Software Engineer Exercise

You are building a cash machine that, when given a banknote will release coins for an equivalent value.
The machine has the following coins available: 0.20£, 0.50£, 1£, 2£.
The machine accepts the following banknotes: 5£, 10£, 20£.

There are 2 operations available:

- an operator can load more coins in the machine
- a customer can exchange a banknote

You are given a text file which contains two types of instructions:

`> LOAD [number_of_coins] [type_of_coin]`

Load the given number of coins of the given type into the machine.
E.g. the command to load 50 coins of 0.20£ in the machine looks like:

```
> LOAD 50 0.20
```

`> EXCHANGE [banknote_amount]`

Takes in a banknote of the given amount and exchange it for an equivalent amount in coins.
E.g. the command to exchange a 20£ banknote is:

```
> EXCHANGE 20
```

The output to an exchange command is a number and value of coins to be exchanged, or a notification that the exchange is not possible if there are not enough coins available.
E.g. to exchange 20£ a *possible* result is:

```
< 5 2£, 10 1£
```

Or, if there are not enough coins to perform the exchange, the message would be:

```
< CANNOT EXCHANGE
```

Write an application that:

- reads the text file with commands
- will output each command received
- after each command will output the number of coins and banknote available in the cash machine. E.g.:

```
= 5 0.20£, 10 0.50£, 5 1£, 3 2£, 0 5£, 6 10£, 2 20£
```

For example, given the following input file (`input.txt`):

```
> LOAD 10 1
> LOAD 20 2
> EXCHANGE 20
> EXCHANGE 20
> EXCHANGE 20
> EXCHANGE 10
```

The command:

```
$ python cash_machine.py input.txt
```

Will output:

```
> LOAD 10 1
= 10 1£
> LOAD 20 2
= 10 1£, 20 2£
> EXCHANGE 20
< 10 1£, 5 2£
= 15 2£, 1 20£
> EXCHANGE 20
< 10 2£
= 5 2£, 2 20£
> EXCHANGE 20
< CANNOT EXCHANGE
= 5 2£, 2 20£
> EXCHANGE 10
< 5 2£
= 1 10£, 2 20£
```

The implementation must be in core python and include tests.