Programming in Java
Prep Term 2018-19
Lab 2

An online store sells bakery and dairy products. However, the store itself does not produce these items. It has a number of suppliers - bakeries and dairies - it works with. When a customer places an order with the store, the store, in turn, passes on the request to one of these bakeries or dairies (depending on the item ordered) and they will in turn supply it to the customer.

The store wants to be fair to all its suppliers. So, as it receives requests from customers, it cycles through its suppliers and places the order on each of them in a *round robin* fashion: for example, for bakery products, the bakery that was created first gets the first order, the second bakery gets the second order and on so on, and then back to the first bakery. Similarly for the dairy products.

Once all the customer orders are processed, the store asks the suppliers to print out the list of orders they have received, in the reverse order in which they received the orders - last order is printed first and the first order is printed last. Each order is printed as 3 items - the order number, the item ordered, and the number of units ordered.

Simulate this as a Java program. The program has to address the following:

- The store can have any number of bakeries and dairies as suppliers.
- Bakeries and dairies are identified by an id in the order in which they were created. Thus bakeries would be called “Bakery 1”, Bakery 2” … and Dairies would be called “Dairy 1”, “Dairy 2”, ...
- The store lists 3 bakery items: bread, cakes, muffins and 2 dairy items - milk, butter
- Customers place their order with the store. An order consists of the name of one of the items (any of the 5 items listed above) and an integer which is the number of units of that item being ordered. Each order has a unique (sequential) “order number” (integer) assigned to it.
- The store sends the request to a bakery or dairy as appropriate, using the cyclic order described above
- Customers cannot directly place orders to the bakery/dairy. They can only access the services of the store. In fact, they are not even aware of the existence of the bakeries and dairies.
- When the store is instantiated, it provides a mechanism to specify the number of bakeries and dairies. It then creates these suppliers and keeps track of them.
- When the list of requests has been processed, the store cycles through each of its suppliers and asks them to print out their order list as described above.

Design the appropriate classes, and group them into packages if relevant.

The **main (**which should be in a **separate** class from the store**)** does the following:

- Creates a store.
- Reads the input file (in the format described below) to determine the number of bakeries and dairies and informs the store of the number of bakeries and dairies it should manage
- Continues reading the input file for the orders and send them to the store. The end of the input is indicated by the word “End” as the first item in an input line.
- Informs the store that the orders have been exhausted
- Note that the **main** also does not know of the existence of bakeries and dairies, and should interact only with the store. (What are the different design options to help enforce this?)

Input file format:

1. The first line contains 2 integers, which are the number of bakeries and dairies, respectively, managed by the store.
2. The next set of lines contain the name of the item (one of Bread, Cakes, Muffins, Milk, Butter) and an integer indicating the number of units of that item ordered
3. An “End” as the first item in a line indicated end of input

Sample input:

1 2
Bread 3
Milk 4
Butter 2
Cakes 3
Milk 3
End

Expected output for above:

Bakery 1
4 Cakes 3
1 Bread 3

Dairy 1
5 Milk 3
2 Milk 4

Dairy 2
3 Butter 2