

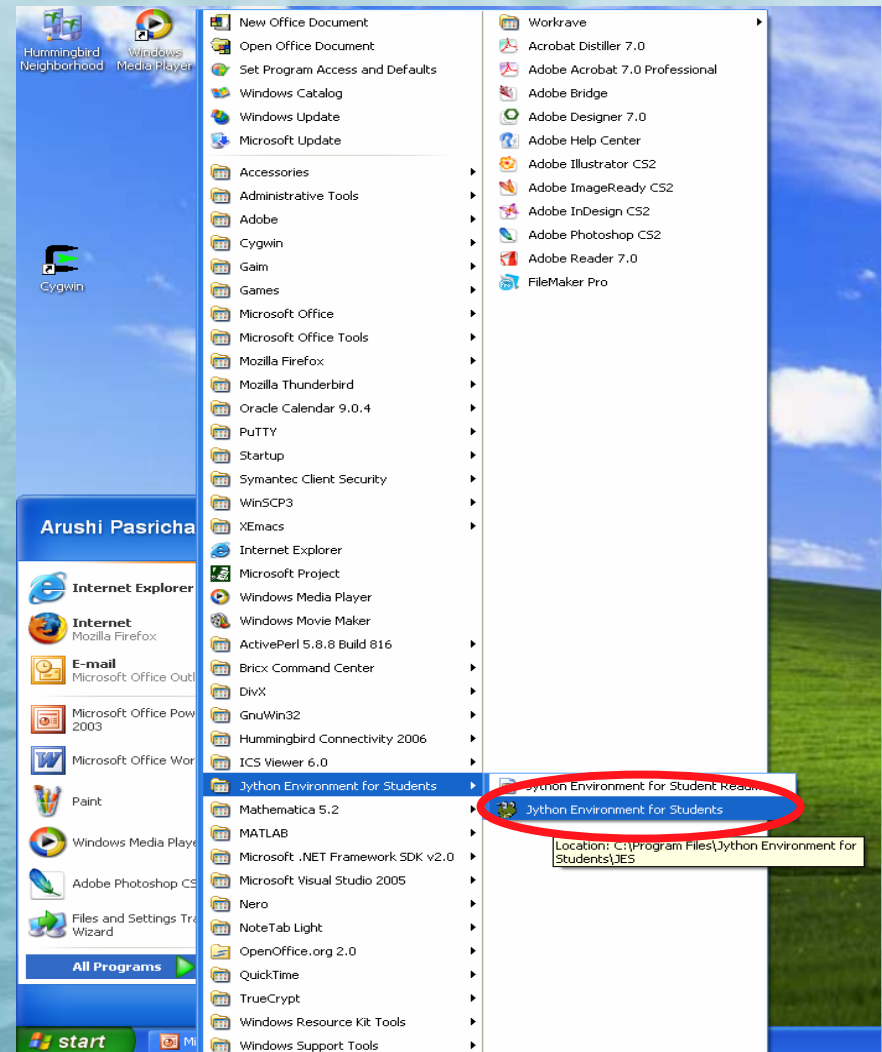
Introduction to Python



JES

(Jython Environment for Students)

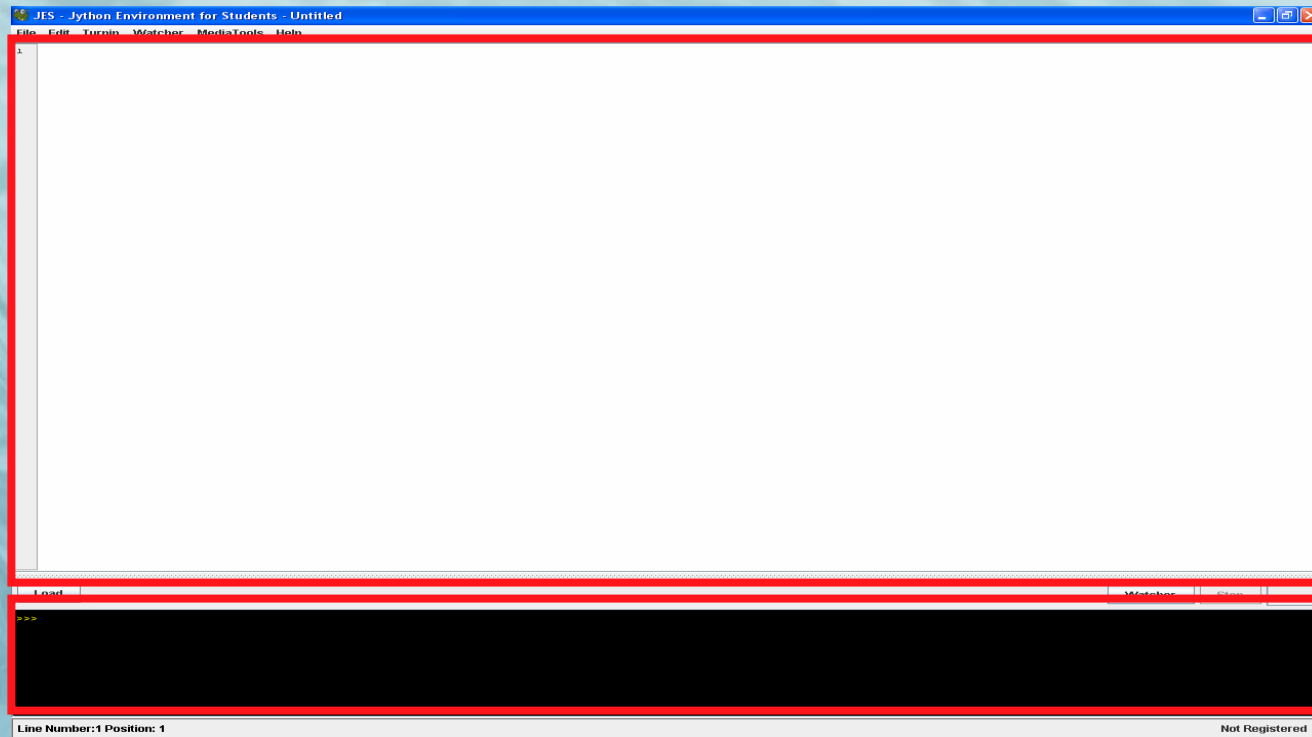
- JES is the program we'll be using to write our Python code
- Opening JES:
Click on Start >> All Programs >> Jython Environment for Students >> Jython Environment for Students



JES Explained

JES has two components to its interface.

The top white screen is where you define methods or classes for your program.

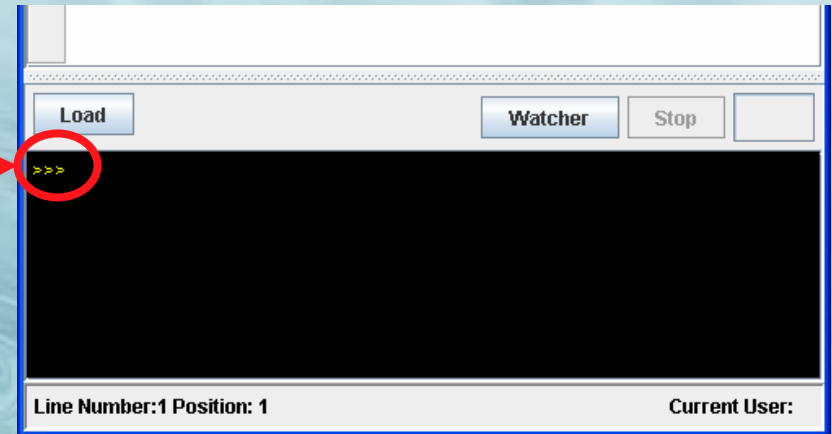


The bottom black screen is where you implement code, but you can also write snippets of code here as well! The main difference is that you get an immediate response. We'll be doing most of our work with this screen first.

Numbers

First, let's look at how we can print numbers into the shell.

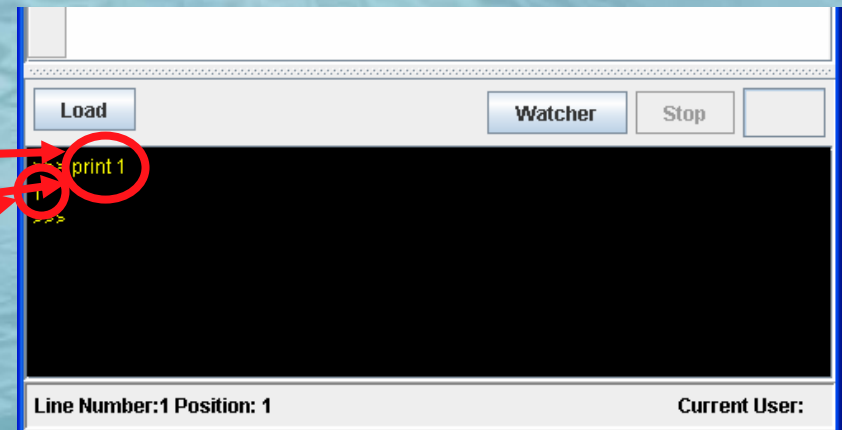
This is your prompt



To print a number, let's say 1, just write at the prompt

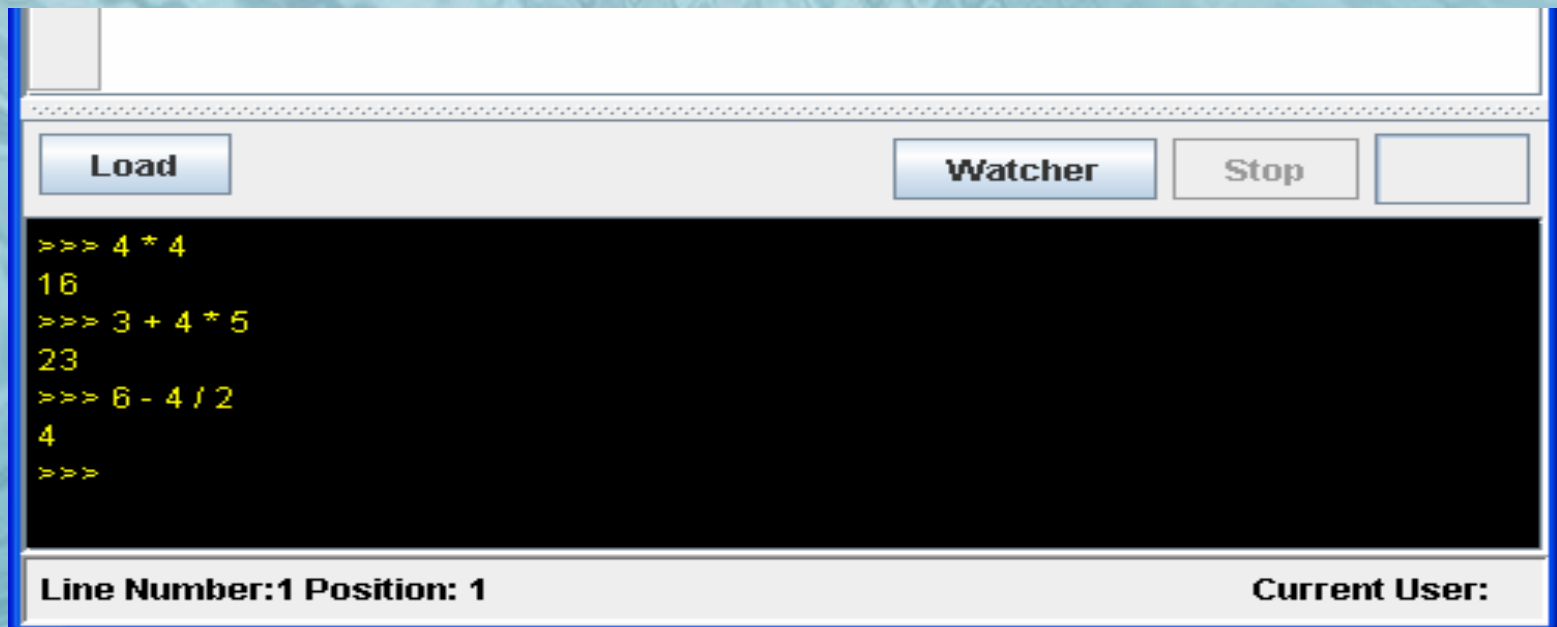
INPUT

OUTPUT



Simple Math

- Python is great at calculating!
- Just type any simple calculation at the prompt, and it will evaluate it for you
- Examples:



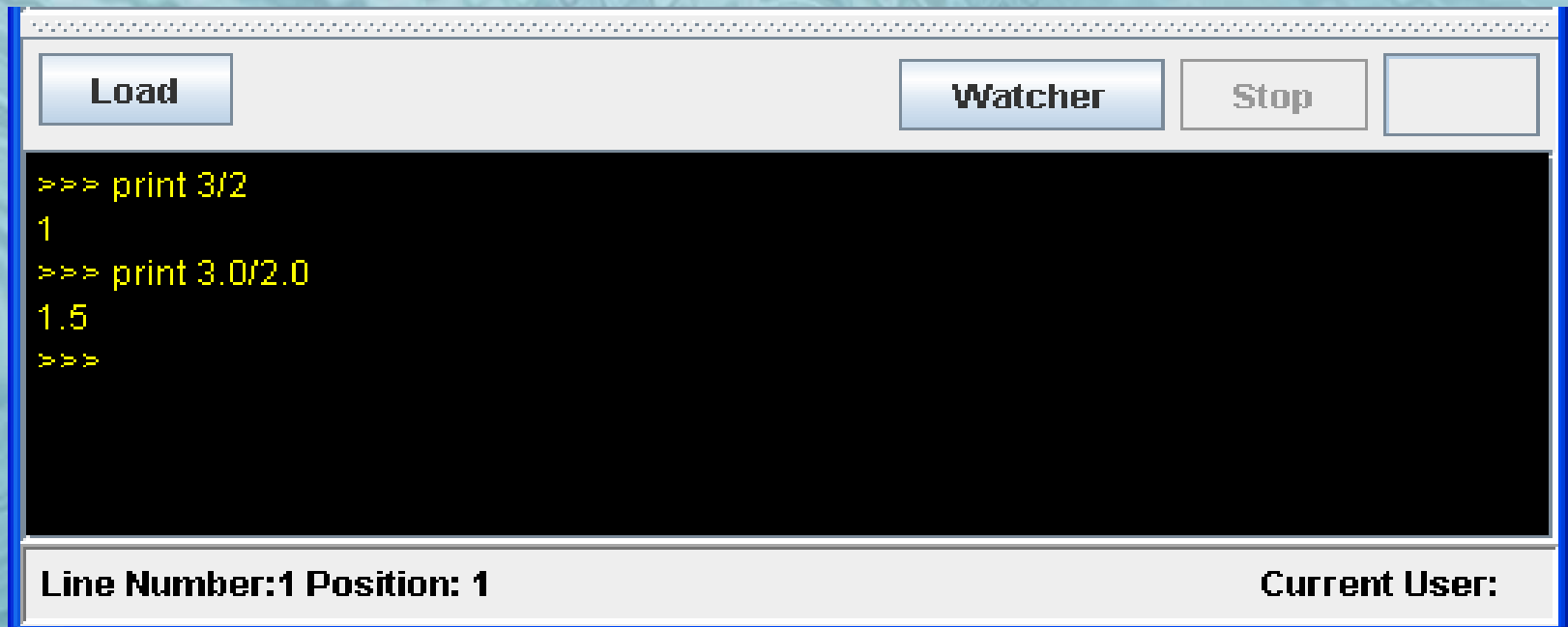
A screenshot of a Python interpreter window. The window has a title bar at the top. Below the title bar is a toolbar with four buttons: "Load", "Watcher", "Stop", and an empty button. The main area of the window is a black console with yellow text showing the following interactions:

```
>>> 4 * 4
16
>>> 3 + 4 * 5
23
>>> 6 - 4 / 2
4
>>>
```

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

Floats and Integers

- The main difference: a float has a decimal point and an integer doesn't
- When doing simple math in JES, the number of decimal points you include in your mathematical expression will be the number of decimal points the given answer will be cut off at:



The screenshot shows the JES interface with a top bar containing 'Load', 'Watcher', 'Stop', and an empty button. The main area is a black code editor with yellow text. It shows two print statements: 'print 3/2' which outputs '1', and 'print 3.0/2.0' which outputs '1.5'. The bottom status bar shows 'Line Number:1 Position: 1' and 'Current User:'.

```
>>> print 3/2
1
>>> print 3.0/2.0
1.5
>>>
```

Line Number:1 Position: 1 Current User:

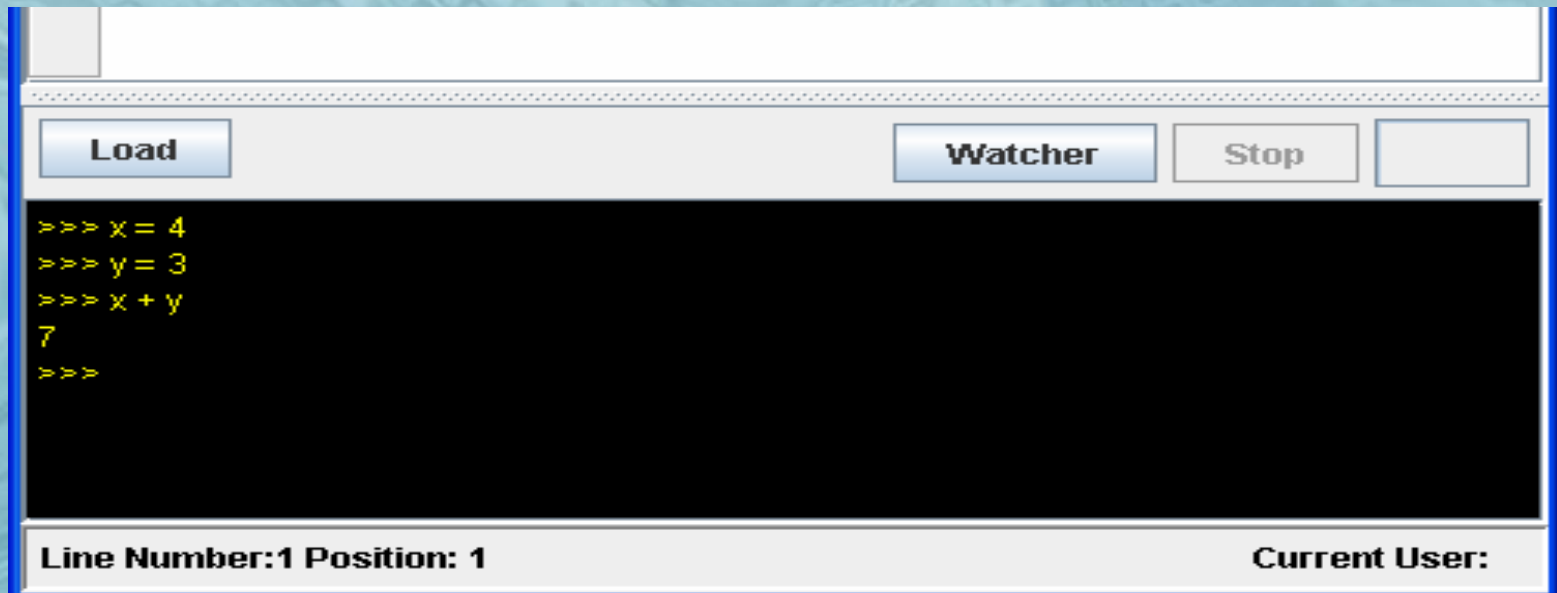
Defining & Using Variables

We can define variables by simply doing:

>>> x = 4 you can use any letter or name for your variable

>>> y = 3

Then you can do:



The screenshot shows a Python interpreter window with a white title bar and a blue border. The window contains a black text area with yellow text. Above the text area is a toolbar with four buttons: 'Load', 'Watcher', 'Stop', and an empty button. The text area contains the following code:

```
>>> x = 4
>>> y = 3
>>> x + y
7
>>>
```

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

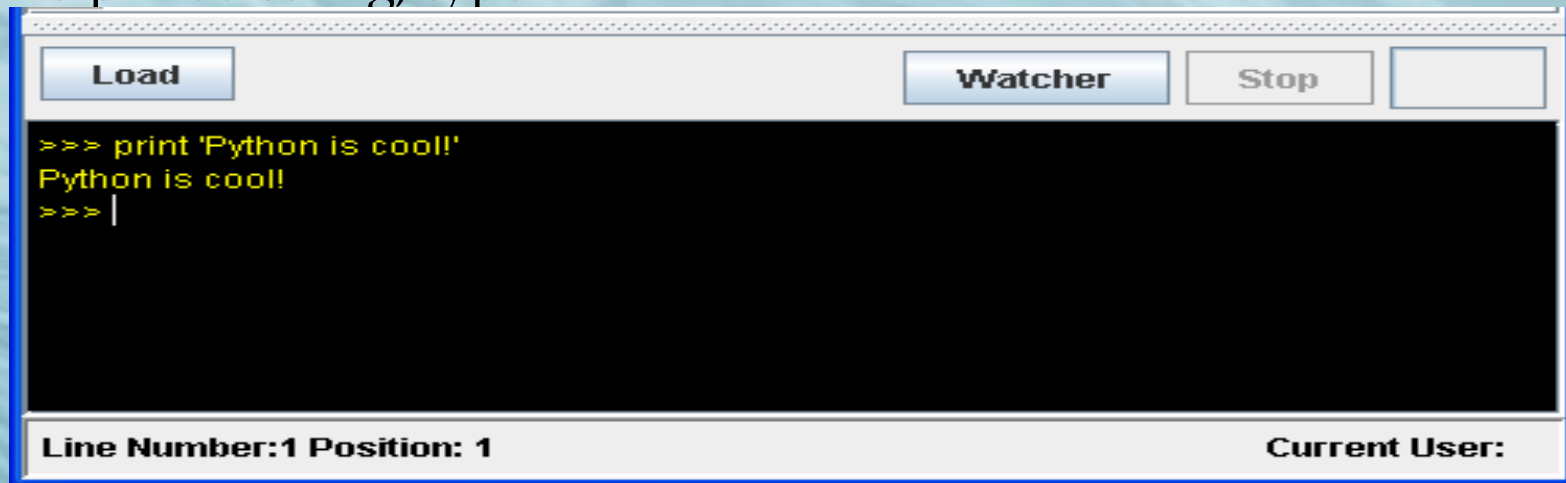
Assigning and Comparing Variables

- When you assign a variable, you use a simple equals sign (=)
- When you compare one variable to another, you use a double equals sign (==)
- You will learn when to use the double equals sign when doing “if” statements later on

Strrrings!

Strings are basically lines of words, phrases, or sentences

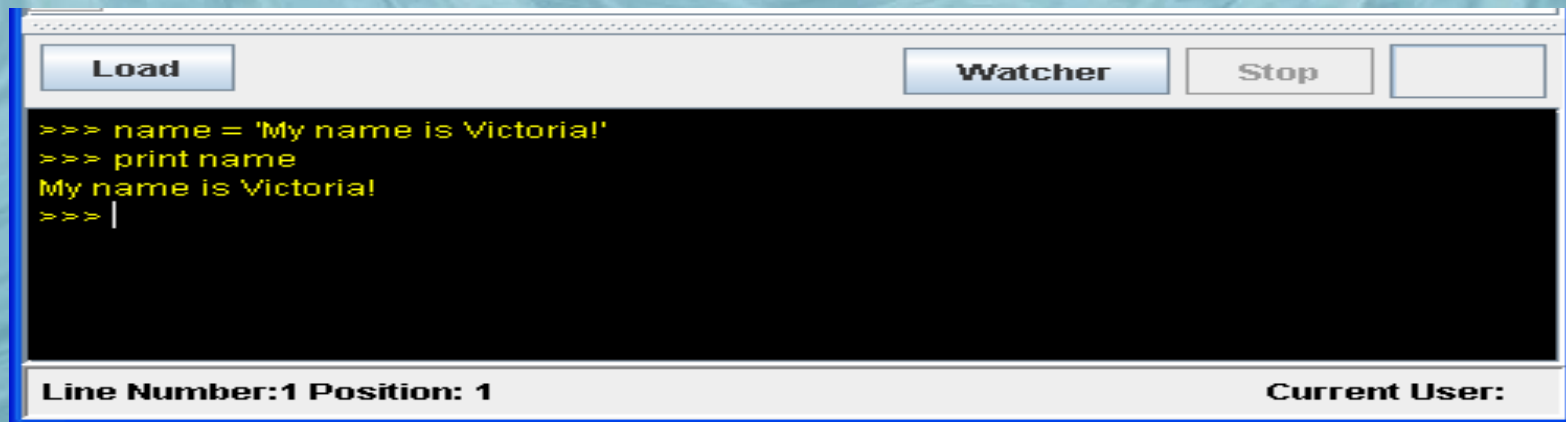
To print a string, type in:

A screenshot of a Python shell window. The window has a title bar and a menu bar. Below the menu bar are four buttons: 'Load', 'Watcher', 'Stop', and an empty button. The main area is a black terminal with yellow text. The text shows a prompt '>>>' followed by the command 'print 'Python is cool!'' and the output 'Python is cool!'. The prompt '>>>' is followed by a vertical bar '|'. 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.

```
>>> print 'Python is cool!'
Python is cool!
>>> |
```

Line Number:1 Position: 1 Current User:

You can even set variables:

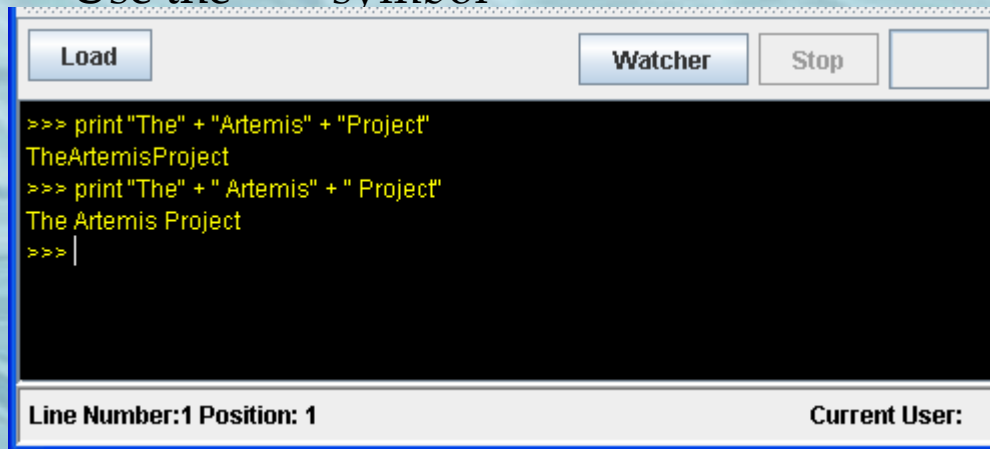
A screenshot of a Python shell window, similar to the one above. It shows a prompt '>>>' followed by the command 'name = 'My name is Victoria!'' and the output 'My name is Victoria!'. The prompt '>>>' is followed by the command 'print name' and the output 'My name is Victoria!'. The prompt '>>>' is followed by a vertical bar '|'. The status bar at the bottom shows 'Line Number:1 Position: 1' and 'Current User:'.

```
>>> name = 'My name is Victoria!'
>>> print name
My name is Victoria!
>>> |
```

Line Number:1 Position: 1 Current User:

String Concatenation

- Strings can be concatenated (glued together) in two ways:
 - Use the “+” symbol



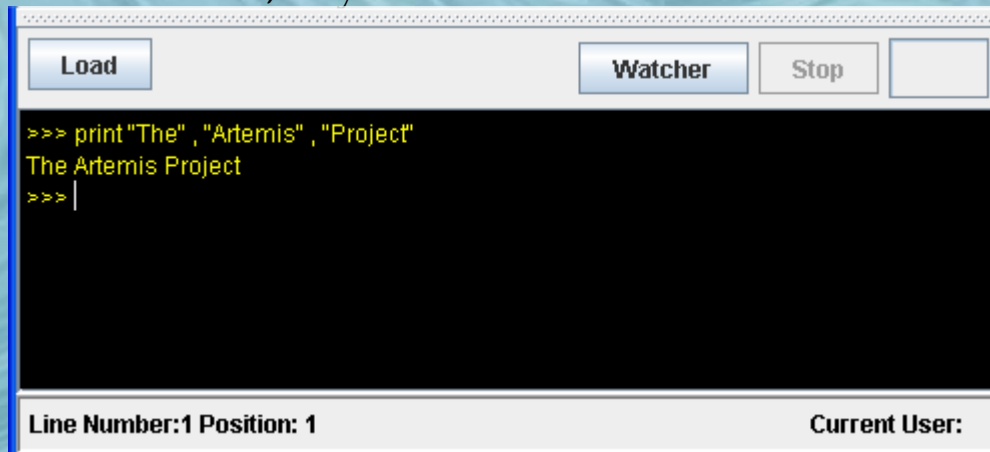
The screenshot shows a Python interpreter window with a title bar containing 'Load', 'Watcher', 'Stop', and an empty button. The main area is a black console with yellow text. It shows two lines of code being executed: `>>> print "The" + "Artemis" + "Project"` which outputs `TheArtemisProject`, and `>>> print "The" + " Artemis" + " Project"` which outputs `The Artemis Project`. The status bar at the bottom shows 'Line Number:1 Position: 1' and 'Current User:'.

```
>>> print "The" + "Artemis" + "Project"
TheArtemisProject
>>> print "The" + " Artemis" + " Project"
The Artemis Project
>>> |
```

Line Number:1 Position: 1 Current User:

Note: When “+” is used, spaces are not automatically added between words. The programmer needs to actually add them. Or, use the second option, the “,” symbol.

- Use the “,” symbol



The screenshot shows a Python interpreter window with a title bar containing 'Load', 'Watcher', 'Stop', and an empty button. The main area is a black console with yellow text. It shows one line of code being executed: `>>> print "The" , "Artemis" , "Project"` which outputs `The Artemis Project`. The status bar at the bottom shows 'Line Number:1 Position: 1' and 'Current User:'.

```
>>> print "The" , "Artemis" , "Project"
The Artemis Project
>>> |
```

Line Number:1 Position: 1 Current User:

Note: The “,” symbol automatically adds space between words.