lec1

August 12, 2016

In [ ]:
```
#You need a language to communicate with a computer just like we have difer
#languages to communicate with each other
#Different languages have different rules: grammar rules in English are di
#Can you give me an example of rules that are different in english and Amha
#The same way there are different langauges to communicate with computers
#Each language has their own rules
#But there are some things common to every language. For example, both Engl
#Words are something common to all languages.
#In this class we will learn about
```

In [298]:
```
#Basic data types: int, long, float, bool, str, list.
#Uncomment these one at a time to see what data type they are
#Uncomment means remove # from the line
#type(5)
#type(5.0)
#type('5')
#type(True)
#type(False)
#type('My name is Timnit')
#type([1,2,3,4])
type(['timnit','5',6,True])
```

Out[298]:
```
list
```

In [305]:
```
#Operators for numerical data types (int,long,float)
#+ adds 2 values just like math
#What are other mathematical operators you know? (I just gave you a hint
5+2

#- subtract 2 values
5-2

#* Multiply 2 values
#5*2

#/ divide two values, rounds down for ints
#5.0/2.0
#5.0/2
```

```
            #%"mod", remainder after division
            #5%2

            #**exponentiation
            #5**2

            #division with floor (rounds down to integer value)
            5.0//2.0
            #5//2
```

Out[305]: 2.0

```
In [311]: #Operators and indexing for strings
          #'hi' + ' there'
          #'hi'*3
          #'abcdefg'[3]
          #'abcdefg'[3:5]
```

Out[311]: 'def'

```
In [317]: #Operators for bools (True,False)
          #and, or, not
          #True and True
          #True and False
          #False and True
          #False and False
          #----------------
          #True or True
          #True or False
          #False or True
          #False or False
          #-----------------
          #not True
          #not False
```

Out[317]: True

```
In [323]: #Operators for lists
          #+ concatenates two lists
          #'5' #What type is this str,bool,int
          #[5] + [2]
          #* repeat a list several times (list*int or int*list)
          #[1,2]*3
          #[i] gives ith element in the list
          #['a','b','c']
          #What about ['a','b','c'][1]
          #[i : j] means take sublist from element i to j - 1
          #['a','b','c','d','e'][1:3] #Can you guess what this will be?
```

Out[323]: ['b', 'c']

In [326]:
```
#Comparison operators
#5 == 2 #==True if a equals b. What is the answer?
#5 != 2 #!= True if a doesn't equal b asnwer? #space between equal and !
#5 <> 2 #<> Same as ! =
#5 > 2  #> True if a is greater than b
#5 < 2  #< True if a is less than b
#5 >= 2 #>= True if a is greater than or equal to b. Answer?
#5 <= 2 #True if a is less than or equal to b
```

```
  File "<ipython-input-326-42c420479d35>", line 3
5 ! = 2 #!= True if a doesn't equal b asnwer? #space between equal and ! gi
  ^
SyntaxError: invalid syntax
```

In [332]:
```
#Assignment Operators
#Left hand side is a variable, call it "x", and right hand side is an exp

#=Sets x to the result of b
x = 5 + 2
#x #or print x

#+= Sets x to x+b
#x= x+5+2
x += 5 + 2 #a space between + and = gives error
x

#-=Sets x to x-b
x=x-(5+2)
x -= 5 + 2 #a space between - and = gives error
x

#*=Sets x to x*b
x=x*(5+2)
x *= 5 + 2 #Guess what this does
#x

#/=Sets x to x/b
x=x/(5+3)
#x /= 5 + 3 #Same as x=x/(5+3)
#x

#%=Sets x to x%b
x=x%(5+2)
```

```
#x %= 5 + 2 #x=x% (5+2) #x%5+2 gives something different what?
#x

#** =Sets x to x**b
#x **= 2 #x=x**2
#x

#//= Sets x to x//b
#x//= 5 + 2
#x
```

Out[332]: 7

In [334]:
```
#Operator Precedence, from highest precedence to lowest
#(remember you can override operator precedence using parentheses)
#Operation
#**
#* / % //
#+-
#<= < > >=
#<>== !=
#= %= /= //= -= += *= **=
#not or and

#not 0 and 0  #Vs. not (0 and 0)
#2**5+2 #vs 2**(5+2) vs (2**5)+2

not True and False
```

Out[334]: True

In [359]:
```
#We define functions to do different things for us.
#For example the following function adds 2 to any number

def addTwo(n):
    n = n+n
    return n
#return n+2
#addTwo(3)
#addTwo('abc')
#addTwo([1,2,3])
X=True
x=False
print X
print x
```

```
True
False
```

```
In [236]: #We define functions to do different things for us.
          #For example the following function adds 2 to any number
          def addTwo(n):  #<----syntax, need colon. What happens if we don't have t
              #return n+2 #<----what happens if we don't return n+2? Just have n +=
              #n +=2     same as return n+2
              #return n

In [237]: addTwo(5) #What do you think this would give us?

Out[237]: 7

In [362]: #Note: variables can point to functions
          x=addTwo
          x
          print x(2) #What is the output here?
          print x()  #What happens here.
          print x(2,3)

4


        ---------------------------------------------------------------------------

        TypeError                                 Traceback (most recent call last)

        <ipython-input-362-9348d1d03f79> in <module>()
          3 x
          4 print x(2) #What is the output here?
    ----> 5 print x()  #What happens here.
          6 print x(2,3)


        TypeError: addTwo() takes exactly 1 argument (0 given)
```