Introduction to Python

Dr. David Urminsky
School of Physics and Astronomy
Rochester Institute of Technology

Getting started

1. open an xterm

   type: command+space->xQuartz

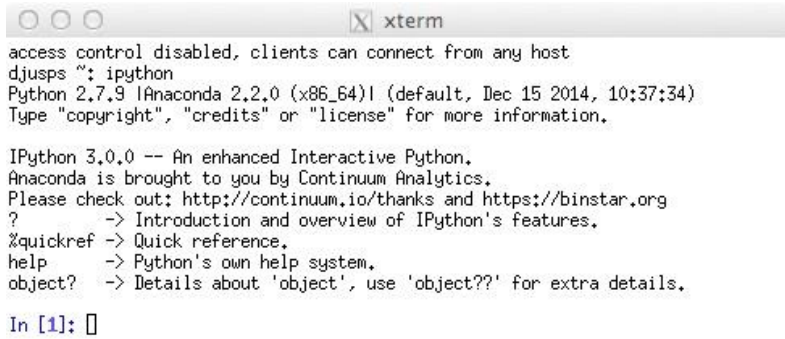
   type: command+n

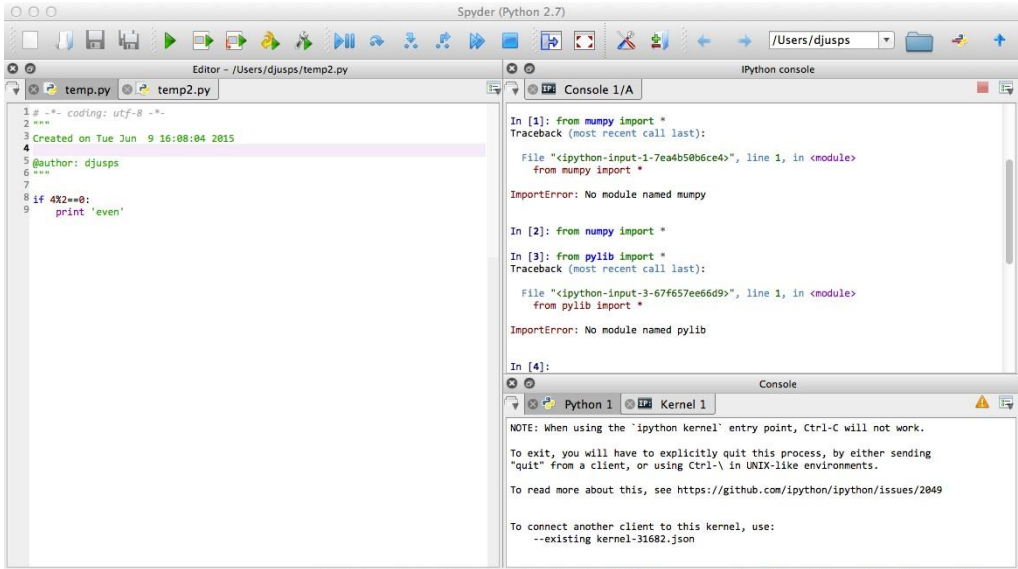
2. from the xterm

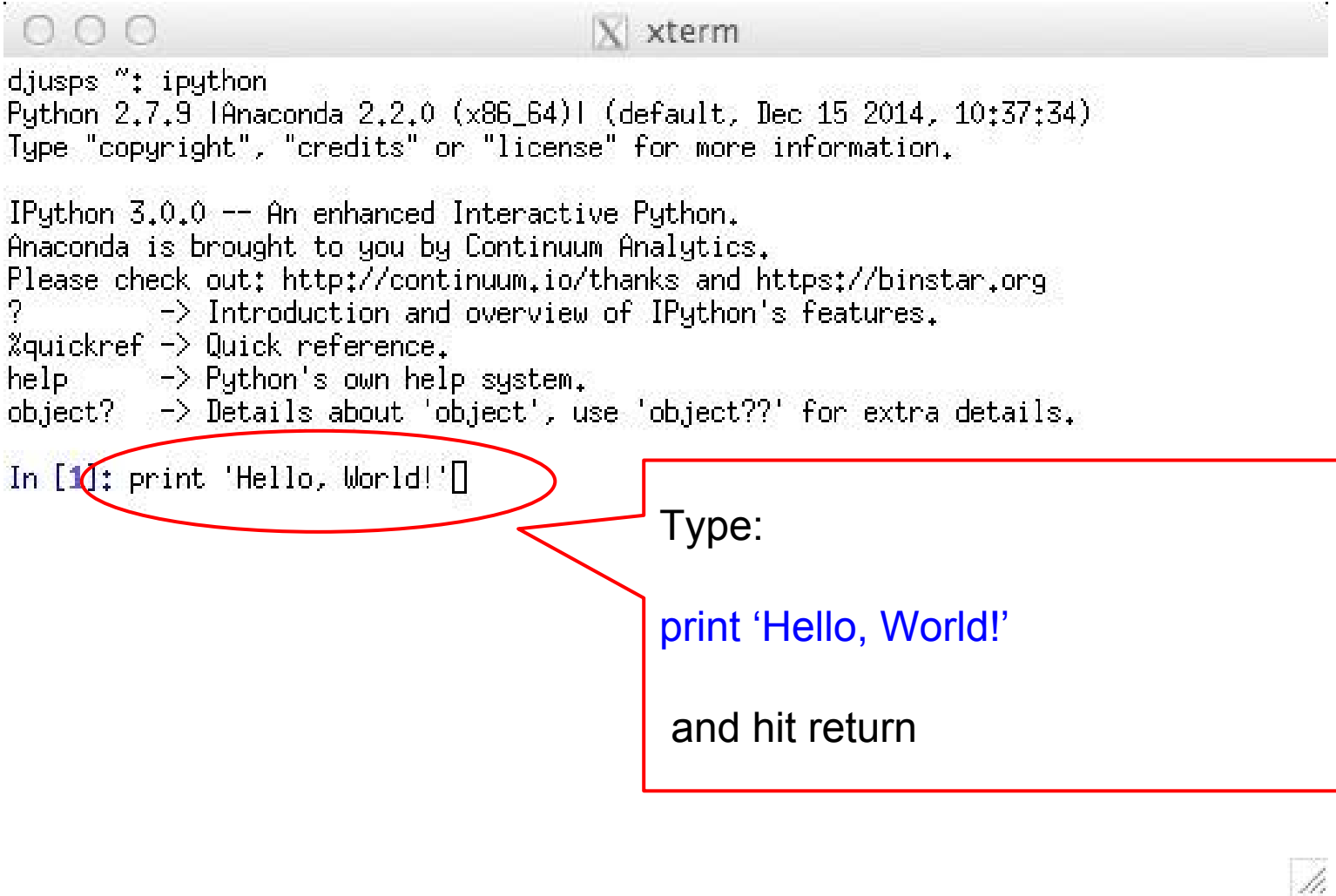
   type: ipython

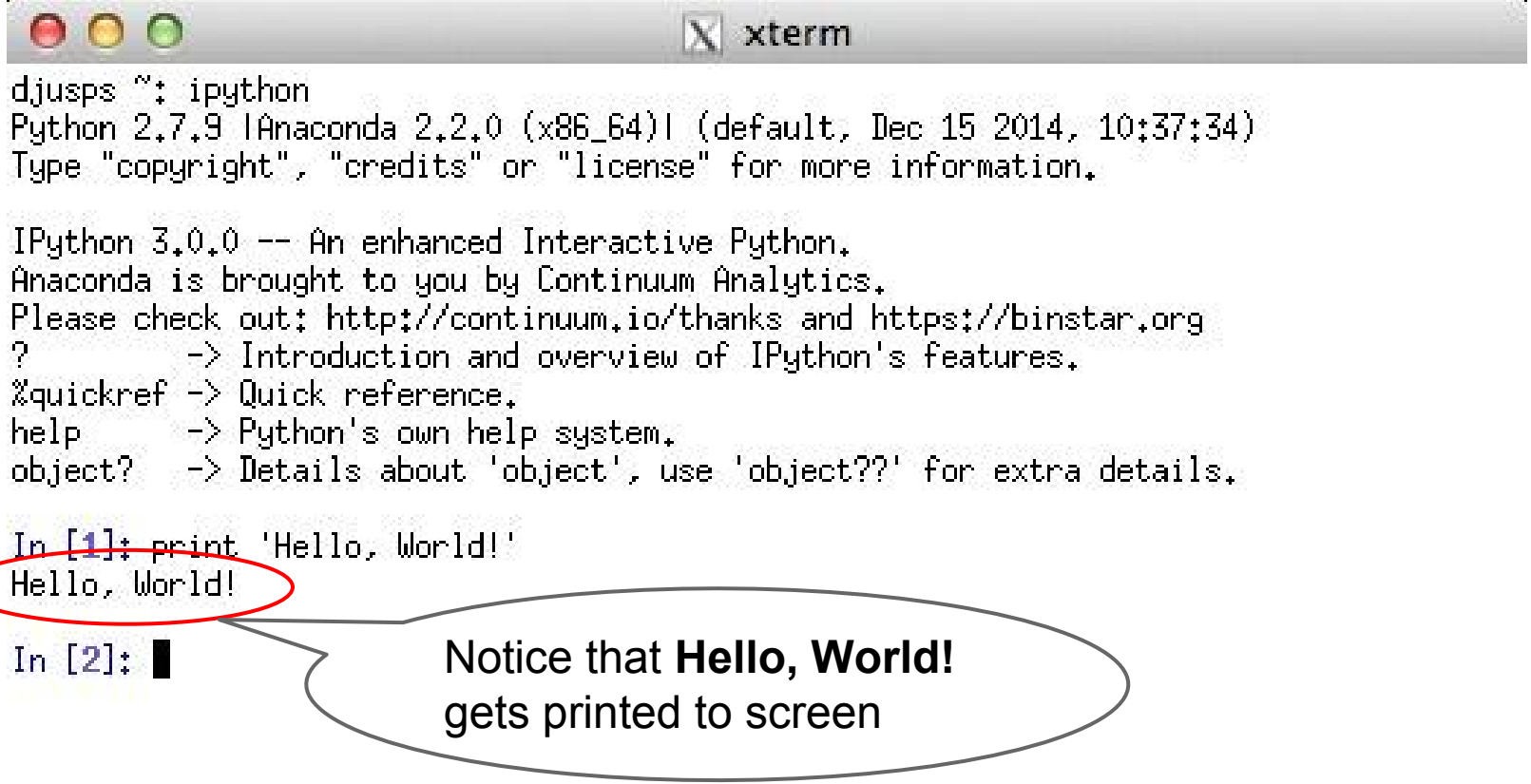
   or

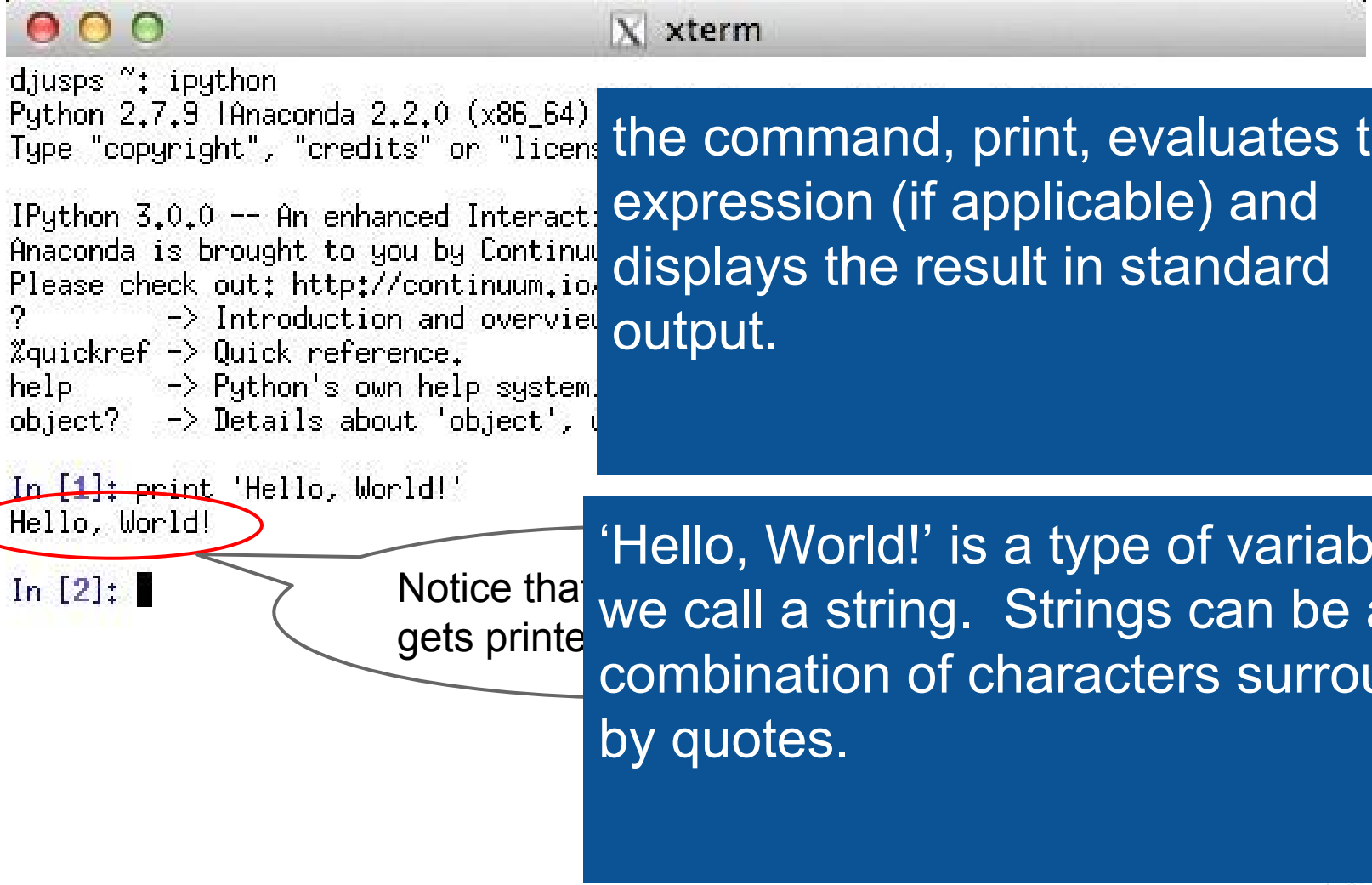
   type: spyder

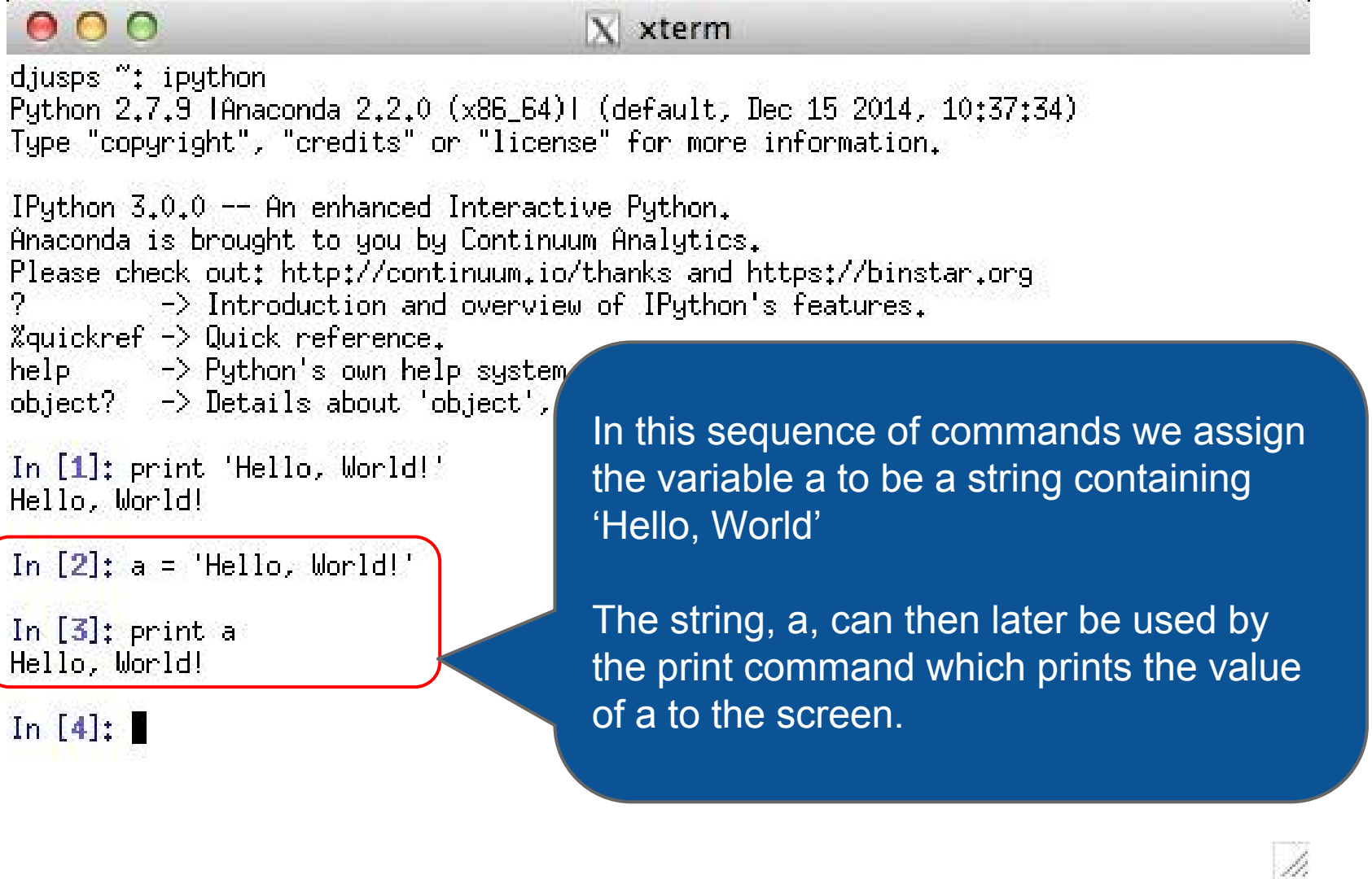





xterm
djusps ~: ipython
Python 2.7.9 |Anaconda 2.2.0 (x86_64)| (default, Dec 15 2014, 10:37:34)
Type "copyright", "credits" or "license" for more information.
IPython 3.0.0 -- An enhanced Interactive Python.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://binstar.org
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.
In [1]: print 'Hello, World!'
Type:
print ‘Hello, World!’
and hit return


xterm
djusps ~: ipython
Python 2.7.9 |Anaconda 2.2.0 (x86_64)| (default, Dec 15 2014, 10:37:34)
Type "copyright", "credits" or "license" for more information.
IPython 3.0.0 -- An enhanced Interactive Python.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://binstar.org
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.
In [1]: print 'Hello, World!'
Hello, World!
In [2]:
Notice that Hello, World!
gets printed to screen


xterm
djusps ~: ipython
Python 2.7.9 |Anaconda 2.2.0 (x86_64)
Type "copyright", "credits" or "licens
IPython 3.0.0 -- An enhanced Interact
Anaconda is brought to you by Continu
Please check out: http://continuum.io
? -> Introduction and overvie
%quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object',
In [1]: print 'Hello, World!'
Hello, World!
In [2]:
the command, print, evaluates the expression (if applicable) and displays the result in standard output.
Notice tha
gets printe
‘Hello, World!’ is a type of variable we call a string. Strings can be a combination of characters surround by quotes.


xterm
djusps ~: ipython
Python 2.7.9 |Anaconda 2.2.0 (x86_64)| (default, Dec 15 2014, 10:37:34)
Type "copyright", "credits" or "license" for more information.
IPython 3.0.0 -- An enhanced Interactive Python.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://binstar.org
? -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help -> Python's own help system
object? -> Details about 'object',
In [1]: print 'Hello, World!'
Hello, World!
In [2]: a = 'Hello, World!'
In [3]: print a
Hello, World!
In [4]:
In this sequence of commands we assign the variable a to be a string containing 'Hello, World'
The string, a, can then later be used by the print command which prints the value of a to the screen.

Types of variables

strings: ‘Hello,World’

int: +/- integers eg. 0, 1, 2 etc

float: real numbers, eg. 3.14159

complex: numbers of the form a+bj (j=sqrt(-1))
eg. 2.03 + 3.22*j

xterm

```
In [41]: a = 'Hello, World!'

In [42]: print a
Hello, World!

In [43]: b = 234  # assigns the variable b to be the integer 234

In [44]: print b
234

In [45]: c = 3.323 # assign the variable c to be the float 3.323

In [46]: print c
3.323

In [47]: d = 3+4j # an example of a complex number

In [48]: print d
(3+4j)

In [49]: d2 = complex(b,c) # use previous variable to construct complex number d2

In [50]: print d2
(234+3.323j)

In [51]: print d2.real, d2.imag
234.0 3.323

In [52]:
```

Now you try it!

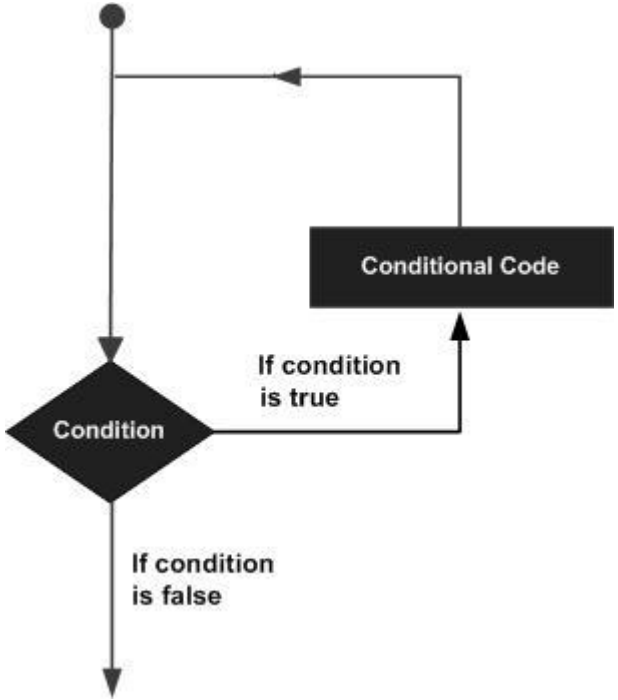
Download the file variables.py

Try executing the commands found in variables.py

Loops

Loops are a sequence of commands which are repeated over and over again until certain conditions are met.

There are two types in python:



Loops

Loops are a sequence of commands which are repeated over and over again until certain conditions are met.

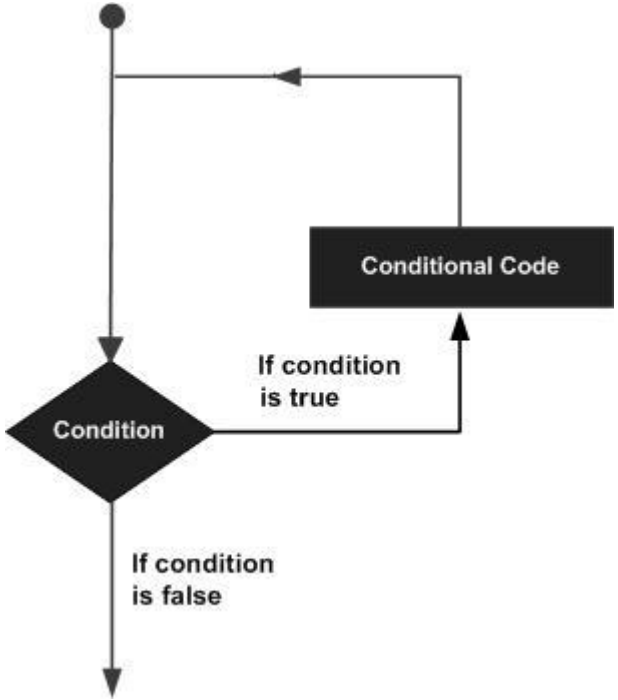
There are two types in python:

While loop

Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body.

for loop

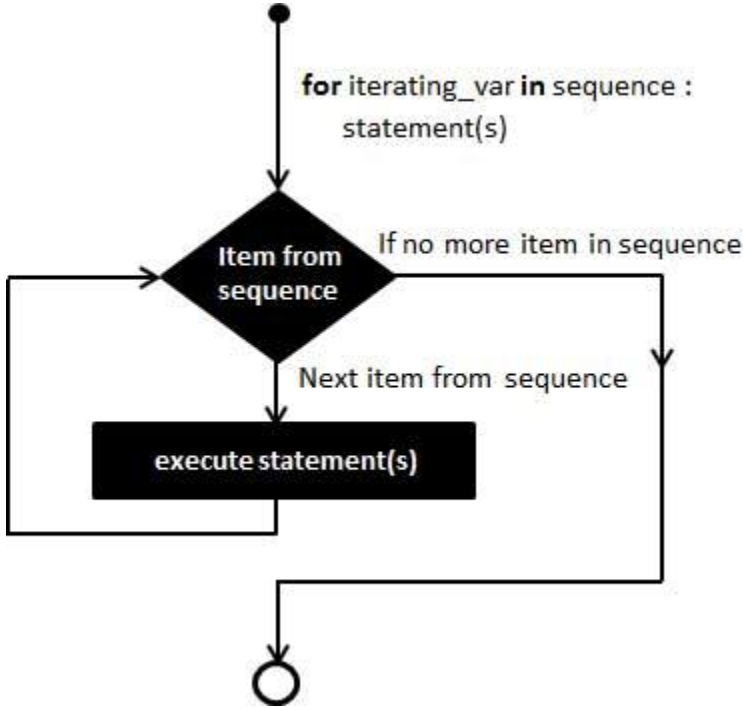
Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.



For loops

Example 1

```
for letter in 'Python':     # First Example
   print 'Current Letter :', letter
```

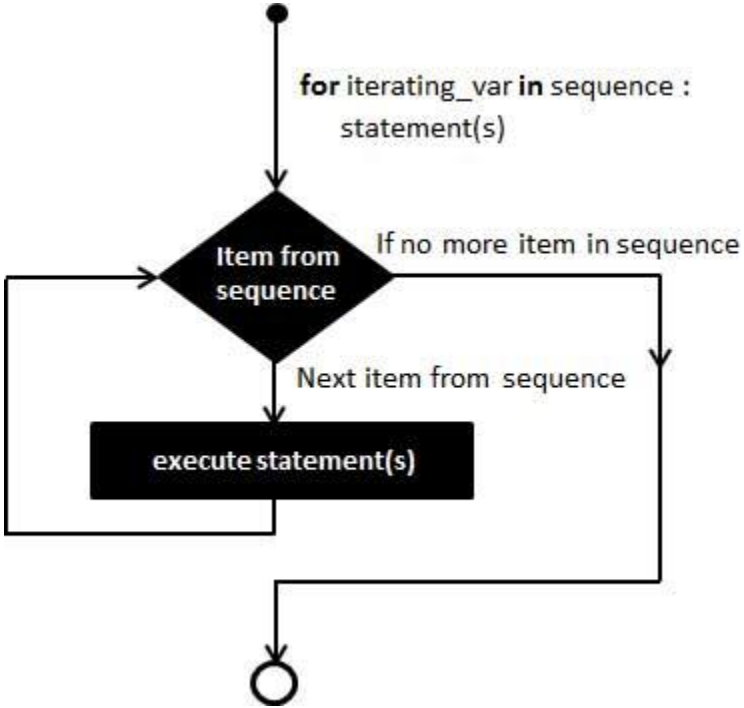


For loops

Example 1

```
for letter in 'Python':     # First Example
   print 'Current Letter :', letter
```

for	indicates the start of for loop
letter	variable which chances on each iteration
in	elements of what follows this will be stored in letter
'Python'	string
:	indicates what follows is in the loop



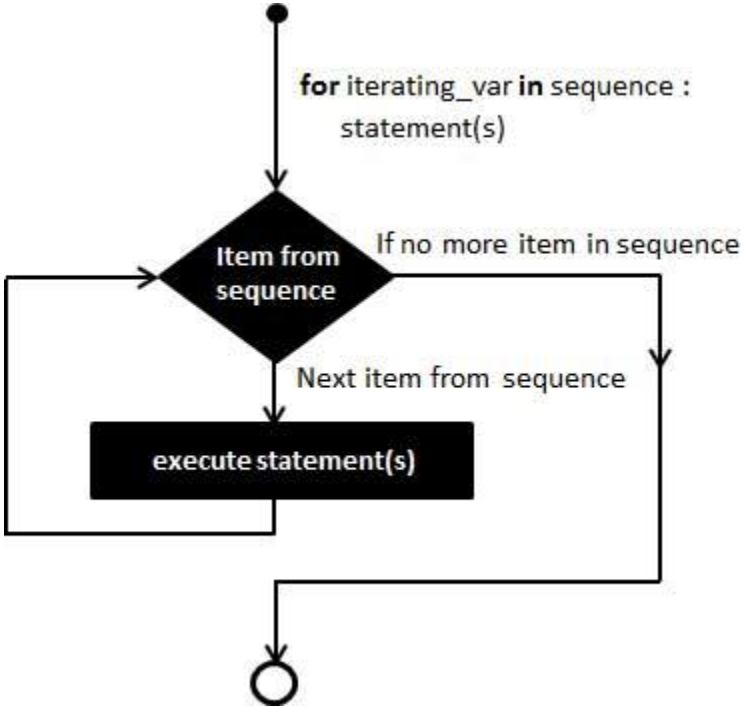
For loops

Example 1

```
for letter in 'Python':     # First Example
   print 'Current Letter :', letter
```

Output

```
Current Letter : P
Current Letter : y
Current Letter : t
Current Letter : h
Current Letter : o
Current Letter : n
```



For loops

Example 2

```
fruits = ['banana', 'apple',  'mango']
for fruit in fruits:        # Second Example
   print 'Current fruit :', fruit

print "Good bye!"
```

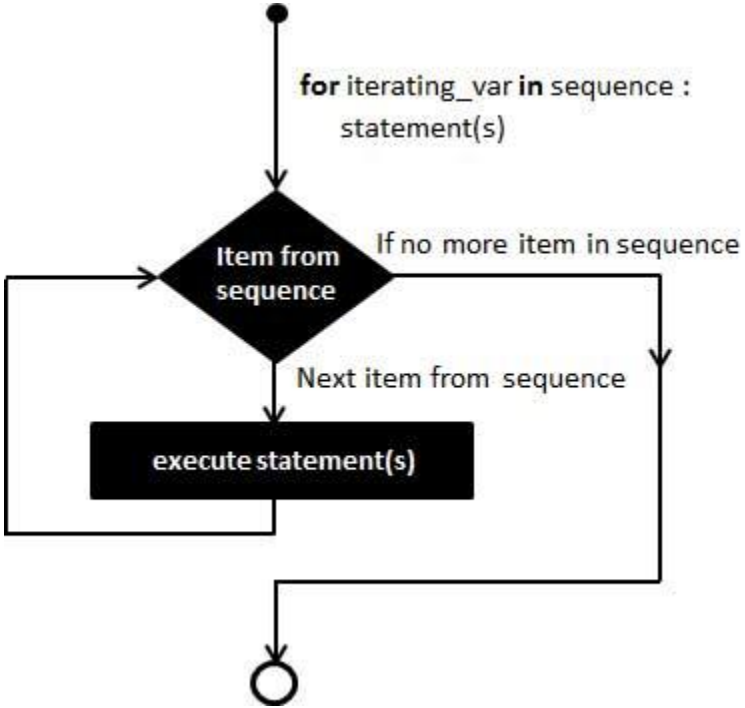
Output

```
Current fruit : banana

Current fruit : apple

Current fruit : mango

Good bye!
```



While loops

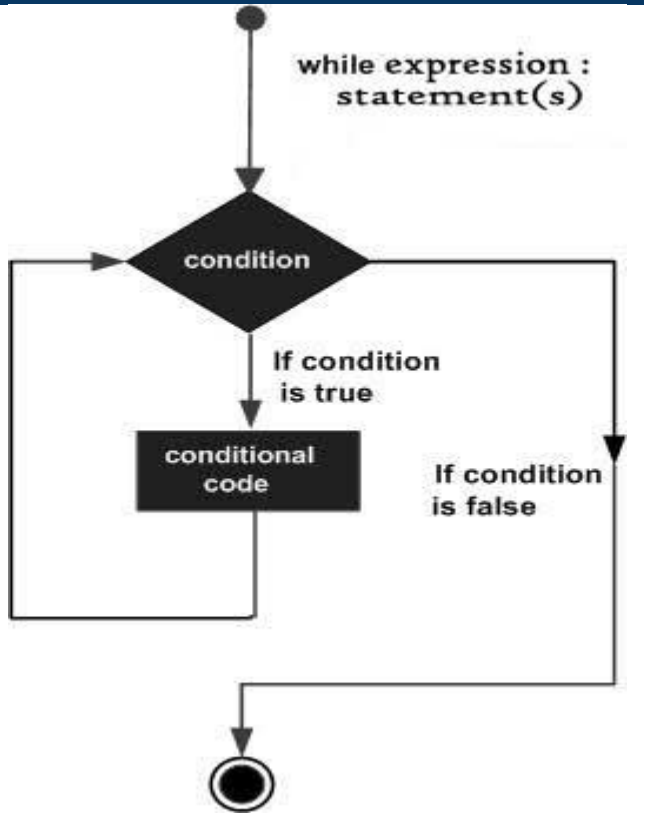
Output

```
The count is: 0
The count is: 1
The count is: 2
The count is: 3
The count is: 4
The count is: 5
The count is: 6
The count is: 7
The count is: 8
Good bye!
```

Example 1

```
count = 0
while (count < 9):
   print 'The count is:', count
   count = count + 1

print "Good bye!"
```



Conditional Statements

If Statements

An if statement consists of a boolean expression followed by one or more statements

Boolean expressions

A boolean expression can be either true or false.

Conditional Statements

Example 1

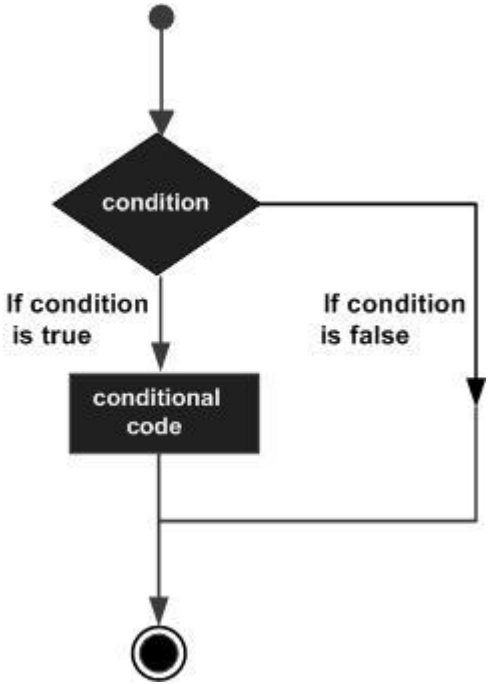
```
var = 100

if ( var  == 100 ) : print "Value of expression is 100"

print "Good bye!"
```

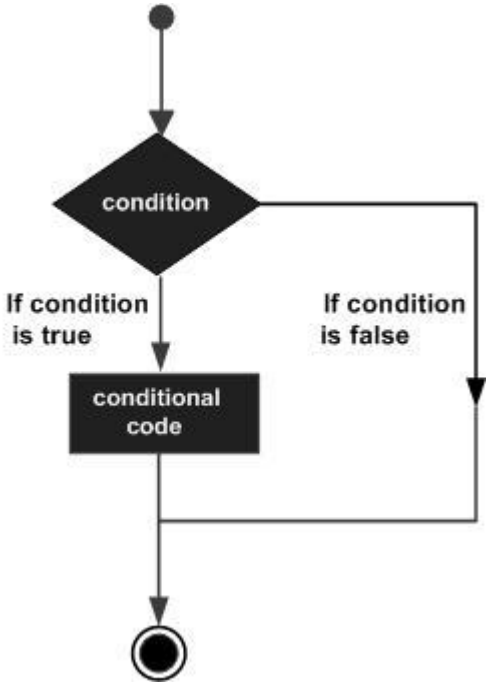
Output

```
Value of expression is 100
Good bye!
```



Conditional Statements

==	If the values of two operands are equal, then the condition becomes true.	a==b is not true
!=	If values of two operands are not equal, then condition becomes true.	
<>	If values of two operands are not equal, then condition becomes true.	a<>b is true
<=	less than or equal	2<=3 is true
>=	greater than or equal	2>=3 is false
<	less than	2<3 is true
>	greater than	2>2 is false



Conditional Statements

Example 1

```
In [9]: var = 110

In [10]: if var == 100:
   ....:     print 'the value of var is', var
   ....: else:
   ....:     print 'the value of var is not 100'
   ....:
```

Output

```
the value of var is not 100
```