William_Chuang_Notes_on_Fundamentals_of_Python

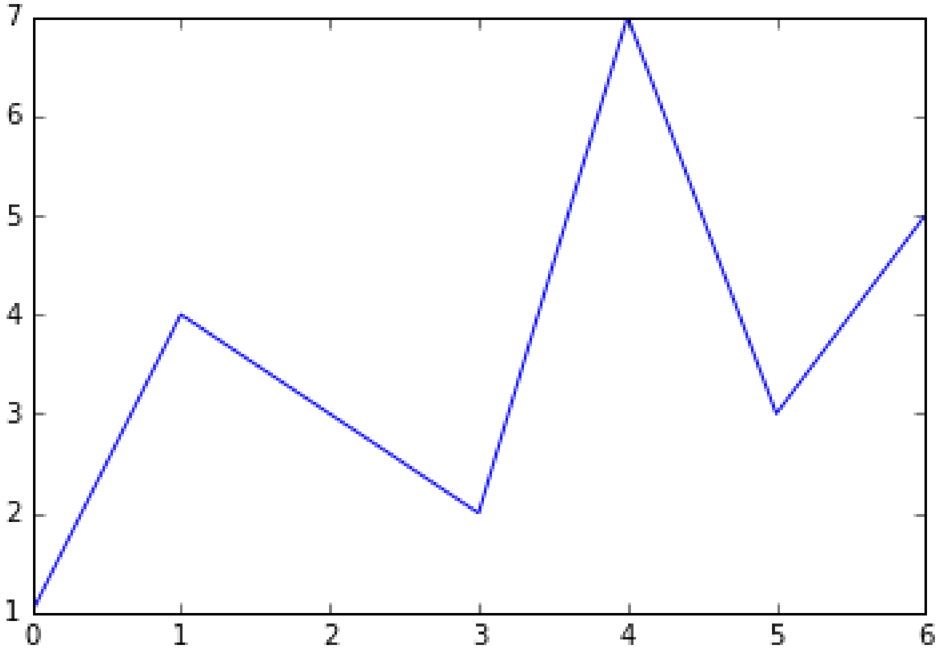
March 21, 2016

```
In [1]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

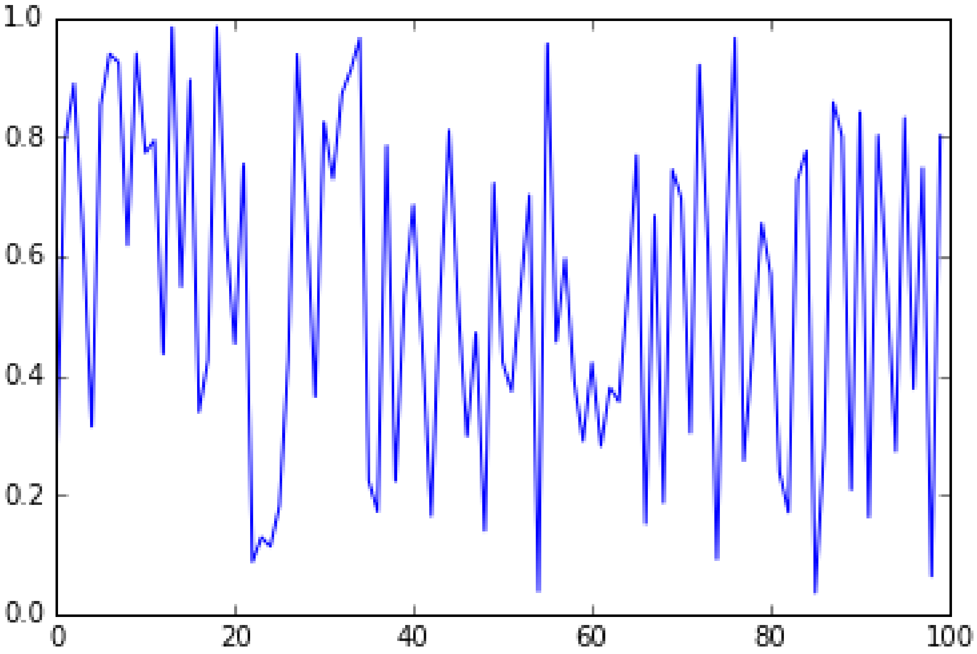
```
In [2]: plot([1,4,3,2,7,3,5])
```

```
Out[2]: [<matplotlib.lines.Line2D at 0x1063a2d10>]
```



```
In [3]: plot(rand(100))
```

```
Out[3]: [<matplotlib.lines.Line2D at 0x1065ec650>]
```



```
In [4]: x = linspace(-10,10,300)

In [14]: x

Out[14]: array([-10.        ,  -9.93311037,  -9.86622074,  -9.7993311 ,
                 -9.73244147,  -9.66555184,  -9.59866221,  -9.53177258,
                 -9.46488294,  -9.39799331,  -9.33110368,  -9.26421405,
                 -9.19732441,  -9.13043478,  -9.06354515,  -8.99665552,
                 -8.92976589,  -8.86287625,  -8.79598662,  -8.72909699,
                 -8.66220736,  -8.59531773,  -8.52842809,  -8.46153846,
                 -8.39464883,  -8.3277592 ,  -8.26086957,  -8.19397993,
                 -8.1270903 ,  -8.06020067,  -7.99331104,  -7.9264214 ,
                 -7.85953177,  -7.79264214,  -7.72575251,  -7.65886288,
                 -7.59197324,  -7.52508361,  -7.45819398,  -7.39130435,
                 -7.32441472,  -7.25752508,  -7.19063545,  -7.12374582,
                 -7.05685619,  -6.98996656,  -6.92307692,  -6.85618729,
                 -6.78929766,  -6.72240803,  -6.65551839,  -6.58862876,
                 -6.52173913,  -6.4548495 ,  -6.38795987,  -6.32107023,
                 -6.2541806 ,  -6.18729097,  -6.12040134,  -6.05351171,
                 -5.98662207,  -5.91973244,  -5.85284281,  -5.78595318,
                 -5.71906355,  -5.65217391,  -5.58528428,  -5.51839465,
                 -5.45150502,  -5.38461538,  -5.31772575,  -5.25083612,
                 -5.18394649,  -5.11705686,  -5.05016722,  -4.98327759,
                 -4.91638796,  -4.84949833,  -4.7826087 ,  -4.71571906,
                 -4.64882943,  -4.5819398 ,  -4.51505017,  -4.44816054,
                 -4.3812709 ,  -4.31438127,  -4.24749164,  -4.18060201,
                 -4.11371237,  -4.04682274,  -3.97993311,  -3.91304348,
                 -3.84615385,  -3.77926421,  -3.71237458,  -3.64548495,
                 -3.57859532,  -3.51170569,  -3.44481605,  -3.37792642,
                 -3.31103679,  -3.24414716,  -3.17725753,  -3.11036789,
                 -3.04347826, -2.97658863, -2.909699  , -2.84280936,
                 -2.77591973, -2.7090301 , -2.64214047, -2.57525084,
                 -2.5083612 , -2.44147157, -2.37458194, -2.30769231,
                 -2.24080268, -2.17391304, -2.10702341, -2.04013378,
                 -1.97324415, -1.90635452, -1.83946488, -1.77257525,
                 -1.70568562, -1.63879599, -1.57190635, -1.50501672,
                 -1.43812709, -1.37123746, -1.30434783, -1.23745819,
                 -1.17056856, -1.10367893, -1.0367893 , -0.96989967,
                 -0.90301003, -0.8361204 , -0.76923077, -0.70234114,
                 -0.63545151, -0.56856187, -0.50167224, -0.43478261,
                 -0.36789298, -0.30100334, -0.23411371, -0.16722408,
                 -0.10033445, -0.03344482,  0.03344482,  0.10033445,
                  0.16722408,  0.23411371,  0.30100334,  0.36789298,
                  0.43478261,  0.50167224,  0.56856187,  0.63545151,
                  0.70234114,  0.76923077,  0.8361204 ,  0.90301003,
                  0.96989967,  1.0367893 ,  1.10367893,  1.17056856,
                  1.23745819,  1.30434783,  1.37123746,  1.43812709,
                  1.50501672,  1.57190635,  1.63879599,  1.70568562,
                  1.77257525,  1.83946488,  1.90635452,  1.97324415,
                  2.04013378,  2.10702341,  2.17391304,  2.24080268,
                  2.30769231,  2.37458194,  2.44147157,  2.5083612 ,
                  2.57525084,  2.64214047,  2.7090301 ,  2.77591973,
                  2.84280936,  2.909699  ,  2.97658863,  3.04347826,
                  3.11036789,  3.17725753,  3.24414716,  3.31103679,
                  3.37792642,  3.44481605,  3.51170569,  3.57859532,
                  3.64548495,  3.71237458,  3.77926421,  3.84615385,
                  3.91304348,  3.97993311,  4.04682274,  4.11371237,
                  4.18060201,  4.24749164,  4.31438127,  4.3812709 ,
                  4.44816054,  4.51505017,  4.5819398 ,  4.64882943,
                  4.71571906,  4.7826087 ,  4.84949833,  4.91638796,
                  4.98327759,  5.05016722,  5.11705686,  5.18394649,
                  5.25083612,  5.31772575,  5.38461538,  5.45150502,
                  5.51839465,  5.58528428,  5.65217391,  5.71906355,
                  5.78595318,  5.85284281,  5.91973244,  5.98662207,
                  6.05351171,  6.12040134,  6.18729097,  6.2541806 ,
                  6.32107023,  6.38795987,  6.4548495 ,  6.52173913,
                  6.58862876,  6.65551839,  6.72240803,  6.78929766,
                  6.85618729,  6.92307692,  6.98996656,  7.05685619,
                  7.12374582,  7.19063545,  7.25752508,  7.32441472,
                  7.39130435,  7.45819398,  7.52508361,  7.59197324,
                  7.65886288,  7.72575251,  7.79264214,  7.85953177,
                  7.9264214 ,  7.99331104,  8.06020067,  8.1270903 ,
                  8.19397993,  8.26086957,  8.3277592 ,  8.39464883,
                  8.46153846,  8.52842809,  8.59531773,  8.66220736,
                  8.72909699,  8.79598662,  8.86287625,  8.92976589,
                  8.99665552,  9.06354515,  9.13043478,  9.19732441,
                  9.26421405,  9.33110368,  9.39799331,  9.46488294,
                  9.53177258,  9.59866221,  9.66555184,  9.73244147,
                  9.7993311 ,  9.86622074,  9.93311037, 10.        ])
```

```
In [5]: l=range(1,10)
```

```
In [20]: l
```

```
Out[20]: range(1, 10)
```

```
In [21]: l[2]

Out[21]: 3

In [17]: l2=range(1,10,2)

In [8]: l2[2]

Out[8]: 5

In [18]: for i in l2:
             print(i)


1
3
5
7
9

In [10]: for i in range(4):
             print(i)

0
1
2
3

In [19]: s = ’Hello, world.’

In [20]: str(s)

Out[20]: ’Hello, world.’

In [21]: repr(s)

Out[21]: "’Hello, world.’"

In [22]: str(1/7)

Out[22]: ’0’

In [23]: x = 10 * 3.25
         y = 200 * 200
         s = ’The value of x is ’ + repr(x) + ’, and y is ’ + repr(y) + ’...’
         print(s)

The value of x is 32.5, and y is 40000...

In [26]: for i in range(4):
             print(’The value of i is ’+repr(i))

The value of i is 0
The value of i is 1
The value of i is 2
The value of i is 3
```

```
In [27]: for x in range(1, 11):
             print(repr(x).rjust(2), repr(x*x).rjust(3), end=' ')
             # Note use of 'end' on previous line
             print(repr(x*x*x).rjust(4))
```

```
          File "<ipython-input-27-c2265e2b19af>", line 2
        print(repr(x).rjust(2), repr(x*x).rjust(3), end=' ')
                                                        ^
    SyntaxError: invalid syntax
```

```
In [40]: for x in range(1, 11):
             print('{0:2d} {1:3d} {2:4d}'.format(x, x*x, x*x*x))
```

```
1   1    1
 2   4    8
 3   9   27
 4  16   64
 5  25  125
 6  36  216
 7  49  343
 8  64  512
 9  81  729
10 100 1000
```

```
In [45]: def foo(love):
             print(love*5)
         foo("s=4")
```

```
s=4s=4s=4s=4s=4
```

```
In [47]: def formula(conversion,input):
             if conversion == "inches_to_centimeters":
                 input=float(input)
                 input = input*2.54
             elif conversion =="centimeters_to_inches":
                 input=float(input)
                 input = input/2.54
             elif conversion =="Fahrenheit_to_Celsius":
                 input=float(input)
                 input = (input-32)/1.8
             elif conversion =="Celsius_to_Fahrenheit":
                 input=float(input)
                 input = 32 + (input*1.8)
             elif conversion =="bytes_to_kilobytes":
                 input=float(input)
                 input=input/1000
             elif conversion =="kilobytes_to_bytes":
                 input=float(input)
                 input=input/1000
             elif conversion =="bytes_to_megabytes":
                 input=float(input)
                 input=input/1000000
```

```
    elif conversion =="megabytes_to_bytes":
        input=float(input)
        input=input*1000000

    return input
formula("Celsius_to_Fahrenheit",30)
```

Out[47]: 86.0

In [49]:

```
__author__ = 'williamchuang'
import turtle
import re

def setup(col, x, y, w, s, shape):
    record.write("DOWN\n")
    turtle.up()
    turtle.goto(x,y)
    turtle.width(w)
    turtle.turtlesize(s)
    turtle.color(col)
    turtle.shape(shape)
    turtle.down()

    wn.onkey(up, "Up")
    wn.onkey(left, "Left")
    wn.onkey(right, "Right")
    wn.onkey(back, "Down")
    wn.onkey(quitTurtles, "Q")
    wn.onkey(quitTurtles, "q")
    wn.onkey(quitTurtles, "Escape")
    wn.listen()
    wn.mainloop()

#Event handlers
def up():
    x=turtle.xcor()
    y=turtle.ycor()
    turtle.fd(5)
    record.write(str(x)+" "+str(y)+"\n")


def left():
    turtle.lt(5)
    x=turtle.xcor()
    y=turtle.ycor()
    record.write(str(x)+" "+str(y)+"\n")


def right():
    turtle.rt(5)
    x=turtle.xcor()
    y=turtle.ycor()
    record.write(str(x)+" "+str(y)+"\n")
```

```
def back():
    turtle.bk(5)
    x=turtle.xcor()
    y=turtle.ycor()
    record.write(str(x)+"  "+str(y)+"\n")


def quitTurtles():
    wn.bye()
    record=open("record.txt","a")
    record.close()




wn = turtle.Screen()
wn.setworldcoordinates(-300, -300, 300, 300)
record=open("record.txt","a")
setup("blue",0,0,2,2,"turtle")
```

```
In [1]: def plotRegression(data):
            import turtle
            wn = turtle.Screen()
            t = turtle.Turtle()
            t.speed(1)

            lit=[]
            x_lst=[]
            y_lst=[]
            # Set up our variables for the formula.
            for i in data:
                lit+=i.split()
                x_lst.append(float(lit[0].strip()))
                y_lst.append(float(lit[1].strip()))
                lit=[]
            print(x_lst)
            print(y_lst)

            x_sum=0
            for j in x_lst:
                x_sum+=float(j)
            x_mean=(x_sum/len(x_lst))
            y_sum=0
            for j in y_lst:
                y_sum+=float(j)
            y_mean=(y_sum/len(y_lst))
            xysum=0
            for k in range(len(x_lst)):
                xysum+=(float(x_lst[k])*float(y_lst[k]))
            xsquaresum=0
            for l in range(len(x_lst)):
                xsquaresum+=(float(x_lst[l])*float(x_lst[l]))
            m=(xysum-len(x_lst)*x_mean*y_mean)/(xsquaresum-len(x_lst)*x_mean*x_mean)
            ymin=y_mean+m*(float(min(x_lst))-x_mean)
            ymax=y_mean+m*(float(max(x_lst))-x_mean)
```

```
    # Get min and max values for coordinate system.
    x_min, x_max, y_min, y_max = float(min(x_lst)), float(max(x_lst)), float(min(y_lst)), float
    #print(x_min, x_max, y_min, y_max)
    # Add 10 points on each line to be safe.
    wn.setworldcoordinates(x_min-10,y_min-10,x_max+10,y_max+10)
    #t.pensize(5)
    t.up()
    for i in range(len(x_lst)):
        #t.down()
        t.setpos(float(x_lst[i]), float(y_lst[i]))
        t.dot()
        t.up()

    t.goto(float(min(x_lst)),ymin)
    t.down()
    t.goto(float(max(x_lst)),ymax)



    wn.exitonclick()

data=open("labdata.txt","r")
plotRegression(data)
data.close()
```

```
[44.0, 79.0, 78.0, 41.0, 19.0, 19.0, 28.0, 22.0, 89.0, 91.0, 53.0, 27.0, 14.0, 8.0, 80.0, 46.0, 83.0, 8
[71.0, 37.0, 24.0, 76.0, 12.0, 32.0, 36.0, 58.0, 92.0, 6.0, 7.0, 80.0, 34.0, 81.0, 19.0, 72.0, 96.0, 18
```

```
    ---------------------------------------------------------------------------

    Terminator                                Traceback (most recent call last)

    <ipython-input-1-389fb0f2c6b6> in <module>()
     57
     58 data=open("labdata.txt","r")
---> 59 plotRegression(data)
     60 data.close()


    <ipython-input-1-389fb0f2c6b6> in plotRegression(data)
     44     for i in range(len(x_lst)):
     45         #t.down()
---> 46         t.setpos(float(x_lst[i]), float(y_lst[i]))
     47         t.dot()
     48         t.up()


    /Users/William_Chuang/anaconda/lib/python3.5/turtle.py in goto(self, x, y)
   1774             self._goto(Vec2D(*x))
   1775         else:
-> 1776             self._goto(Vec2D(x, y))
   1777
   1778     def home(self):
    /Users/William_Chuang/anaconda/lib/python3.5/turtle.py in _goto(self, end)
   3177                                      (start, self._position),
   3178                                      self._pencolor, self._pensize, top)
-> 3179                 self._update()
   3180             if self._drawing:
   3181                 screen._drawline(self.drawingLineItem, ((0, 0), (0, 0)),


    /Users/William_Chuang/anaconda/lib/python3.5/turtle.py in _update(self)
   2658             return
   2659         elif screen._tracing == 1:
-> 2660             self._update_data()
   2661             self._drawturtle()
   2662             screen._update()                  # TurtleScreenBase


    /Users/William_Chuang/anaconda/lib/python3.5/turtle.py in _update_data(self)
   2644
   2645     def _update_data(self):
-> 2646         self.screen._incrementudc()
   2647         if self.screen._updatecounter != 0:
   2648             return


    /Users/William_Chuang/anaconda/lib/python3.5/turtle.py in _incrementudc(self)
   1290         if not TurtleScreen._RUNNING:
   1291             TurtleScreen._RUNNING = True
-> 1292             raise Terminator
   1293         if self._tracing > 0:
   1294             self._updatecounter += 1


    Terminator:
```

```
In [7]: fo = open("labdata.txt", "r")
        print("Name of the file: ", fo.name)

        # Assuming file has following 5 lines
        # This is 1st line
        # This is 2nd line
        # This is 3rd line
        # This is 4th line
        # This is 5th line

        line = fo.readline()
        print("Read Line: %s" % (line))
        line = fo.readline()
        print("Read Line: %s" % (line))
        line = fo.readline()
        print("Read Line: %s" % (line))
        # Close opend file
        fo.close()

Name of the file:  labdata.txt
Read Line: 44 71

Read Line: 79 37

Read Line: 78 24

In [11]: class Tree:
           def __init__(self, cargo, left=None, right=None):
             self.cargo = cargo
             self.left  = left
             self.right = right
           def __str__(self):
             return str(self.cargo)

In [12]: left = Tree(2)
         right = Tree(3)

In [13]: tree = Tree(1, left, right)

In [11]: tree = Tree(1, Tree(2), Tree(3))

In [12]: def total(tree):
           if tree == None: return 0
           return total(tree.left) + total(tree.right) + tree.cargo

In [14]: tree = Tree("+", Tree(1), Tree("*", Tree(2), Tree(3)))

In [15]: def printTree(tree):
           if tree == None: return
           print(tree.cargo)
           printTree(tree.left)
           printTree(tree.right)

In [16]: printTree(tree)

+
1
*
2
3

In [21]: def multadd (x, y, z):
           return x * y + z

In [22]: multadd (3, 2, 1)

Out[22]: 7

In [42]: class Point:
           def __init__(self, x=0, y=0):
             self.x = x
             self.y = y
           def __str__(self):
             return ’(’ + str(self.x) + ’,’ + str(self.y) + ’)’
           def __add__(self, other):
             return Point(self.x + other.x, self.y + other.y)
           def __mul__(self, other):
             return self.x * other.x + self.y * other.y
           def __rmul__(self, other):
             return Point(other * self.x,  other * self.y)
           def reverse(self):
             self.x , self.y = self.y, self.x
         p1 = Point(3, 4)
         p2 = Point(5, 7)
         p3 = p1 + p2
         p = Point(3, 4)
         print(p3)
         print(p1 * p2)
         print(2 * p2)
         str(p)
         print(multadd (2, p1, p2))
         print(multadd (p1, p2, 1))
```

```
(8,11)
43
(10,14)
(11,15)
44
```

```
In [40]: def frontAndBack(front):
           import copy
           back = copy.copy(front)
           back.reverse()
           print(str(front) + str(back))
         myList = [1, 2, 3, 4]
         frontAndBack(myList)
```

```
[1, 2, 3, 4][4, 3, 2, 1]
```

```
In [43]: p = Point(3, 4)
         frontAndBack(p)
```

```
(3,4)(4,3)
```

```
In [44]: tel = {’jack’: 4098, ’sape’: 4139}
         tel[’guido’] = 4127
```

```
In [45]:  tel
```

```
Out[45]: {’guido’: 4127, ’jack’: 4098, ’sape’: 4139}
```

```
In [16]: del tel[’sape’]
         tel[’irv’] = 4127
         tel
```

```
        ---------------------------------------------------------------------------

        NameError                                 Traceback (most recent call last)
        <ipython-input-16-aa37d1dc8b9c> in <module>()
    ----> 1 del tel['sape']
          2 tel['irv'] = 4127
          3 tel


        NameError: name 'tel' is not defined
```

```
In [47]: list(tel.keys())
```

```
Out[47]: ['irv', 'guido', 'jack']
```

```
In [48]: sorted(tel.keys())
```

```
Out[48]: ['guido', 'irv', 'jack']
```

```
In [49]: 'guido' in tel
```

```
Out[49]: True
```

```
In [50]: 'jack' not in tel
```

```
Out[50]: False
```

```
In [51]: knights = {'gallahad': 'the pure', 'robin': 'the brave'}
```

```
In [53]: for k, v in knights.items():
             print(k, v)
```

```
gallahad the pure
robin the brave
```

```
In [55]: while True:
              try:
                  x = int(input("Please enter a number: "))
                  break
              except ValueError:
                  print("Oops!  That was no valid number.  Try again...")
```

```
Please enter a number: as
Oops!  That was no valid number.  Try again...
Please enter a number: 1
```

```
In [97]: class Node:
           def __init__(self, cargo=None, next=None):
             self.cargo = cargo
             self.next  = next
           def __str__(self):
             return str(self.cargo)
           def print_backward(self):
             if self.next != None:
                 tail = self.next
                 tail.print_backward()
             print(self.cargo, end=' ')
```

```
In [66]: node = Node("test")
         print(node)
```

```
test
```

```
In [90]: node1 = Node(1)
         node2 = Node(2)
         node3 = Node(3)
         node1.next = node2
         node2.next = node3
```

```
In [88]: def print_list(node):
           while node != None:
             print(node, end=’ ’)
             node = node.next
           print()
```

```
In [91]: print_list(node1)
```

```
1 2 3
```

```
In [92]: def print_backward(list):
             if list == None: return
             head = list
             tail = list.next
             print_backward(tail)
             print(head,end=’ ’)
```

```
In [93]: print_backward(node1)
```

```
3 2 1
```

```
In [94]: def removeSecond(list):
             if list == None: return
             first = list
             second = list.next
             # make the first node refer to the third
             first.next = second.next
             # separate the second node from the rest of the list second.next = None
             return second
```

```
In [95]: removed = removeSecond(node1)
         print_list(removed)
```

```
2 3
```

```
In [96]: print_list(node1)
```

```
1 3
```

```
In [98]: class LinkedList:
             def __init__(self):
                 self.length = 0
                 self.head = None
             def print_backward(self):
                 print("[", end=’ ’)
                 if self.head != None:
                     self.head.print_backward()
                 print("]", end=' ')
             def addFirst(self, cargo):
                 node = Node(cargo)
                 node.next = self.head
                 self.head = node
                 self.length = self.length + 1
```

```
In [99]: class Stack :
             def __init__(self):
                 self.items = []
             def push(self, item):
                 self.items.append(item)
             def pop(self):
                 return self.items.pop()
             def is_empty(self):
                 return (self.items == [])

         s = Stack()
         s.push(54)
         s.push(45)
         s.push("+")
```

```
In [100]: while not s.is_empty():
              print(s.pop(), end=' ')
```

```
+ 45 54
```

```
In [101]: import string
          "Now is the time".split(" ")
```

```
Out[101]: ['Now', 'is', 'the', 'time']
```

```
In [15]: import re
         re.split("([^0-9])", "123+456*/")
```

```
Out[15]: ['123', '+', '456', '*', '', '/', '']
```

```
In [108]: #In this code:

          class A(object):
              def __init__(self):
                  self.x = 'Hello'

              def method_a(self, foo):
                  print(self.x + ' ' + foo)
          #... the self variable represents the instance of the object itself. Most object-oriented lan

          a = A()               # We do not pass any argument to the __init__ method
          a.method_a('Sailor!') # We only pass a single argument
          #The __init__ method is roughly what represents a constructor in Python. When you call A() Py
```

```
Hello Sailor!
```

```
In [14]: import numpy
         import gzip
         import six.moves.cPickle
         # Load the dataset
         f = gzip.open('mnist.pkl', 'rb')
         train_set, valid_set, test_set = cPickle.load(f)
         f.close()
```

```
---------------------------------------------------------------------------

NameError                                 Traceback (most recent call last)

<ipython-input-14-0de1d2a4b8f4> in <module>()
  4 # Load the dataset
  5 f = gzip.open('mnist.pkl', 'rb')
----> 6 train_set, valid_set, test_set = cPickle.load(f)
  7 f.close()


NameError: name 'cPickle' is not defined
```

```
In [ ]:
```