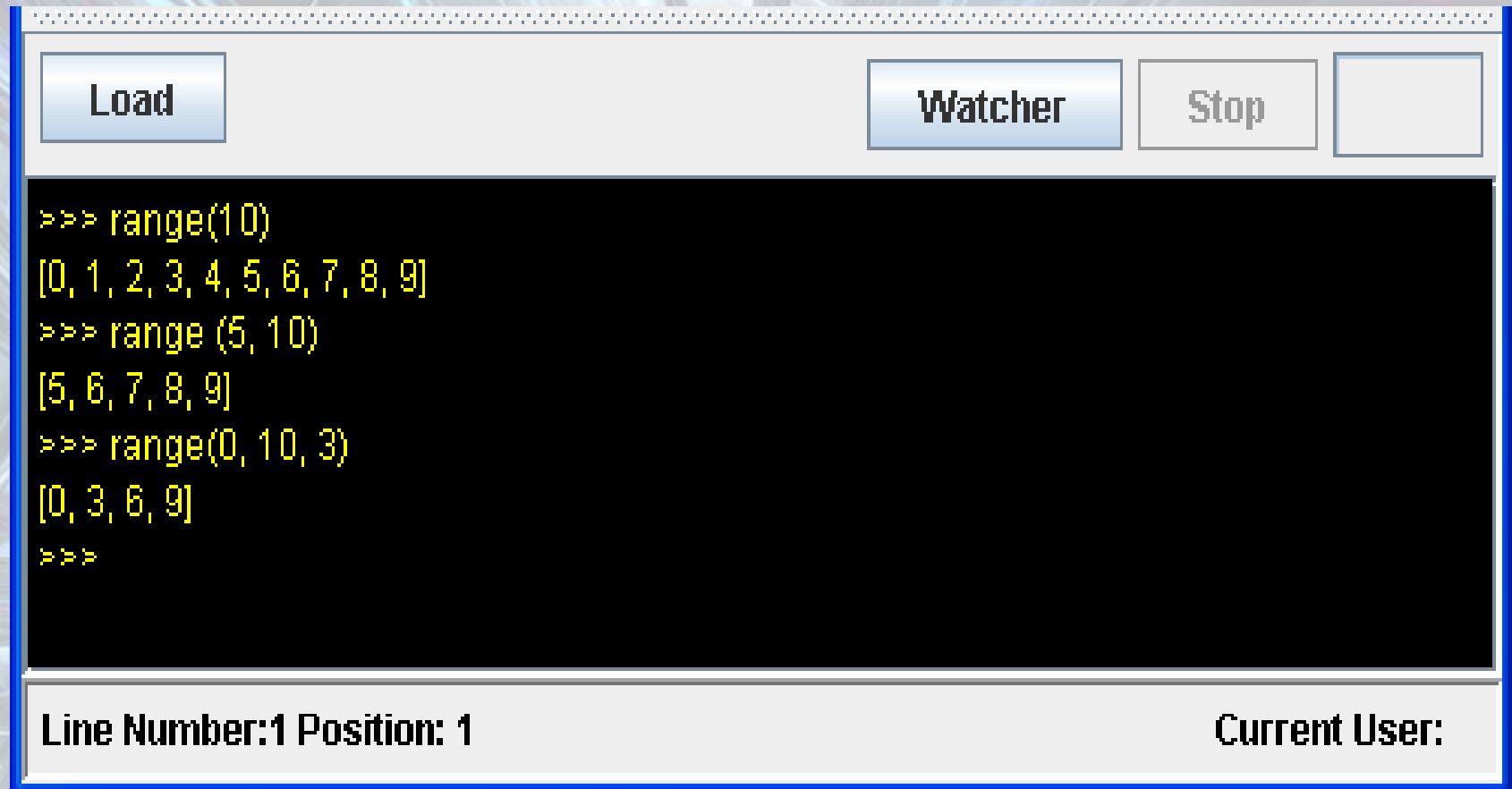


# **The Range Function, Conditionals, and Loops**

# The Range Function: Numbers

- `range(x)` – gives a list of the numbers from 0 to  $x-1$ :  $[0, \dots, x-1]$
- `range(x, y)` – gives a list of the numbers from  $x$  to  $y-1$ :  $[x, \dots, y-1]$
- `range(x, y, z)` – gives a list of the numbers that are multiples of  $z$  from  $x$  to  $y-1$

# The Range Function: Numbers - Examples



The image shows a screenshot of a Python IDLE window. At the top, there is a menu bar with 'Load', 'Watcher', 'Stop', and an empty button. Below the menu bar is a large black text area with yellow text showing three examples of the range function: range(10) returns [0, 1, 2, 3, 4, 5, 6, 7, 8, 9], range(5, 10) returns [5, 6, 7, 8, 9], and range(0, 10, 3) returns [0, 3, 6, 9]. The prompt '>>>' is shown at the end of the last line. At the bottom of the window, there is a status bar with 'Line Number:1 Position: 1' on the left and 'Current User:' on the right.

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> range(5, 10)
[5, 6, 7, 8, 9]
>>> range(0, 10, 3)
[0, 3, 6, 9]
>>>
```

Line Number:1 Position: 1

Current User:

# The Range Function: Strings

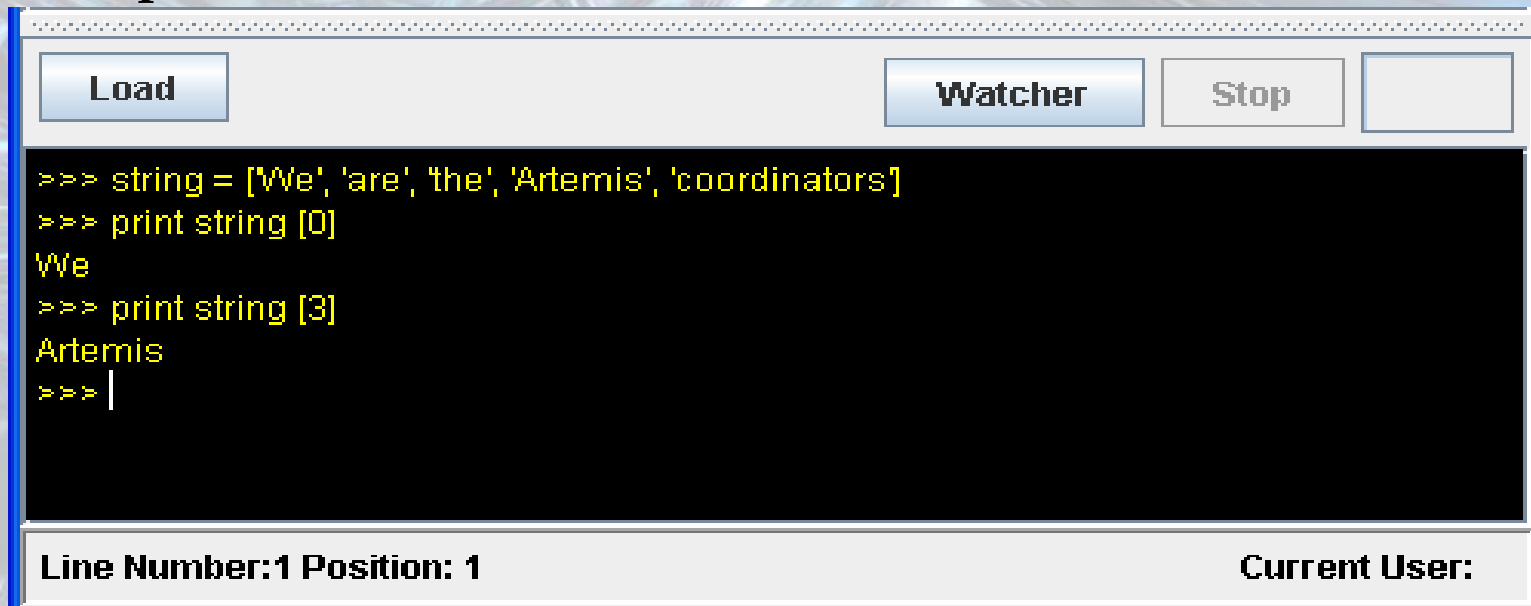
- You can store strings as a range as well, and use their indices to access them:

```
string = ['some, text']
```

```
string[0] = some
```

```
string[1] = text
```

Example:



The screenshot shows a Python IDE window with a dark background. At the top, there are four buttons: 'Load', 'Watcher', 'Stop', and an empty button. The main area contains a Python REPL session with the following code and output:

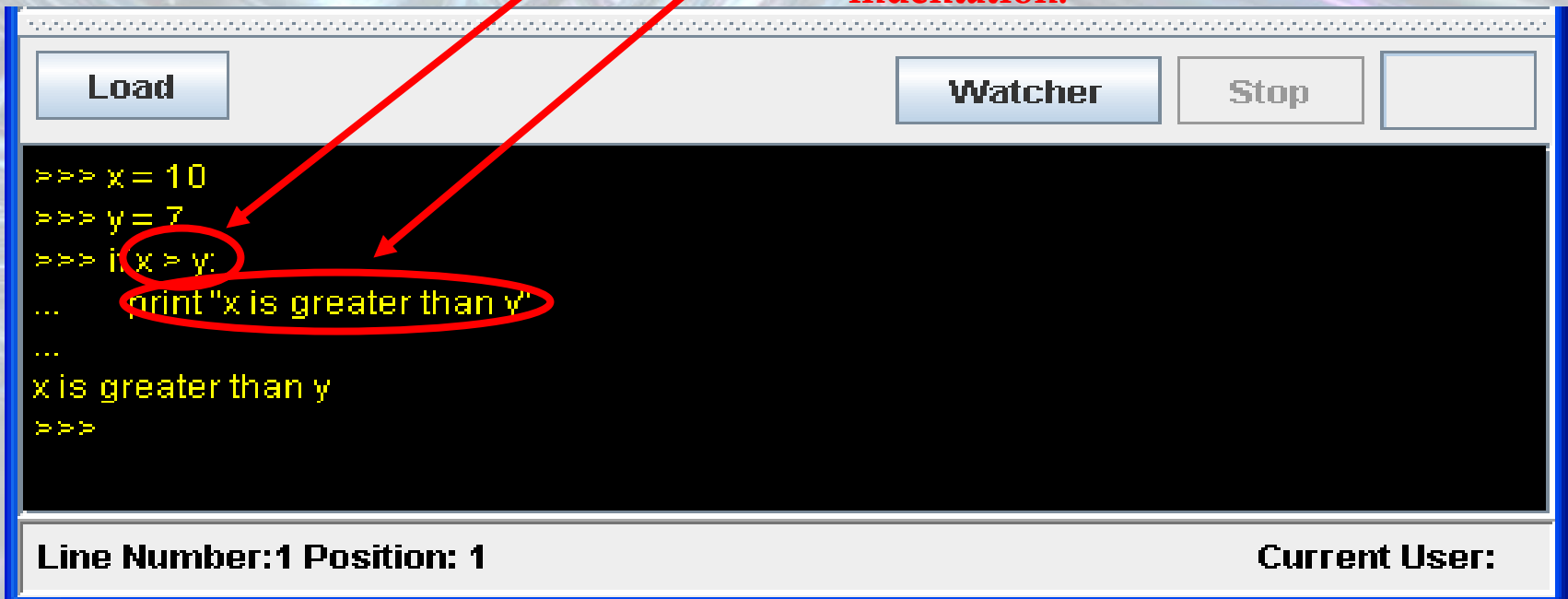
```
>>> string = ['We', 'are', 'the', 'Artemis', 'coordinators']
>>> print string [0]
We
>>> print string [3]
Artemis
>>> |
```

At the bottom of the window, there is a status bar with the text 'Line Number:1 Position: 1' on the left and 'Current User:' on the right.

# Conditionals:

## The “If” Statement

- If you want a certain block of code to only be executed if a certain condition is true, you can use an “if” statement!
  - An example:  
if variable x is greater than variable y  
then print that variable x is greater
- In code:
- Condition:** note that it is followed by a “:”
- Consequence:** note that it is in a “block”. Press the tab button for each consequence for indentation.



```
>>> x = 10
>>> y = 7
>>> if x > y:
...     print "x is greater than y"
...
x is greater than y
>>>
```

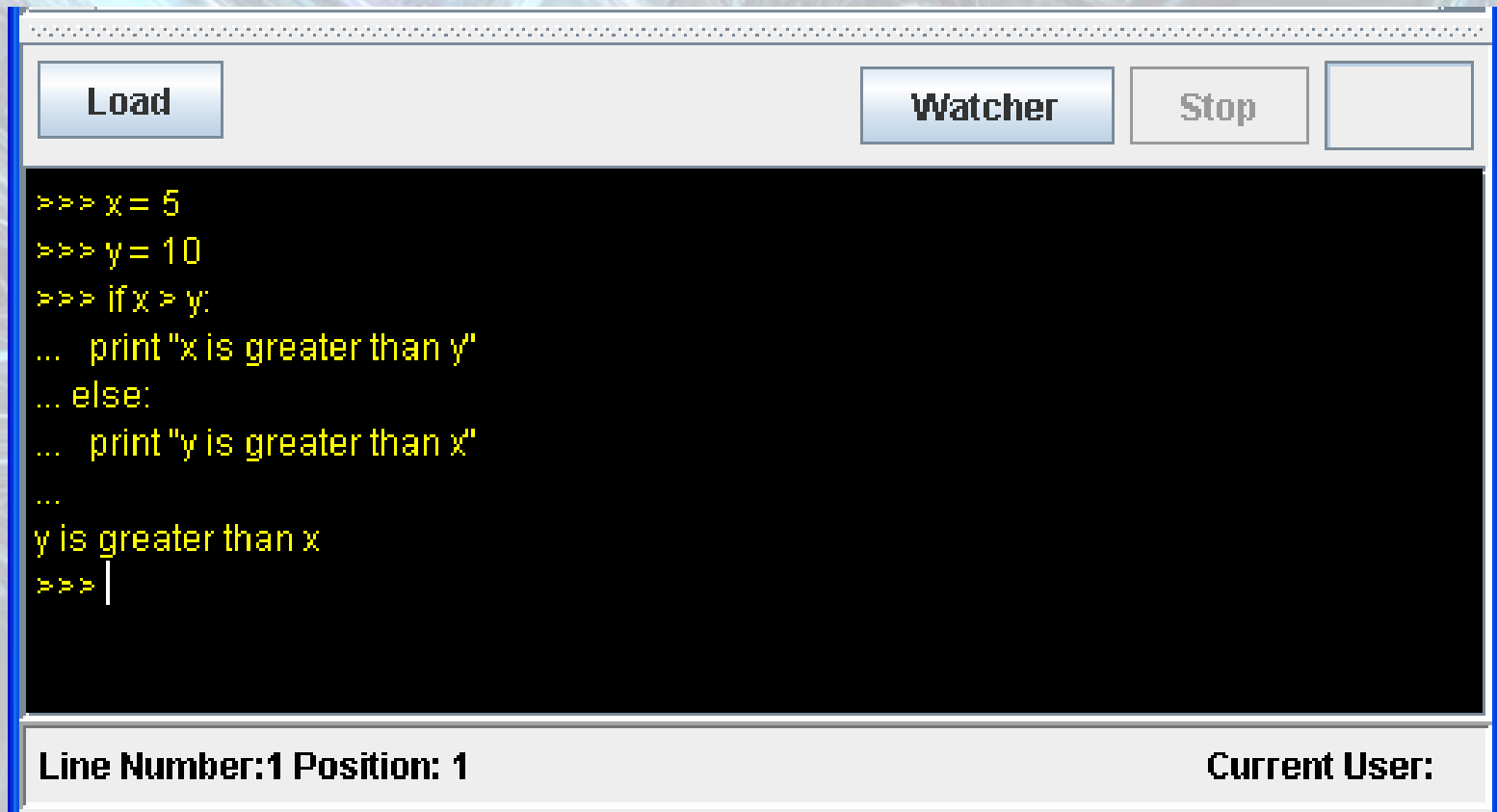
Line Number:1 Position: 1

Current User:

# Conditionals:

## The “If...Else” Statement

- If a certain condition produces a certain result, what happens when that condition is not true? **Note: the indentation – one tab for each consequence**
- You can use an “if...else” statement!



The screenshot shows a Python IDE window with a dark background. At the top, there are four buttons: "Load", "Watcher", "Stop", and an empty button. The main area contains a Python script with the following code:

```
>>> x = 5
>>> y = 10
>>> if x > y:
...     print "x is greater than y"
... else:
...     print "y is greater than x"
...
y is greater than x
>>> |
```

At the bottom of the window, there is a status bar with two fields: "Line Number:1 Position: 1" and "Current User:".

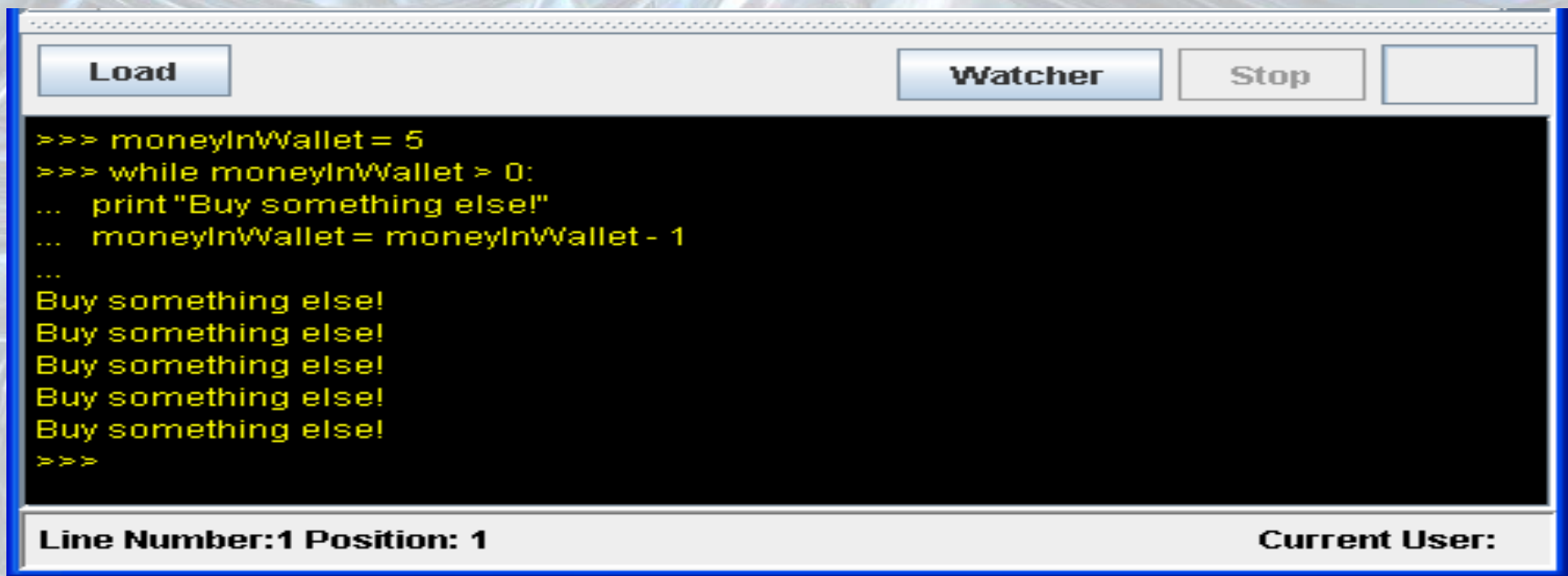
# Loops: The “While” Loop

- If you want some code to execute only while a certain condition is true (and the condition is changing), use a “while” loop!
- Example: Let’s say you’re shopping at a dollar store and you have 5 dollars in your wallet and you want to spend it all:

while I have money in my wallet

print that I must buy something else

decrease the amount of money in my wallet



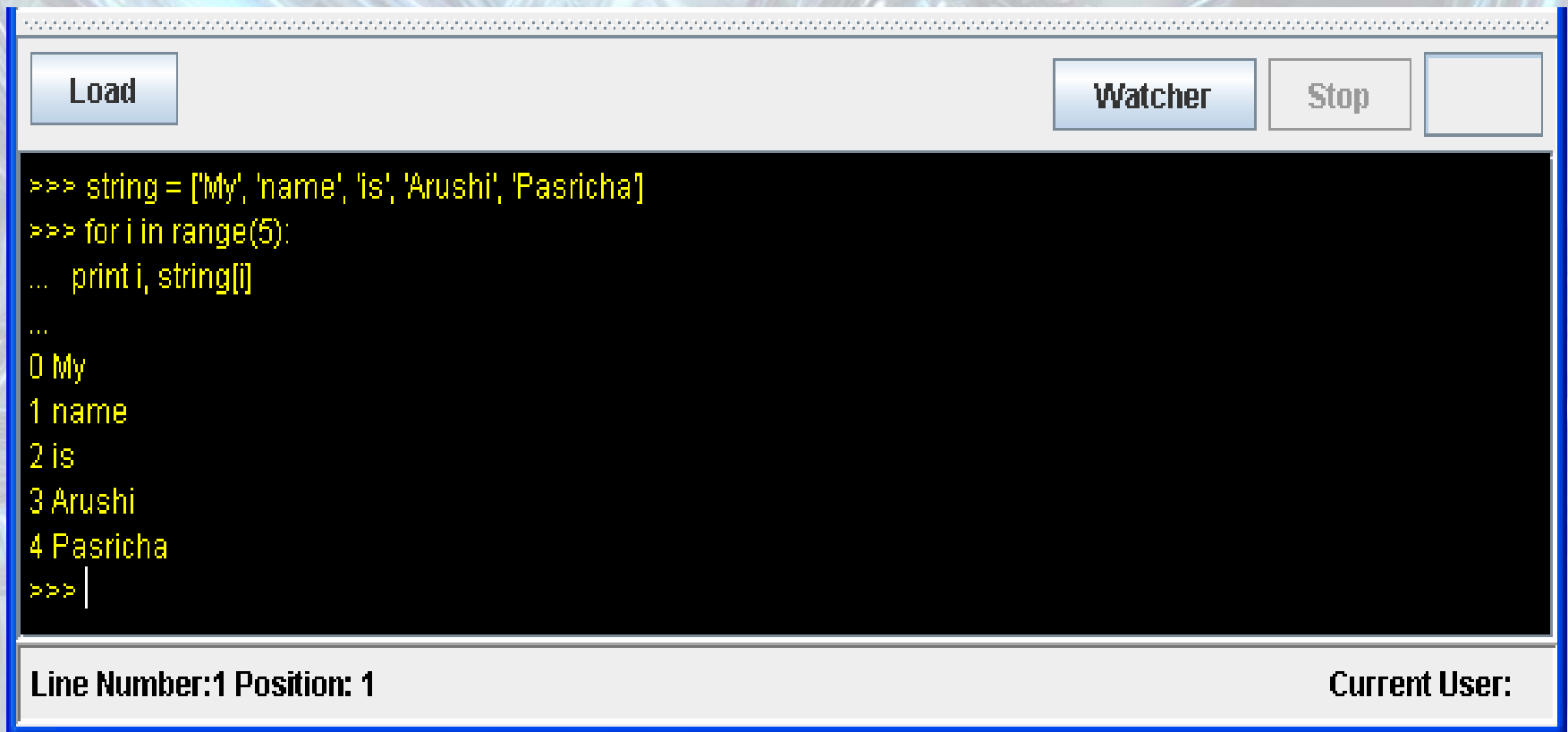
The screenshot shows a Python interpreter window with a title bar. It contains four buttons: 'Load', 'Watcher', 'Stop', and an empty button. The main area is a black console with yellow text showing the execution of a while loop. The loop prints 'Buy something else!' and decrements the 'moneyInWallet' variable from 5 to 0. The status bar at the bottom shows 'Line Number:1 Position: 1' and 'Current User:'.

```
>>> moneyInWallet = 5
>>> while moneyInWallet > 0:
...   print "Buy something else!"
...   moneyInWallet = moneyInWallet - 1
...
Buy something else!
Buy something else!
Buy something else!
Buy something else!
Buy something else!
>>>
```

Line Number:1 Position: 1 Current User:

# Loops: The “For” Loop

- Another type of loop is a “for” loop.
- You can use this kind of loop with a range function:



The screenshot shows a Python IDE window with a toolbar at the top containing buttons for 'Load', 'Watcher', 'Stop', and an empty button. The main text area has a black background with yellow text. It displays a Python script that defines a list 'string' and uses a 'for' loop to iterate over its elements, printing the index and the corresponding element. The output of the loop is visible below the code. At the bottom of the window, a status bar shows 'Line Number:1 Position: 1' on the left and 'Current User:' on the right.

```
>>> string = ['My', 'name', 'is', 'Arushi', 'Pasricha']
>>> for i in range(5):
...     print i, string[i]
...
0 My
1 name
2 is
3 Arushi
4 Pasricha
>>> |
```

Line Number:1 Position: 1

Current User:

