# Coffee Machine

*You are hired to create a program that is processing orders from a coffee machine. Your will be receiving some coffee orders and your job is to calculate whether the money dropped in the machine is enough to make the order and print the corresponding output depending on that.*

The input will come as **array of strings**. Each string represents one order. It will contain information about the order. The pieces of **information** will be separated by a **', '**. The **first information** will be the **coins inserted** in the machine; the **second** will be the **type of the drink**, the **third** (**optional**; only for **coffee**) **'caffeine'** or **'decaf'**, the **fourth** (also **optional**) will be **'milk'** if the person wants his drink with milk, and the **last** will be the **quantity of sugar** the person ordered.

The **types of drinks** are described in the table below

|  |  |
| --- | --- |
| **Type** | **Price** |
| coffee caffeine | 0.80 |
| coffee decaf | 0.90 |
| cappuccino | 1.30 |
| tea | 0.80 |

**Notes:** for the coffee, **caffeine** and **decaf** come as **separated information**; the **drinks** that can be ordered **with milk** are the **tea** and **coffee** (there is **NOT** going to be order **cappuccino** with **additional** **milk**); **all** of the drinks will have **sugar quantity between 0 and 5**  
Examples for coffee orders:  
 - '1.10, coffee, decaf, milk, 4'  
 - '0.80, coffee, caffeine, 2'  
In the first example the **coffee** is **decaf** with **milk** and quantity of **sugar 4**; in the second the coffee is with **caffeine**, **no milk** and quantity of **sugar 2**.

Prices for additions:  
- milk – **costs** **10% of the price so far rounded to the next higher number. Examples: 0.03 -> 0.1; 0.12 -> 0.2; 1.05 -> 1.10; 0.005 -> 0.01**- sugar – **no matter the quantity (except from 0) it costs 0.10. Add the sugar at the end!**

After you finish calculating the price for the order, there are **two possible outputs**:  
- The money is enough: **'You ordered {drink}. Price: {price}. Change: {change}'**- The money is not enough: **'Not enough money for {drink}. Need {money needed} more'**All of the **numbers should be formatted to the second decimal point**

At the end of the day, you need to print the **total money earned** from the **successful** orders in the format: **'Income Report: {total money}'**; **format** the money to the **second decimal point**

### Input

* The input comes as **array of strings**
* Each string is order you need to process

### Constrains

* The input will always be **valid**

### Output

* For each order there are **two possible** outputs (described above)
* At the end print the **total income** as described above

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| ['1.00, coffee, caffeine, milk, 4',  '0.50, cappuccino, 1',  '0.40, tea, milk, 2'] | You ordered coffee. Price: 1.00 Change: 0.00  Not enough money for cappuccino. Need 0.90 more  Not enough money for tea. Need 0.60 more  Income Report: 1.00 |
| **Comments** | |
| The first order is coffee with caffeine, milk and sugar. (0.80 + 0.10 + 0.10 = 1.00)  The second order is cappuccino with sugar (1.30 + 0.10 = 1.40)  The third order is tea with milk and sugar (0.80 + 0.10 + 0.10 = 1.00)  Total income = 1.00 | |