: day, Drinks: Espresso, Cappuccino  Barista: Bob, Shift: night, Drinks: Latte, Mocha  Barista: Carol, Shift: day, Drinks: Americano, Mocha, Latte |
| **Input** | **Output** |
| ['4',  'Alice day Espresso,Cappuccino',  'Bob night Latte,Mocha',  'Carol day Americano,Mocha',  'David night Espresso',  'Prepare / Alice / day / Espresso',  'Change Shift / Bob / day',  'Learn / Carol / Latte',  'Prepare / Bob / night / Latte',  'Learn / David / Cappuccino',  'Prepare / Carol / day / Cappuccino',  'Change Shift / Alice / night',  'Learn / Bob / Mocha',  'Prepare / David / night / Espresso',  'Closed']; | Alice has prepared a Espresso for you!  Bob has updated his shift to: day  Carol has learned a new coffee type: Latte.  Bob is not available to prepare a Latte.  David has learned a new coffee type: Cappuccino.  Carol is not available to prepare a Cappuccino.  Alice has updated his shift to: night  Bob knows how to make Mocha.  David has prepared a Espresso for you!  Barista: Alice, Shift: night, Drinks: Espresso, Cappuccino  Barista: Bob, Shift: day, Drinks: Latte, Mocha  Barista: Carol, Shift: day, Drinks: Americano, Mocha, Latte  Barista: David, Shift: night, Drinks: Espresso, Cappuccino |

function main(input) {

    let n = Number(input.shift())

    let baristas = {}

    for (let index = 0; index < n; index++) {

        let [name, shift, drinks] = input.shift().split(' ')

        drinks = drinks.split(',')

        baristas[name] = { shift, drinks }

    }

    function prepare(order) {

        let [\_, baristaName, shift, coffeeType] = order.split(' / ')

        if (baristas[baristaName].shift === shift && baristas[baristaName].drinks.includes(coffeeType)) {

            console.log(`${baristaName} has prepared a ${coffeeType} for you!`)

        } else {

            console.log(`${baristaName} is not available to prepare a ${coffeeType}.`)

        }

    }

    function changeShift(command) {

        let [\_, baristaName, newShift] = command.split(' / ')

        baristas[baristaName].shift = newShift

        console.log(`${baristaName} has updated his shift to: ${newShift}`)

    }

    function learn(command) {

        let [\_, baristaName, newCoffeeType] = command.split(' / ')

        if (baristas[baristaName].drinks.includes(newCoffeeType)) {

            console.log(`${baristaName} knows how to make ${newCoffeeType}.`)

        } else {

            baristas[baristaName].drinks.push(newCoffeeType);

            console.log(`${baristaName} has learned a new coffee type: ${newCoffeeType}.`)

        }

    }

    for (let command of input) {

        if (command === 'Closed') break

        if (command.startsWith('Prepare')) {

            prepare(command)

        } else if (command.startsWith('Change Shift')) {

            changeShift(command)

        } else if (command.startsWith('Learn')) {

            learn(command)

        }

    }

    for (let [name, data] of Object.entries(baristas)) {

        console.log(`Barista: ${name}, Shift: ${data.shift}, Drinks: ${data.drinks.join(', ')}`);

    }

}