Mike Taatgen

DPW 2

Final

Assignment operators:

|  |  |
| --- | --- |
| x += y | x = x + y |
| x -= y | x = x - y |
| x \*= y | x = x \* y |
| x /= y | x = x / y |
| x %= y | x = x % y |

Logical operators:

&&

||

Else

Elif

* Loops:
  + For x in range(a,b)
  + For x in something:
* Classes
  + Class Base(object):

def \_\_init\_\_(self)

* + class ChildB(Base):

def \_\_init\_\_(self):

super(ChildB, self).\_\_init\_\_()

* MVC
  + Models: In the model you put your communication to get the data
  + Views: In the views you put all your information that displays the page which is often the page.py
  + Controlller: The rest
* Traversing:
  + import urllib2 #Needed for importing from URL's
  + from xml.dom import minidom #convert XML into an object
  + self.request.GET[‘Something’] # Takes that something from the url and stores it
  + url = "http://something.com=? #url we are going to load the page from
  + reg = urllib2.Request(url + zip) #concat zip with the url and format as request
  + opener = urllib2.build\_opener() #magic to load request - creates framework to get url
  + result = opener.open(reg) # gets url and puts result in "result"
  + xmldoc = minidom.parse(result) #parse through string to get XML object
  + self.response.write(xmldoc.getElementsByTagName('title')[2].firstChild.nodeValue)
* Traversing from an .xml file
  + self.\_\_xmldoc = minidom.parse(open('???.xml', 'r'))
  + self.\_\_?? = self.\_\_xmldoc.getElementsByTagName('???')
* Remember to write docstrings and a lot of commenting for the exam
* Format Locals
  + def update(self):

self.\_something = self.\_something.format(\*\*locals())

* to get the value out of the XML you would write getElementsByTagName(‘something’)[0].firstChild.nodeValue
* Make sure to pass it list of items in init and not in the class self
* Pass it into the class instantiating and remember to make a do function which returns the content and call it out as an attribute like page.do(somethings)