|  |  |  |
| --- | --- | --- |
| * In your Terminal program, run **>python FILE.py**, hit **CTRL-C (^C)** to get out of python * **For help**, on Windows type **python -m pydoc raw\_input** instead. Get out of pydoc by typing q to quit * Use strings to represent numbers with a leading zero NOT integers because if integer is used it will print an octal number (>>>0142 prints 98) | | |
| name = "Hello world"; char1 = "Hello"; char2 = " world"  print "Hello World" **|** print "%s" %name; print "%s" %("Hello World"); print name; print "", char1 + char2; **print repr(“Hello World”) #’Hello World’, %r**  print """There's something going on here. \ With the three double-quotes.""" , **print ‘Age:’, 42** | | |
| **print 'I "said" do not touch this.'** # prints “I “said” do not touch this”  **print "." \* 10** # prints “……….” to the screen  **formatter = "%r %r %r %r"** # assigns variable “%r %r %r %r”to the variable “formatter”  **print formatter % (1, 2, 3, 4)** # prints “1, 2, 3, 4” to the screen  **print formatter % ("one", "two", "three", "four")** # prints“’one’ ’two ’ ’three ’ ’four ’” to the screen  **print formatter % (True, False, False, True)** # prints “True False False True” to the screen  **print formatter % (formatter, formatter, formatter, formatter)** # prints "%r %r %r %r" 4 times to the screen  **print formatter % ("I had.", " Type.", "But it.", "I said.")** # prints “ ’I had.’ ‘Type.’‘But it.’ ‘I said’ ” to the screen | | |
| **print 3 + 2 + 1 - 5 + 4 % 2 - 1 / 4 + 6** # prints ”7”  **print 3 + 2 < 5 – 7** # prints “False”  **print "Is it greater or equal?", 5 >= -2** # prints "Is it greater or equal? True”  **print float(36) /float(7)** # 5.14285714286**,** int(float) converts to integer, float(int) converts integer back to float  **print 21 % 3** # 0, returns the remainder of division  **print 6 \*\* 2**  # 6 to the power of 2; **same as pow(2,3)**  **print id(5)** # 31548456, it's the object's memory address  **print sqrt(5)** # 2.2360679775, when "from math import sqrt" use "sqrt(5)";# when "import math", type "math.sqrt =(5)" | | |
| **longIntEx = 22000000000000000000000L** # Put L to make it long, if there is no L it's an integer  **print type(longIntEx)** # <type 'long'>, prints a type of the variable  **booleanEx = True** # The same as booleanEx = 1 ==1  **print not booleanEx**  # False  **print int(booleanEx)** **# 1, if false print 0; All False: bool(False), bool(None),bool( 0),bool( “”), bool(()), bool([]), bool({})**  **To increment a number use x = 0, x += 1, x \*=2** | | |
| pen = 20; print "I have 20 pens" | print "I have", 20, "pens" | print "I have", pen, "pens" | print "I have %d pens" %pen | print "I have %d pens" %20 | | |
| pen = 20, num1 = 100, num2 = 80, num20 = num1 - num2; print pen |print 20 | print num1 - num2 | print "%d" %num20| print "%d" %(num1 - num2) | | |
| name = "Vlad”, pen = 20; print "%s has %d pens" %("Vlad", 20) | print "%s has %d pens" %(name, pen) **|** | | |
| * **%d or %i**= print numbers; **%s** = print sentences; **%r** = print this no matter what, **%c** = prints a single character * **print “pi” %10.3f** (3.142 -field width 10, 3 decimals); **‘%.\*s’ % (5, ‘Guidance’) (same as ‘%.\*s’, prints ‘Guida’), ‘%-10.2f’ %pi** (allighns pi left) * **\n** (back-slash n) puts a new line character, **\\** will print only one double back-slash; **\t** creates an indent * **"I am 6'2\" tall."** # escape double-quote inside string; **'I am 6\'2" tall.'** # escape single-quote inside string | | |
| print "How old are you? >",  age = raw\_input() | age = input("How old are you? ")  or age = int(raw\_input(How old are you?)) | prompt = “What is your name? >"  name = raw\_input(prompt) |

|  |
| --- |
| **>>> ten\_things = "Apples Oranges Crows Telephone Light Sugar" >>>print ten\_things or >>>print[:]**  **>>> print ten\_things[0]** #a, similar to **print ten\_things[0]** #r  **>>> print ten\_things[0:3]** # ‘her’; **print ten\_things[1:20:2]** # Prints every other character from the string  **>>> print ten\_things[-3:]** # ‘gar’, **similar to print ten\_things[:3]** #’app’  **>>> print ten\_things.find('e')** # 4, prints location of the string  **>>> stuff = ten\_things.split()** # ['Apples', 'Oranges', 'Crows', 'Telephone', 'Light', ‘Sugar’]  **>>> more\_stuff = ["Day", "Night", "Song", "Frisbee", "Corn", "Banana", "Girl", "Boy"]**  **>>> while len(stuff) != 10:** #Goes through the loop until 10 items are added to the list  **stuff.append(more\_stuff.pop())**  **>>> print ' '.join(stuff)** # Apples Oranges Crows Telephone Light Sugar Boy Girl Banana Corn -> converts list to **string**  **>>> print '#'.join(stuff[3:5])** # Telephone#light |
| **database = [[‘albert’,’1234’],[‘david’, ‘3456’],[‘martha’,’6789’]]**  **username = raw\_input(); pin = raw\_input()**  **if [username, pin] in database: print “Access Granted!”** |
| **def split\_sentence(stuff):** """ Break up words."""# ['All', 'good', 'things', 'come', 'to', 'those', 'who', 'wait.']  **words = stuff.split()**  **return words**  **def sort\_words(stuff):** """Sorts the words and turns string into a list."""# ['All', 'come', 'good', 'things', 'those', 'to', 'wait.', 'who'**]**  **sorted\_words = sorted(stuff)**  **return sorted\_words**  **def first\_words(stuff):**  **print "The first word is %s" %stuff.pop(0)** ‘’’Prints the first word after popping it off; **similar to stuff.pop(-1)** #prints/pops the lastword’’’  **>>> sentence = " All good things come to those who wait."**  **>>> words = split\_sentence(sentence)**  **>>> sorted\_words = sort\_words(words)**  **>>> print words**  **>>> print sorted\_words**  **>>> first\_words(words)**  Enter functions in FILE.py; >> >import python; >>> sentence = “….” >>> FILE.split\_sentence(sentence) |

|  |  |
| --- | --- |
| **>>> listEx = ['Vlad', 26, 'New York', 'NY']; listEx1 = ['f','e','c','d','a','b']; listEx[1]=101** #**['Vlad', 101, 'New York', 'NY']**  **>>> print ‘Vlad’ in listEx**  # True  >>> **print listEx.index(‘Vlad’)** # 0  **>>> print listEx[0:2]** # ['Vlad', 26]  **>>> print listEx[-1]** # NY  **>>> listEx.append('Joy')** # ['Vlad', 26, 'New York', 'NY',’Joy’]  **>>> listEx.remove('Joy')** # ['Vlad', 26, 'New York', 'NY']; same as **del listEx[4];** same as **listEx.pop(4)**  **>>> listEx.insert(2 ,'hello')** # ['Vlad', 26, 'hello','New York', 'NY']  **>>> del listEx[1]** # ['Vlad', 'hello','New York', 'NY']  **>>> print sorted(listEx1)** # ['a', 'b', 'c', 'd', 'e', 'f']  **>>> print listEx1[-3:]** # ['d', 'e', 'f'] ; same as **print listEx1[:3]** **# [‘a’,’b’,’c’]**  **>>> print len(listEx1)** # 6 , shows the number of indexes  **>>> listName = list('Fred')** # ['F','r','e','d']  **>>> listName[4:] = 'dy'** # ['F','r','e','d','d','y']  **>>> print ' ‘.join(listName)** # Freddy | |
| **x = [‘abcde’,’abc’,’abcdef’]**  **x.sort(key=len)**  **print x** #[**’abc’, ‘abcde’, ’abcdef’**] | **x = [13,75,3,35,65]**  **x.sort(reverse=True)**  **print x #[75,65,35,3,13]** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **listCustNum = [0,1,2]; listCustName = ['Bob', 'Helen', 'Mark']; listCustAge = [23,70,45] same as dictionary**  **for i in listCustNum: print '%s is %d' %(listCustName[i], listCustAge[i])** | | | | | |
| **>>> names = [‘vlad’, ‘martha]; >>> ages = [12,54]**  **>>> for i in range(len(names)): print names[i], ages[i]** | | | | | |
| **>>> names = [‘vlad’, ‘martha]; >>> ages = [12,54] >>>zip(names, ages) #[( , ),( , )]**  **>>> for name, age in zip(names,ages): print name, age** | | | | | |
| **num = [1, 2, 3, 4]**  **for number in num:**  **print "%d" %number**  **var = ["one", "two", "three", "four"]**  **for variable in var:**  **print "%s" %variable**  **mix = [1, "one", 2, "three", 4]**  **for mixed in mix:**  **print "%r" %mixed** | # to create a list of #  **num = []**  **for i in [1, 2, 3, 4]:**  **print "%d" %i** | # to create a list of #  **number = []**  **i = 0**  **while (i < 10):**  **num = input("Number: >")**  **number.append(num)**  **i += 1**  **print number**  \*\* **break (if)= stops while loop** | | # to create a list of words  **sent = []**  **for word in ['one', 'two', 'three']:**  **sent.append(word)**  **print sent** | |
| # to create a list  **num = []**  **for i in range(1, 5, 1):**  **num.append(i)**  **print num** |
| **The truth terms:** and, or, not, not equal (!=), equal (==), >= (greater-than-equal), <= (less-than-equal), True, False; **Find each not and invert it**  **x=y=[1,2,3], z=[1,2,3] # x is y true, even though x==z but x is not z because it tests for identity not equality**   |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **OR** | **T?** | **AND** | **T?** | **NOT OR** | **T?** | **NOT AND** | **T?** | **!=** | **T?** | **=** | **T?** | **NOT** | **T** | | T or F | T | T and F | F | not (T or F) | F | not (T and F) | T | 1!=0 | T | 1==0 | F | not F | T | | T or T | T | ***T and T*** | ***T*** | not (T or T) | F | **not (T and T)** | **F** | 1!=1 | F | 1==1 | T | not T | F | | F or T | T | F and T | F | not (F or T) | F | not (F and T) | T | 0!=1 | T | 0==1 | F |  | | | ***F or F*** | ***F*** | F and F | F | ***not (F or F)*** | ***T*** | not (F and F) | T | 0!=0 | F | 0==0 | T |   . | | | | | |
| **people = 20; cats = 30; dogs = 15**  **if people < cats:**  **print "Too many cats! The world is doomed!"** #prints "Too many cats! The world is doomed!"  **if people < dogs:**  **print "The world is drooled on**!" #does not print anything  **dogs += 5**  **if people <= dogs:**  **print "People are less than or equal to dogs**." #prints "People are less than or equal to dogs."  **if people == dogs:**  **print "People are dogs."** #prints "People are dogs." | | | **if x < 0:     print 'Negative' else:     if x == 0:         print 'Zero'     else:         print 'Positive'** | | **if x < 0:     print 'Negative' elif x == 0:     print 'Zero' else:     print 'Positive'** |
| **name = raw\_input(‘What is your name? ’)**  **if ‘s’ in name: print “name contains s”**  **else: print “name doesn’t contain s”** | | |
| **Assertion 🡪 if not condition: crash program; checkpoint; use assertions when something must be true to work correctly**  **>>>age = 10 >>>assert 0 < age < 0; >>>age = -1 >>>assert 0<age<100 ERROR** | | | | | |
| **for i in range(1,31):** # Function prints all odd numbers using **break** and **continue**  **if(i%2) == 0: continue** # Skips else statement  **elif i == 25: break**  **else: print '%d' %i, # 1 3 5 7 9 11 13 15 17 19 21 23** | | | | | |
| **while True:** #program terminates when <enter> is pressed  **word = raw\_input()**  **if not word: break #any if statement to stop the while loop** | | | | | |
| **Pass statement is added to an unfinished block**  **if(statement): pass** | | | | | |

|  |
| --- |
| **>>> items = [(‘name', 'Zed'), ('age', 36), ('height', 6\*12+2)]**  **print dict(items)** #{'name': 'Zed', 'age': 36, 'height': 74} **<-- dict('name'= 'Zed', 'age'= 36, 'height'= 74)**  **>>> stuff = {}**  **stuff[‘name’] = ‘Zed’** # {‘name’:’Zed’}  **>>> stuff = {'name': 'Zed', 'age': 36, 'height': 6\*12+2}** # pair of a key and value  >>> **len(stuff)** # 3  >>> **‘name’ in stuff**  # True, checks whether there is a name in the stuff; **same as stuff.has\_key(’name’)**  **>>> print stuff['name']** # Zed  **>>> stuff[1] = "Wow"** # adds an item to the list  **>>> print stuff.items()** # [{('name': 'Zed'), (1: 'Wow'), ('age': 36),( 'height': 74)}]  **>>> print stuff.values()** # [‘Zed’, ‘Wow’, ‘36’, 74"]  **>>> del stuff['name']** # deletes an item; **same as stuff.pop("Height"); stuff.popitem()** pops an arbitrary item  # key,value = **stuff.popitem(); key = ‘name’, value = ‘Zed’**  >>> **stuff.clear()** # empties dictionary and returns None or {}  >>> **item = stuff.copy()** # copies stuff to item  >>> **print stuff.get('name')** # Zed, if an item is not in the dict **print stuff.get('weight')** or **print stuff.get('weight', None)** prints **None**  **>>> city = stuff.get('TX', 'Sorry, nothing’)**  **print city** # Sorry, no Texas  >>> **item = stuff.get('')**  **if not item: print "No %s in the dict" % item**  **else: print item**  **>>> while True:**  **item = raw\_input()**  **print item, items.get(item)**  **if not item: break** |
| **>>> items = [(‘name', 'Zed'), ('age', 36), ('height', 6\*12+2)]**  **>>> for key in items:**  **print key, items[key]** |
| **>>>states = {'New York State':'NY', 'New Jersey State':'NJ'}**  **>>>cities = {'NY':'New York City', 'NJ':'Jersey City'}**  **>>>print “%s” % cities[states['New York State']]**  # same as print cities['NY']  **>>>for state,abbrev in states.items(): print "%s, %s, %s" %(state,abbrev,cities[abbrev])** |
| **info = {'New York State':{'abbrev':'NY','city':'New York City'}}**  **labels = {'abbrev':'abbreviation', 'city':'city'}**  **state = raw\_input("Enter state: ")**  **option = raw\_input("Enter a for abbreviation, enter c for city ")**  **if option == 'a': print "%s's %s is %s" %(state,labels['abbrev'],info[state]['abbrev'])**  **if option == 'c': print "%s's %s is %s" %(state,labels['city'],info[state]['city'])** |
| **>>>phonebook = {‘Vlad’ = ‘1234’}**  **>>>print “Vlad’s phone number is %(Vlad)s” % phonebook**  **\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **>>> template = “<html><head><title>%(title)s</title><body>%(text)s</body></head></html>”**  **>>> data = {‘title’:’Home Page’,’text’:’My first webpage!’}**  **>>> print template % data** |

|  |  |
| --- | --- |
| **>** python python.py 1st 2nd  **from sys import argv**  **script, var1, var2 = argv**  **print "%s %s %s", %(script,var1,var2)** # prints python.py 1st 2nd | **var1 = raw\_input()** > 1st  **var2 = raw\_input()** > 2nd  **print "Var1 is %s\nVar2 is %s" %(var1, var2)** |

|  |
| --- |
| **close** -- Closes the file; **read** -- Reads the file; **readline** -- Reads just one line of a text file; **truncate** -- Empties the file; **write** -- Writes to file. |
| **.**   |  |  |  |  | | --- | --- | --- | --- | | > python python.py sample.txt  **from sys import argv**  **from os.path import exists**  **script, file = argv**  **print exists(file)** #True/False  **print len(file)** #bytes  **file\_open = open(file)**  **print file\_open.read()**  **file\_open.close** | **file = raw\_input()**  > python sample.txt  **file\_open = open(file)**  **print file\_open.read()** | > python python.py sample.txt  **from sys import argv**  **script, file = argv**  **file\_open = open(file,'w')**  **file\_open.truncate()** *#truncates file*  **file\_open.write("Line1" + "\n" + "Line2")**  **file\_open.close()** | > python python.py sample\_copy.txt  **from sys import argv**  **script, file1, file2 = argv**  **file1\_open = open(file1)**  **file1\_content = file1\_open.read()**  **file2\_open = open(file2, 'w')**  **file2\_open.truncate()**  **file2\_open.write(file1\_content)**  **file1\_open.close()**  **file2\_open.close()** |   **.** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **.**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **print\_two("Zed","Shaw"); print\_two\_again("Zed","Shaw"); print\_one(12 + 6); print\_none()** | | | | | | **def print\_two(\*args):**  **arg1, arg2 = args**  **print "arg1: %r, arg2: %r" % (arg1, arg2)**  #prints arg1: “Zed”, arg2: “Shaw” | **def print\_two\_again(arg1, arg2):**  **print "arg1: %r, arg2: %r" % (arg1, arg2)**  #prints arg1: “Zed”, arg2: “Shaw” | | **def print\_one(val):**  **print "val is %d" % val**  #prints val is 18 | **def print\_none():**  **print "I got nothin'."**  #prints Zed | | **def print\_file(f):**  **print f.read()**  **def rewind(f):**  **f.seek(0)**  **def print\_line(f):**  **print f.readline()**  **def close\_file(f):**  **f.close()**  **file = open(raw\_input("Enter file name: >"))**  **print\_file(file)**  **rewind(file)** #rewinds a file, sets cursor to 0 bytes  **print\_line(file)** #prints a line of the file | | **def secret\_formula(started):**  **jelly\_beans = started \* 500**  **jars = jelly\_beans / 1000**  **crates = jars / 100**  **return jelly\_beans, jars, crates**  **start\_point = 10000**  **start\_point = start\_point / 10**  **print "We'd have %d beans, %d jars, and %d crates." % secret\_formula(start\_point)** | | | | **def subtract(x,y):**  **return x - y**  **print "1st # - 2nd # = %d" % subtract(input("Enter 1st #: "), input("Enter 2nd # "))** | | |   **.** |

|  |  |
| --- | --- |
| Example of **module** named mystuff.py that contains function apple:  # This goes in mystuff.py:  **def apple():**  **print "I AM APPLES!"**  **tangerine = "Living reflection of a dream"**  # Access function and tangerine through python:  >>>import mystuff >>>mystuff.apple() >>>print mystuff.tangerine | |
| A **class**is a way to take a grouping of functions and data and place them inside a container so you can access them with the '.' (dot) operator.  class MyStuff(object):  def \_\_init\_\_(self):  self.tangerine = "And now a thousand years between"  def apple(self):  print "I AM CLASSY APPLES!" | |
| **Example Class 1:**  **class Song(object):**  **def \_\_init\_\_(self, lyrics):**  **self.lyrics = lyrics**  **def sing\_me\_a\_song(self):**  **for line in self.lyrics:**  **print line**  **happy\_bday = Song(["Hello World"])**  **happy\_bday.sing\_me\_a\_song()** | **class Sent():**  **def printing (self, lyrics):**  **for i in lyrics:**  **print i,**  **c = Sent()**  **sent =['Hello','World','Vlad','Efros']**  **c.printing(sent)** |
| **class X(Y) : "Make a class named X that is-a Y."**  **class X(object): def \_\_init\_\_(self, J) : "class X has-a \_\_init\_\_ that takes self and J parameters."**  **class X(object): def M(self, J) : "class X has-a function named M that takes self and J parameters."**  **foo = X() : "Set foo to an instance of class X."**  **foo.M(J) : "From foo get the M function, and call it with parameters self, J."**  **foo.K = Q : "From foo get the K attribute and set it to Q."** | |

|  |  |
| --- | --- |
| **class Parent(object):**  **def override(self):**  **print "PARENT override()"**  **def implicit(self):**  **print "PARENT implicit()"**  **def altered(self):**  **print "PARENT altered()"**  **class Child(Parent):**  **def override(self):**  **print "CHILD override()"**  **def altered(self):**  **print "CHILD, BEFORE PARENT altered()"**  **super(Child, self).altered()**  **print "CHILD, AFTER PARENT altered()"**  **dad = Parent()**  **son = Child()**  **dad.implicit()**  **son.implicit()**  **dad.override()**  **son.override()**  **dad.altered()**  **son.altered()** | **Output:**  PARENT implicit()  PARENT implicit()  PARENT override()  CHILD override()  PARENT altered()  CHILD, BEFORE PARENT altered()  PARENT altered()  CHILD, AFTER PARENT altered() |

|  |
| --- |
| **Module is group of functions. You can import from module you created such as “>>> import finance import addTax, calculateDiscount”**  **If modules are in the different folders: Turn folder into the package(folder) with lots of files so than you can import folder with lots of modules in it**  **Create a special python final in the folder which tells python it’s a package. Ex: create a scrip.y fiel and subdirectory folder. Subdirectory will have many mymodule.py files in addition to \_\_init\_\_.py file that can be completely empty. \_\_init\_\_.py (can be used for setting things up or keep it completely blank so it would not do anything) is run before all other files so it turns subdirectory into a package.**  **In a script file type:**  **import subdir.mymodule (now we have access to that module)**  **subdir.mymodule.greeting()**  **In mymodule type:**  **def greeting():x**  **print “Hello World!”**  **Run script by typing in the terminal: python script or type python and >>> import script** |
| **>>> import math**  **>>> math.sqrt(49)**  **>>>s = math.sqrt**  **>>> s(49)** |
| **from random import randint**  **#print "%d" %randint(100, 999)** |
| **def random\_with\_N\_digits(n):**  **range\_start = 10\*\*(n-1)**  **range\_end = (10\*\*n)-1**  **return randint(range\_start, range\_end)**  **print random\_with\_N\_digits(2) #75**  **print random\_with\_N\_digits(3) #567**  **print random\_with\_N\_digits(4) #6545** |
| **foo = ['a', 'b', 'c', 'd', 'e']**  **from random import choice**  **print choice(foo)**  **>>> users = [‘John’, ‘Joe’,’Jim’] # prints one of the three names randomly**  **>>> random.choice(users)** |
| **>>> Import random**  **>>> Random.randint(0,5) # Picks any number from 0 to 5**  **For decimal random: >>> random.random()**  **For decimal random between 0 and 100: >>> random.random() \* 100**  **>>> r = random.random**  **>>> r()** |
| **>>> import urllib2**  **>>> urllib2.urlopen(“http://glavelle.co.uk”).read(100) # accepts string of a website address, sends HTTP request for 100 characters** |
| **>>> import datetime**  **>>> import time # Both are related to each other**  **>>> from datetime import date**  **>>> time.time() # returns current time from, not user readable**  **>>> date.fromtimestamp(34543654) #date from that many sec**  **>>> date.fromtimestamp(time.time()) # get current time**  **>>> currentDate = date.fromtimestamp(time.time())**  **>>> currentDate # datetime.date(2011, 7, 3)**  **>>> currentDate.strftime(“%d/%m/%y”) or (“%d-%m-%y”) or (“%d:%m:%y”) or (“%d-%m”)**  **>>> currentDate.isoformat() #‘2011-07-09’** |
| **>>> import os**  **>>> from os import path # useful for loading file from your server, load images to display, useful to get paths**  **>>> path.exists(“/users/Vladimir”) #True or False**  **>>> path.getatime(“/Users/Vladimir”) # Tells how many seconds ago specific path was accessed**  **>>> path.getmtime(“/Users/Vladimir”) # Tells the last time the file was saves/modified in the user’s directory**  **>>> path.getsize(“/users/Vladimir/Desktop/mystuff”) #4096L** |
|  |

|  |
| --- |
| **class pet:**  **number\_of\_legs = 0**  **doug = pet() # an instance of the class**  **print "Doug has %s of legs" %doug.number\_of\_legs # 0, print pet.number\_of\_legs**  **doug.number\_of\_legs = 4**  **print "Doug has %s of legs" %doug.number\_of\_legs # 4, print pet.number\_of\_legs** |
| **class pet():**  **legs = 0**  **def count(self):**  **print "%d" %self.legs**  **dog = pet()**  **dog.count()**  **dog = pet()**  **dog.legs = 1**  **dog.count()**  **cat = pet()**  **cat.legs = 2**  **cat.count()** |
| **class pet():**  **def sleep(self):**  **print "zzz"**  **number\_of\_legs = 0**  **def count\_legs(self):**  **print "I have %d legs" %self.number\_of\_legs**  **class dog(pet):**  **def bark(self):**  **print "woof"**  **doug = dog()**  **doug.bark() #woof**  **doug.sleep() #zzz**  **doug.number\_of\_legs = 4**  **doug.count\_legs() #I have 4 legs** |
|  |
|  |
|  |
|  |
|  |
|  |