# Simple Database (https://thumbtack.com)

This task is create a very simple in-memory database, which has a very limited command set. All of the commands are going to be fed to you one line at a time via stdin, and your job is to process the commands and to perform whatever operation the command dictates.

## Data commands

Your database should accept the following commands:

SET [name] [value]:   Set a variable [name] to the value [value]. Neither variable names nor values will ever contain spaces.

GET [name]:           Print out the value stored under the variable [name]. Print NULL if that variable name hasn't been set.

UNSET [name]:         Unset the variable [name]

NUMEQUALTO [value]:   Return the number of variables equal to [value]. If no values are equal, this should output 0.

END:                  Exit the program

Note: [name] cannot begin with a number and [value] cannot be character or string,

## Examples

So here is a sample input:

```
SET a 10
GET a
UNSET a
GET a
END
```

And its corresponding output:

```
10
NULL
```

---------------------------------------------------------

And another one:

```
SET a 10
SET b 10
NUMEQUALTO 10
NUMEQUALTO 20
UNSET a
NUMEQUALTO 10
SET b 30
NUMEQUALTO 10
END
```

And its corresponding output:

```
2    # it means the number of names that have value 10 is 2
0    # it means the number of names that have value 20 is 0
1    # after UNSET a, so the number of names that have value 10 is 1
0    # after SET b 30, so the number of names that have value 10 is 0
```

---------------------------------------------------------

## Transaction commands

In addition to the data commands, your database should support transactions, accepting the following commands:

BEGIN:    Open a transactional block

ROLLBACK: Rollback all of the commands from the most recent transactional block.

COMMIT:   Permanently store all of the operations from all presently open transactional blocks.
Both ROLLBACK and COMMIT cause the program to print 'NO TRANSACTION' if there are no open transaction blocks.

Your database needs to support nested transactions. ROLLBACK only applies to the most recent transaction block, but COMMIT applies to all transaction blocks. (Any data command run outside of a transaction is committed immediately.) Examples

Here are some sample inputs and expected outputs using these commands:

## Examples

Input:

```
BEGIN
SET a 10
GET a
BEGIN
SET a 20
GET a
ROLLBACK
GET a
ROLLBACK
GET a
END
```

Output:

```
10
20
10
NULL
```

-------------------------------------------------------------------------------------

Input:

```
BEGIN
SET a 30
BEGIN
SET a 40
COMMIT
GET a
ROLLBACK
END
```

Output:

```
40
```

```
NO TRANSACTION
------------------------------------------------------------------------------------

Input:

SET a 50
BEGIN
GET a
SET a 60
BEGIN
UNSET a
GET a
ROLLBACK
GET a
COMMIT
GET a
END

Output:

50
NULL
60
60
------------------------------------------------------------------------------------

Input:

SET a 10
BEGIN
NUMEQUALTO 10
BEGIN
UNSET a
NUMEQUALTO 10
ROLLBACK
NUMEQUALTO 10
END

Output:

1
0
1
------------------------------------------------------------------------------------
```

I implement this simple database application by using Python and ply library.

How to execute the Simple Database?  Compatible with Python 2.xx or 3.xx version.

The command:

```
        $python database.py         #on Linux Platform
                          or
        C:\python database.py       #on Windows Platform
```