## 2. Python: Vending Machine

Implement class: *VendingMachine* according to the following requirements:

- can be instantiated using the constructor *VendingMachine*(num_items, item_price) where num_items denotes the number of items in the machine, and item_price denotes the required number of coins to buy a single item.
- has a method *buy*(req_items, money) that represents a buy request where *req_items* denotes the requested number of items, and *money* is the amount the customer puts into the machine. Depending on the state of the machine, one of the following happens
  - If there are enough items in the machine to serve the request and the given money is sufficient to buy the requested number of items, the number of items in the machine is reduced by the requested number of items. The method returns an integer denotes the change given back after the purchase.
  - If there are fewer items in the machine than the requested number, it raises a *ValueError* exception with the message "Not enough items in the machine".
  - If there are enough items in the machine to serve the request but the given amount of money is less than their cost, it raises a *ValueError* exception with the message "Not enough coins".

The class implementation will be tested by a provided code stub and several input files. Each input file contains parameters to test the implementation. First, the provided code stub initializes an instance of the *VendingMachine*. Next, it performs the given operations on the *VendingMachine* instance. The result of their execution will be printed to the standard output by the provided code.

**Constraints**

- There will be at most 100 operations to be performed

**▼ Input Format Format for Custom Testing**

Input from stdin will be processed as follows and passed to the function.

In the first line, there are two space-separated integers *num_items* and *item_price* denoting the parameters to initialize the *VendingMachine*.
In the second line, there is an integer *n* denoting the number of operations to be performed on a *VendingMachine* instance.
Each of the following *n* lines contains two space-separated integers, *num_items* and *money*, that denote a single operation to be performed on the *VendingMachine*, along with its parameters if any.

**▼ Sample Case 0**

**Sample Input 0**

```
STDIN       Function
-----       --------
10 2   →    num_items = 10, item_price = 2
4      →    n = 4
1 5    →    req_items = 1, money = 5       (1st transaction)
10 100 →    req_items = 10, money = 100    (2nd transaction)
7 100  →    req_items = 7, money = 100     (3rd transaction)
2 3    →    req_items = 2, money = 3       (4th transaction)
```

**Sample Output 0**

```
3
Not enough items in the machine
86
Not enough coins
```

**Explanation 0**

The code initializes the *VendingMachine*: *machine* = VendingMachine*(10, 2)*, i.e. a machine with 10 items, each costing 2 coins. Then, there are 4 operations to be performed:

1. The method *buy(1, 5)* is called, i.e. a request of buying 1 item with 5 units of currency is performed. There are enough items in the machine to serve the request and the given money is sufficient to purchase them. The change, 5-1*2 = 3 in this case, is returned by the method, and the number of items in the machine is reduced by 1. There are 9 items in the machine after the request.
2. The method *buy(10, 100)* is called, i.e. a request of buying 10 items with 100 units of currency is performed. There are not enough items in the machine to serve the request , so a ValueError is raised with the message "Not enough items in the machine".
3. The method *buy(7, 100)* is called, i.e. a request of buying 10 items with 100 units of currency is performed. There are enough items in the machine to serve the request and the given money is sufficient to purchase them. The change, 100-7*2 = 86, is returned and the number of items in the machine is reduced by 7. Now there are *9 - 7 = 2* items in the machine.
4. The method *buy(2, 3)* is called, i.e. a request of buying 2 items with 3 units of currency is performed. There are enough items in the machine but the money is not enough. A ValueError is raised with the message "Not enough coins".

▼ **Sample Case 1**

**Sample Input 1**

```
100 1
2
50 50
50 60
```

**Sample Output 1**

```
0
10
```

**Explanation 1**

The code initializes the *VendingMachine*: *machine* = VendingMachine*(100, 1)*, i.e. a machine with 100 items, each costing 1 unit. There are 2 operations to be performed:

- The method *buy*(50, 50) is called, i.e. a request of buying 50 items with 50 units of currency is performed. There are enough items in the machine and the given money is sufficient to purchase them. Change = 50-50*1 = 0 in this case, which is returned by the method. The item count is reduced by 50 leaving 50 items in the machine.
- The method *buy*(50, 60) is called, i.e. a request of buying 50 items with 60 units of currency is performed. There are enough items in the machine to serve the request and the given money is sufficient, so the change, 60-50*1 = 10, is returned and the number of items is reduced by 50. Now there are 0 items in the machine.