DSC 10 Programming Basics Week 1

Python Visualizer

http://pythontutor.com/visualize.html#mode=edit

Check out this tool if you want to visualize how Python executes your code line by line. This program was created by UCSD professor Philip Guo, and this will be a recommended helper program for DSC 20, but you might find it very useful also in DSC 10.

Import Statements

```
from datascience import *
import numpy as np
```

These two lines of code will be at the beginning of most, if not all, of your homework assignments. Their purpose is to import the datascience and numpy libraries into the environment so you can use the functions from those libraries. The asterisk (*) means “all”, so you are importing “all” of the datascience library, and when you use functions from this module, you don’t need to include the module name. So instead of datascience.make_array(3, 5, 7), you can simply type make_array(3, 5, 7). For the numpy package, importing it “as np” means that instead of numpy.arange(10), you can use np.arange(10). We’ll provide the import statements you need for this class, but since they’ll pop up so often, it wouldn’t hurt to know what they’re doing. :)

Expressions

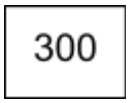
- Python operators (+, -, *, **, /, etc.) follow the typical order of precedence.
- When multiple operators of the same precedence appear in an expression, the expression is evaluated in a certain order. Most of the time, it will be from left to right, but multiple exponents (**) is a special case and they’re evaluated from right to left.
- The modulus operator (%) is useful and might show up in future classes. It gives the remainder of the division of two numbers and it has the same precedence as multiplication and division. ex.) 9%4 = 1, 4%2 = 0
- Using the regular division operator (/) always gives the answer in a float, even if the operands are integers.

Built-in Functions

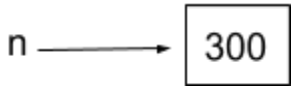
- These functions can be invoked by calling them and putting the right inputs as parameters. Inputs to the function have to match the function's parameters in both data type and order.
- The parameters are shown between parentheses right after the function name.
  ex.) param = 1.0
  int(param)
  "int" is the function name and it takes a number or string as a parameter, and converts it to an integer.
- Be careful not to use the name of a function as a variable name. For example, if you want to remember the maximum of several numbers, don't store it in a variable called "max," since "max" is already the name of a function in Python.
- If you ever forget what the parameters of a function are, you can Google the Python documentation for it, or use the question mark or help features. Soon you'll be learning how to define your own functions!

Variables and Data Types

- When you create a variable, you assign a certain value to a certain name. That value is an object and the name you assign it is the object reference. Python is an object-oriented programming language where every item of data is an object with a data type or class.
- >>> 300

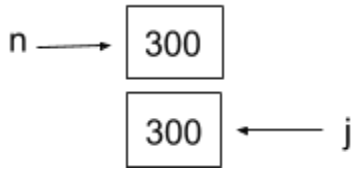
  

  300 is an integer object. In general, think of an object as a box in memory with the value in it. If you don't assign it to a variable there's no way to refer to that object anywhere else. So there is no way to refer to the number 300 unless we assign it a name.
  >>> n = 300

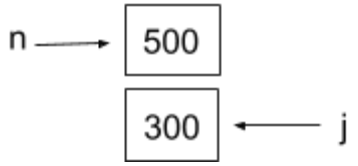
  This creates an integer object with value 300 and assigns it to 'n' (variable 'n' now points to that object). You can now use 'n' to refer to 300.

- >>> j = n



This defines a new variable ‘j’, which has the same value as n, 300.

- >>> n = 500



A new integer object with value 500 is created and ‘n’ is a variable that refers to 500 now. Note that ‘j’ still refers to 300.

- An object stays in memory as long as there is a reference to it.
- Remember that strings are objects and using the print() function only displays whatever printable value/object you put in it as a parameter.

Sequences

- Each object in an array has an index. It is important to remember that the index of the first object is 0. You can access a specific element of an array using the command .item(), where you use the index of the element you want as a parameter. For example a.item(0) gives the first element of the array named “a.”

Extra: Array slicing

- Slicing is a way to get specific subsets of an array. The item command gives you one element of an array, and slicing gives you several consecutive elements.
- >>> a = make_array(1, 7, 3, 9, -11, 14, 2)
- >>> a[1:4]
  This returns [7, 3, 9] which is a sub-array with the elements of array ‘a’ from index range 1 to 3. The result doesn’t include the element at the last index specified, 4 in this case.
- >>> a[3:]
  This returns [9, -11, 14, 2]. If the end of range is not specified, the sub-array will have elements from the specified index to the rest of the array.
- >>> a[:4]
  This returns [1, 7, 3]. If the beginning of the range is not specified, the sub-array will have elements from the beginning of the original array up to but not including the specified index.

Practice Problems

1. What is the value of x after running each of these pieces of code? Does it return an error, and if so, what is the problem?
    a. x = 2**2**3
    b. x = 3
       x = (x + 2) * x
    c. a = 11
       x = a / 2 + a % 2
    d. x = 2 / 0
2. Fix this code with an error so that x's value is 3.
       x = int("1"+"2.0")
3. What is the data type of the elements in this array? Will this code cause an error?
       a = make_array(5, '2')
4. Is there an error? Try to fix it if there is.
    a. int(str(math.pi))
    b. str(int(44.5))
    c. a = make_array(1, '2', 3)
       max(a)
    d. a = make_array(5, '2')
       int(a)
5. What are the values of x and y after the code below is completed?
       x=3
       y=x+4
       x=y-2
       y=y*2+x*3
       x=x*2 -y/2 +4

   This is a good problem to try out the Python visualizer tool.
6. Suppose you have already defined two variables x and y, each with different values. How do you switch the values x and y without manually typing the numbers? (Think about the variable boxes discussed above.)
7. Given the array
       arr = np.arange(1, 2, 3)
    a. What is the value of max(arr) - min(arr)?
    b. How do you use Python to count the number of elements in "arr"?
8. Suppose you have a variable called "my_array", which is an array of integers. Assign x to the last element in "my_array".
       x = _____
9. What are the values of x and y in the code below?
       arr = np.arange(-2, -17, -3)
       x = arr.item(2)
       y=len(arr)