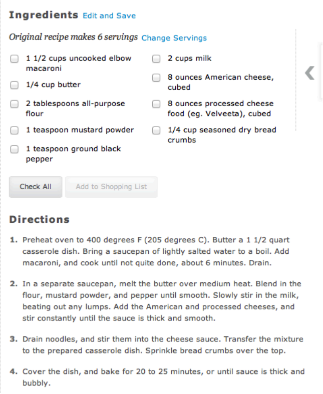
MCS-177
Introduction to
Computer Science I
- Spring 2014

Louis Yu

What is computer science?

- Computer Science is the study of Algorithms.
  - Solving problem -> turning the solution into step by step instruction for computers to perform
  - Programs



Ingredients Edit and Save

Original recipe makes 6 servings Change Servings

□ 1 1/2 cups uncooked elbow macaroni	□ 2 cups milk
□ 1/4 cup butter	□ 8 ounces American cheese, cubed
□ 2 tablespoons all-purpose flour	□ 8 ounces processed cheese food (eg. Velveeta), cubed
□ 1 teaspoon mustard powder	□ 1/4 cup seasoned dry bread crumbs
□ 1 teaspoon ground black pepper	

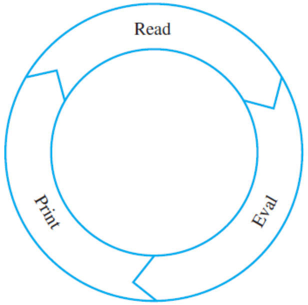
Directions

1. Preheat oven to 400 degrees F (205 degrees C). Butter a 1 1/2 quart casserole dish. Bring a saucepan of lightly salted water to a boil. Add macaroni, and cook until not quite done, about 6 minutes. Drain.
2. In a separate saucepan, melt the butter over medium heat. Blend in the flour, mustard powder, and pepper until smooth. Slowly stir in the milk, beating out any lumps. Add the American and processed cheeses, and stir constantly until the sauce is thick and smooth.
3. Drain noodles, and stir them into the cheese sauce. Transfer the mixture to the prepared casserole dish. Sprinkle bread crumbs over the top.
4. Cover the dish, and bake for 20 to 25 minutes, or until sauce is thick and bubbly.

Python

- Python is a good beginning tool
  - Language such as Java and C++ are great, but they require you to keep track of many more details
  - They are also harder to learn

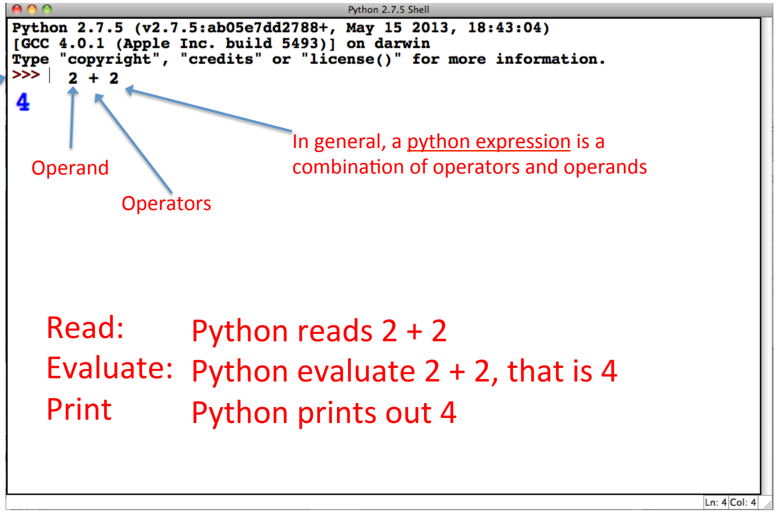
At a higher level, the Python interpreter does 3 things:
1) Read
- Python readers one line of input
2) Evaluate
- Python evaluate that one line of input
3) Print
- Python prints out the result



The Read-Eval-Print loop in Python

Example

Let's click on Python IDLE, this will start a Python shell



```
Python 2.7.5 (v2.7.5:ab05e7dd2788+, May 15 2013, 18:43:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 2 + 2
4
```

prompt

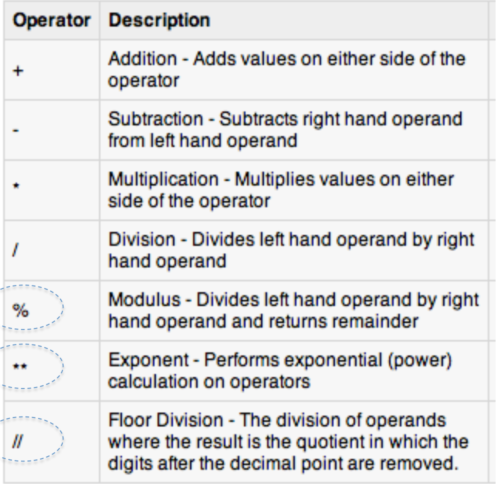
Operand

Operators

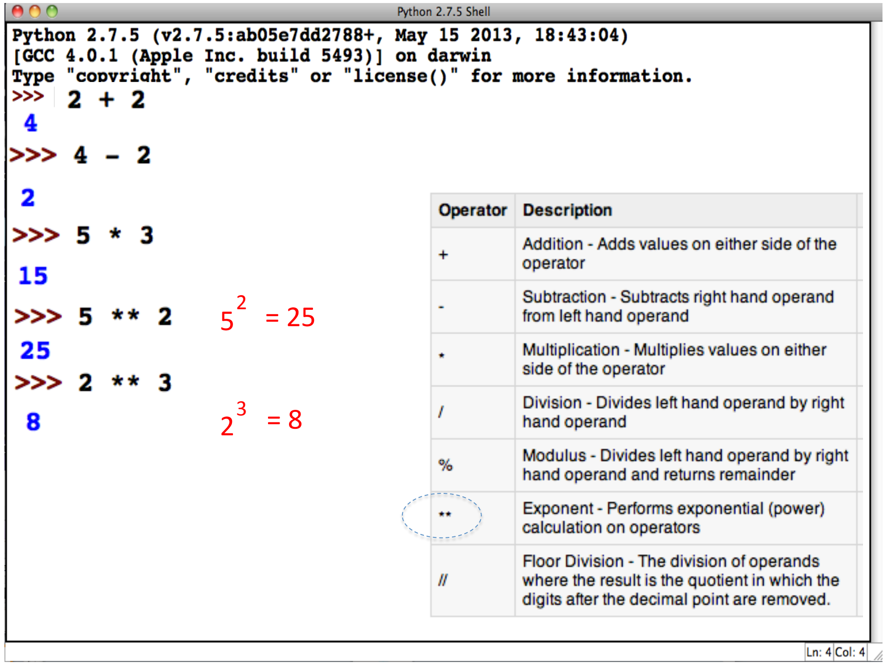
In general, a python expression is a combination of operators and operands

Read: Python reads 2 + 2
Evaluate: Python evaluate 2 + 2, that is 4
Print Python prints out 4

Python Operators



Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.

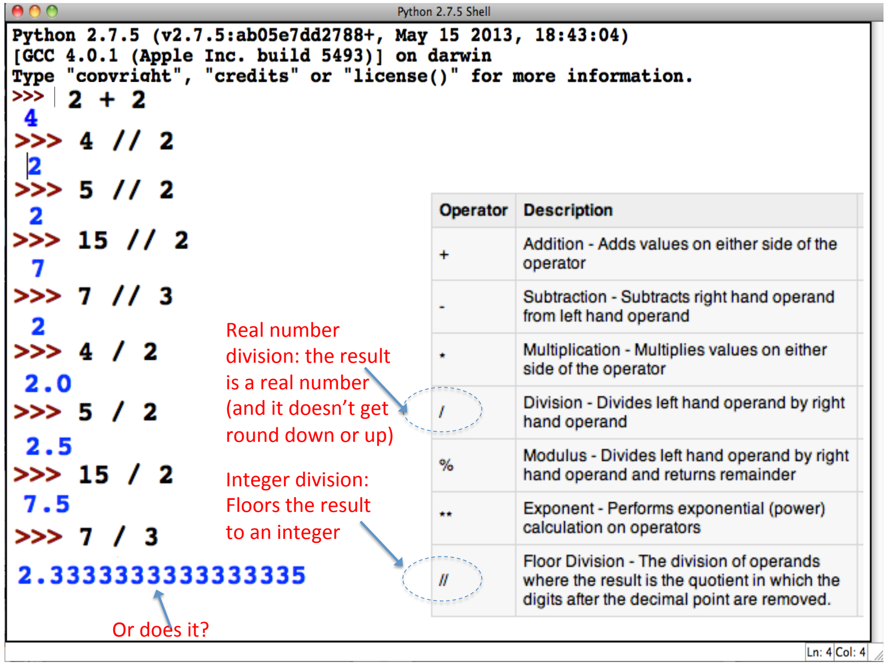


```
Python 2.7.5 (v2.7.5:ab05e7dd2788+, May 15 2013, 18:43:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 2 + 2
4
>>> 4 - 2
2
>>> 5 * 3
15
>>> 5 ** 2
25
>>> 2 ** 3
8
```

$5^2 = 25$

$2^3 = 8$

Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.



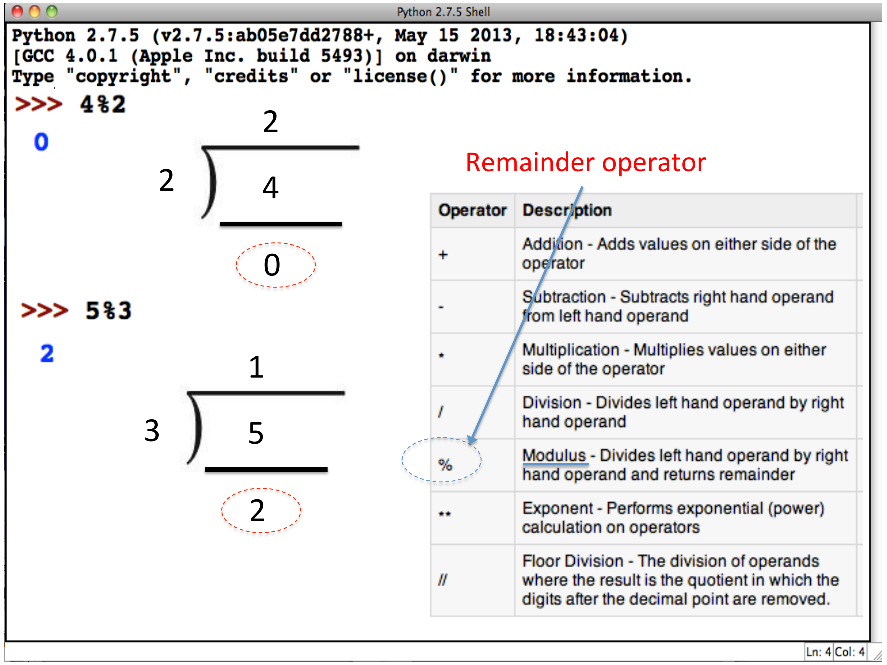
```
Python 2.7.5 (v2.7.5:ab05e7dd2788+, May 15 2013, 18:43:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 2 + 2
4
>>> 4 // 2
2
>>> 5 // 2
2
>>> 15 // 2
7
>>> 7 // 3
2
>>> 4 / 2
2.0
>>> 5 / 2
2.5
>>> 15 / 2
7.5
>>> 7 / 3
2.3333333333333335
```

Real number division: the result is a real number (and it doesn't get round down or up)

Integer division: Floors the result to an integer

Or does it?

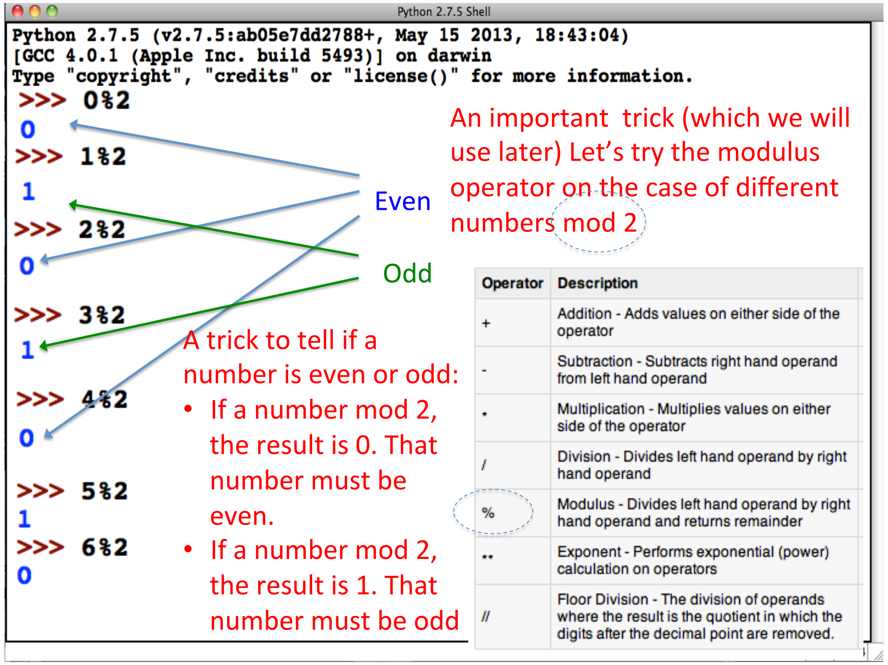
Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.



```
Python 2.7.5 (v2.7.5:ab05e7dd2788+, May 15 2013, 18:43:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 4%2
0
>>> 5%3
2
```

Remainder operator

Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.



```
Python 2.7.5 (v2.7.5:ab05e7dd2788+, May 15 2013, 18:43:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 0%2
0
>>> 1%2
1
>>> 2%2
0
>>> 3%2
1
>>> 4%2
0
>>> 5%2
1
>>> 6%2
0
```

An important trick (which we will use later) Let's try the modulus operator on the case of different numbers mod 2

A trick to tell if a number is even or odd:

- If a number mod 2, the result is 0. That number must be even.
- If a number mod 2, the result is 1. That number must be odd

Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.

Integer and Floating Point

- Integers are the whole numbers you learned in math class. We've seen nothing but integer operations so far (even though some times the result might be floating point)
- Floating point numbers are Python's approximation of what you called real numbers in math class.
- Approximation because floating point numbers cannot have an infinite number of digits following the decimal point

```
>>> 5/3
1.6666666666666667
```

```
>>> 2.0 + 2.5
 4.5
>>> 2 + 2.5
 4.5
>>> 4.5 - 2.0
2.5
>>> 4.5 - 2
2.5
>>> 3.0 **2
9.0
>>> 3 ** 2
9
>>> 3 ** 3.0
27.0
```

Mixing integers and floats

When mixing different types of numbers, you can figure out what the result will be converted to by applying the following rules:

- If either argument is a floating-point number, the other is converted to floating point. Thus result is a floating point
- Only if both arguments are plain integers, then no conversion is needed

Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.

```
>>> 4 // 2
2
>>> 5 // 2
2
>>> 15 // 2
7
>>> 5.0//2.0
2.0
>>> 4.0//2
2.0
>>> 15.0//2
7.0
>>> 4 / 2
2.0
>>> 5 / 2
2.5
>>> 15 / 2
7.5
>>> 4.0/2
2.0
>>> 5.0/2
2.5
>>> 15.0/2.0
7.5
```

Mixing integers and floats

- If either argument is a floating-point number, the other is converted to floating point. Thus result is a floating point
- If both arguments are plain integers, then no conversion is needed

The expression is evaluated first (integer division), then the result is convert to float/int

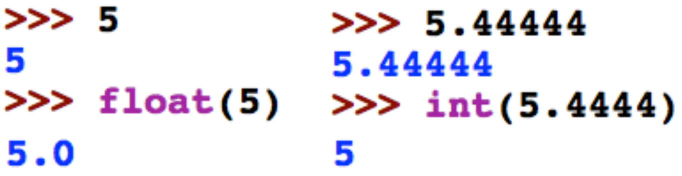
Real number division: the result is a real number

Integer division: Floors the result to an integer

Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.

Casting

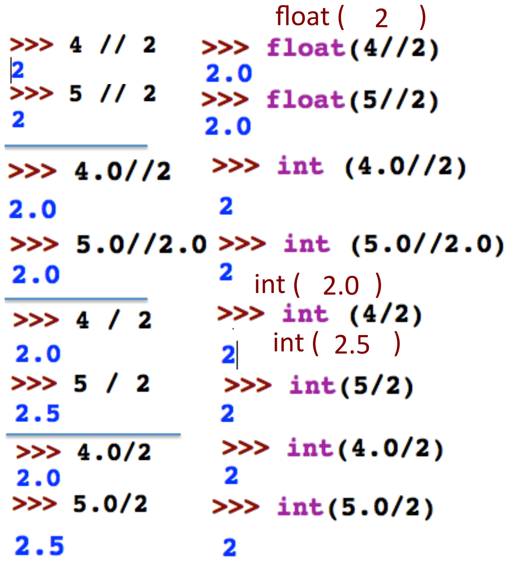
- You can also tell Python to explicitly convert a number to either an integer or floating-point by using the int() or float() function. The bracket is used to surround the expression you want to convert
- Caution: if you convert an expression. The expression is evaluated first, then the result (from the evaluation) is convert to float/int



```
>>> 5                  >>> 5.44444
5                      5.44444
>>> float(5)           >>> int(5.4444)
5.0                    5
```

Casting:

- int(), float()
- Caution: if you convert an expression. The expression is evaluated first, then the result (from the evaluation) is convert to int/float

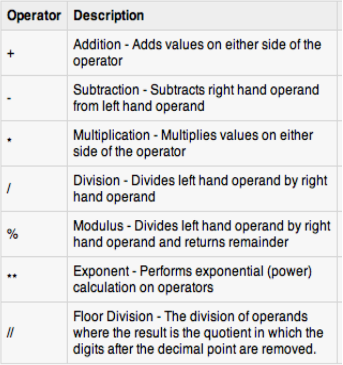


```
                         float (   2   )
>>> 4 // 2           >>> float(4//2)
2                    2.0
>>> 5 // 2           >>> float(5//2)
2                    2.0

>>> 4.0//2           >>> int (4.0//2)
2.0                  2
>>> 5.0//2.0         >>> int (5.0//2.0)
2.0                  2   int (  2.0  )

>>> 4 / 2            >>> int (4/2)
2.0                  2    int (  2.5   )
>>> 5 / 2            >>> int(5/2)
2.5                  2

>>> 4.0/2            >>> int(4.0/2)
2.0                  2
>>> 5.0/2            >>> int(5.0/2)
2.5                  2
```



Operator	Description
+	Addition - Adds values on either side of the operator
-	Subtraction - Subtracts right hand operand from left hand operand
*	Multiplication - Multiplies values on either side of the operator
/	Division - Divides left hand operand by right hand operand
%	Modulus - Divides left hand operand by right hand operand and returns remainder
**	Exponent - Performs exponential (power) calculation on operators
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed.

Naming objects

- Very often we have an object that we would like to remember, if we want to refer to that object later.
- e.g: pi = 3.14159
- In python we can name an object using an "assignment statement"

Assignment statement – this particular form of equal sign is used to assign value to a variable

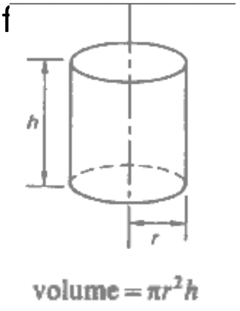
Variable name = python expression

↑
variable

Yes, there is another type of equal sign (which we will introduce later)

Example:

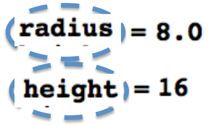
- We want to calculate the volume of a cylinder



volume = area of base * height

Let's calculate a case that:

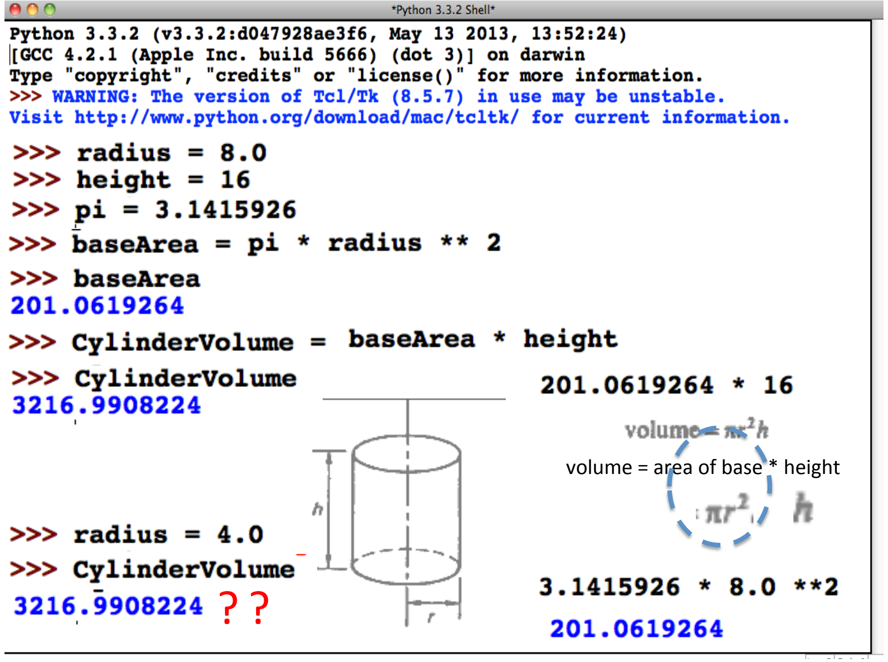
r = 8.0
h = 16



```
radius = 8.0
height = 16
```

Note: you can pretty much name the variable ANYTHING (except a few name reserved for other purposes which we will mention later). Remember, it's just a name

Variable name = python expression



```
Python 3.3.2 (v3.3.2:d047928ae3f6, May 13 2013, 13:52:24)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.7) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.
>>> radius = 8.0
>>> height = 16
>>> pi = 3.1415926
>>> baseArea = pi * radius ** 2
>>> baseArea
201.0619264
>>> CylinderVolume = baseArea * height
>>> CylinderVolume
3216.9908224
>>> radius = 4.0
>>> CylinderVolume
3216.9908224
```

? ?

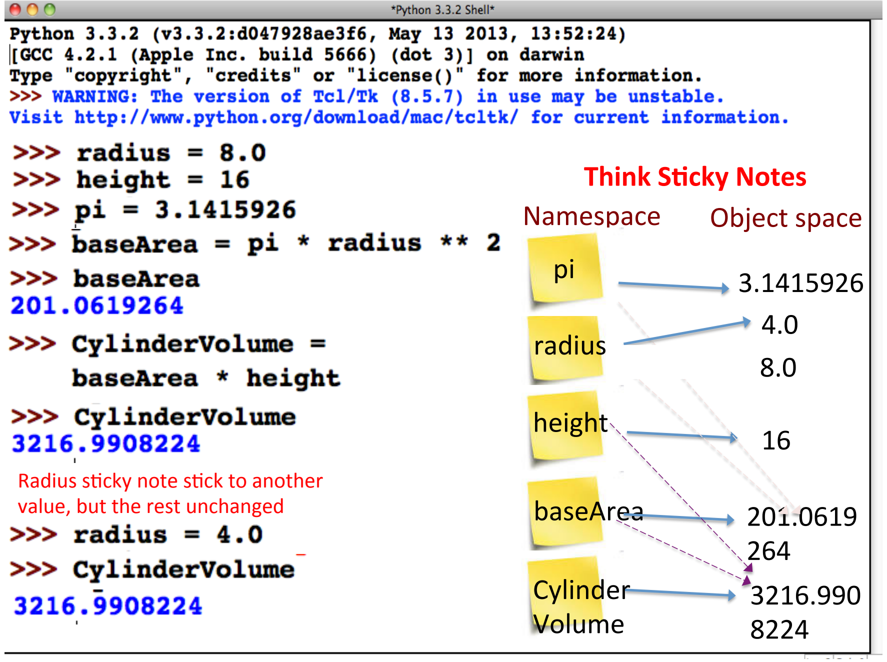
201.0619264 * 16

volume = $\pi r^2 h$

volume = area of base * height

3.1415926 * 8.0 **2

201.0619264

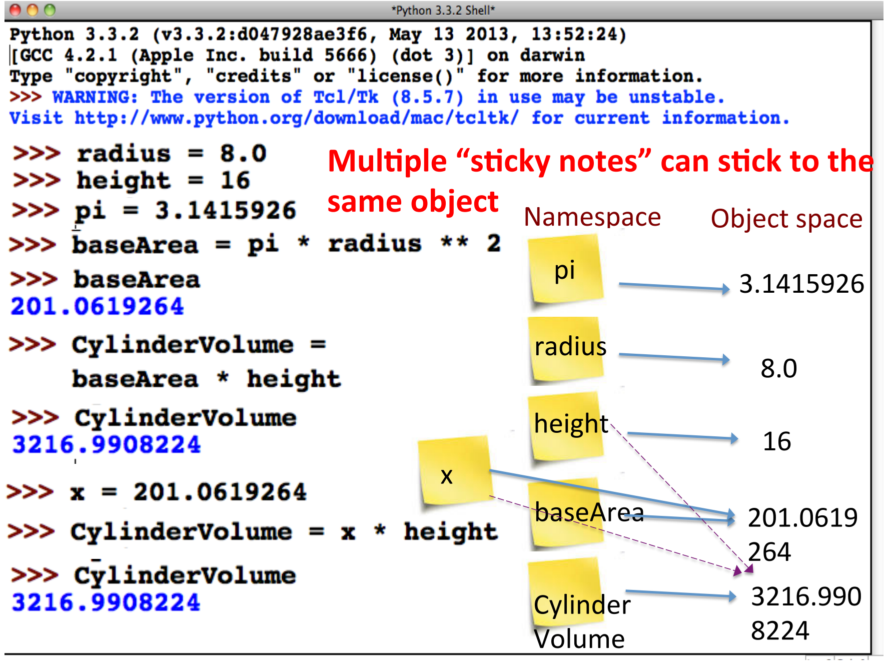


```
Python 3.3.2 (v3.3.2:d047928ae3f6, May 13 2013, 13:52:24)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.7) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.
>>> radius = 8.0
>>> height = 16
>>> pi = 3.1415926
>>> baseArea = pi * radius ** 2
>>> baseArea
201.0619264
>>> CylinderVolume =
    baseArea * height
>>> CylinderVolume
3216.9908224
```

Radius sticky note stick to another value, but the rest unchanged

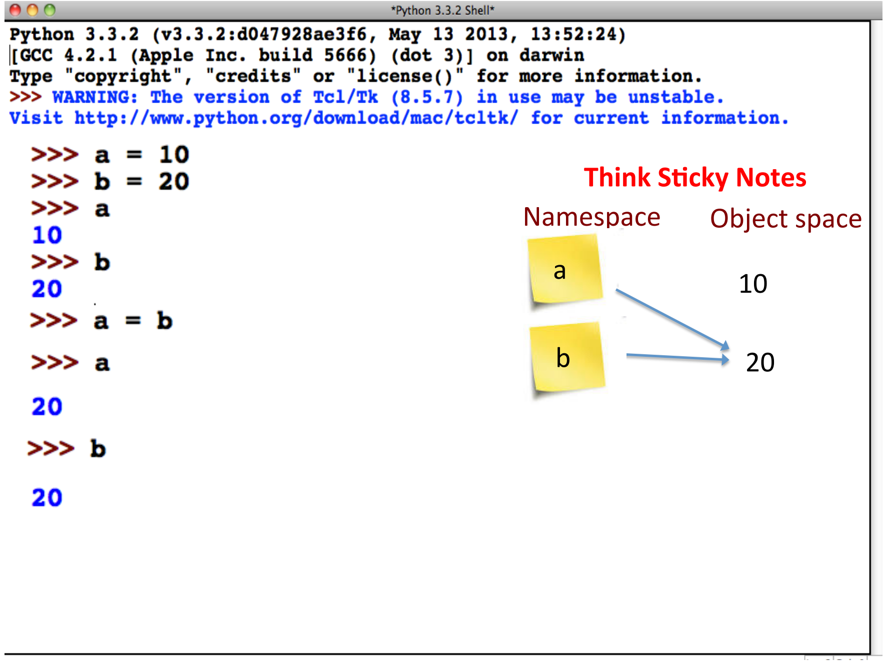
```
>>> radius = 4.0
>>> CylinderVolume
3216.9908224
```

Think Sticky Notes



```
Python 3.3.2 (v3.3.2:d047928ae3f6, May 13 2013, 13:52:24)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.7) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.
>>> radius = 8.0
>>> height = 16
>>> pi = 3.1415926
>>> baseArea = pi * radius ** 2
>>> baseArea
201.0619264
>>> CylinderVolume =
    baseArea * height
>>> CylinderVolume
3216.9908224
>>> x = 201.0619264
>>> CylinderVolume = x * height
>>> CylinderVolume
3216.9908224
```

Multiple “sticky notes” can stick to the same object

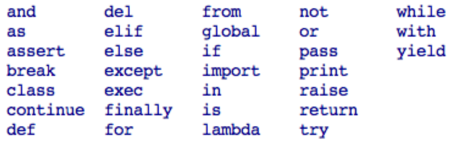


```
Python 3.3.2 (v3.3.2:d047928ae3f6, May 13 2013, 13:52:24)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.7) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.
>>> a = 10
>>> b = 20
>>> a
10
>>> b
20
>>> a = b
>>> a
20
>>> b
20
```

Think Sticky Notes

Rules about naming things in python

- Name must start with either a letter or a _
- Python is case sensitive
  - baseArea, basearea, BaseArea are all different
- Some names are reserved by python for its own use.



```
and       del       from      not       while
as        elif      global    or        with
assert    else      if        pass      yield
break     except    import    print
class     exec      in        raise
continue  finally   is        return
def       for       lambda    try
```