PLTW COMPUTER SCIENCE

Activity 1.3.6

Tuples and Lists

Introduction

A string can store letters, numbers, and anything else that can be represented by characters. If we want to know the fifth character, we look at the fifth element in the string.

Some collections have elements of varying sizes. When people stand in line to buy tickets, some of them might buy many tickets! Who will get the fourth seat? Which seat will go to the person in the bright blue shirt? Each person holds one place in line. Each person can tell us more about the seats they will get. As you will see, this is similar to a *Python*® tuple or list: each element is an object that can say more about the data it points to.

Strings, tuples, and lists are just three examples of collections of data. The power of computers comes from iterating across large collections of data. What are some large collections you can think of?

Image courtesy the Hindu ©2013

Image courtesy NASA ©2005

M81, a galaxy 12 million light years behind the Big Dipper, contains about 250 billion stars.

Copyright © 2017 Project Lead The Way, Inc. All rights reserved.

Materials

- Computer with Enthought Canopy distribution of *Python*® programming language

Procedure

1. Form pairs as directed by your teacher. Meet or greet each other to practice professional skills and establish norms.

2. Launch Canopy and open an editor window.

3. If your teacher directs you to turn in your work with an IPython log, set the working directory for the IPython session and turn on session logging.

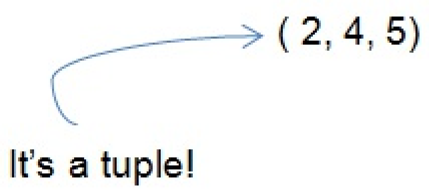
```
In []: %logstart -ort JDoeBSmith1_3_6.log
In []: # Jane Doe 1.3.6 IPython log
```

4. Start a new program in the code editor by choosing **File > New > Python file**. Save the file as `JDoe_BSmith_1_3_6.py.`

Part I: Tuples, Syntax

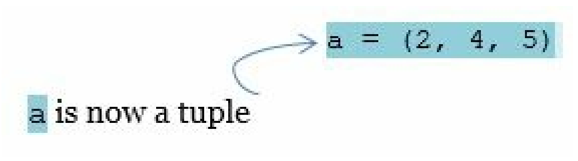
Copyright © 2017 Project Lead The Way, Inc. All rights reserved.

5. There are other iterables besides strings. One kind is called a tuple, or a sequence of values:



   Write another tuple. Be sure to put it in parentheses - that's what tells *Python* that it's a tuple!

6. One variable can be bound to an entire tuple with assignment operator =.



   Bound means that the variable name is now listed in the namespace in a table that shows variable names and memory addresses. *Python* programmers usually choose all lowercase letters and numbers for any variable name, with underscores separating the words. This is a style convention and not a syntax rule. Other conventions for making up variable names exist, like "camelCase." Employers sometimes specify conventions for their programmers to follow, and your teacher might direct you to use a particular convention for your work.

   Can you think of a convention required by one of your teachers regarding the format for submitted work?

   What convention does your teacher want you to follow for *Python* variable names?

7. Syntax is the required spelling and grammar of a computer language. *Python* syntax, for example, says that you can use an underscore but not the tab character in a variable name.

```
In []: # 7.Tuples
In []: some_values=('a','b',3)
```

   The characters in parentheses are a tuple, which is a list of values that can't change. A tuple begins and ends with parentheses: `(` and `)`. The variable `some_values` is bound to the tuple.

```
In []: some_values
Out[]: ('a', 'b', 3)
```

   Just like characters in strings of characters, the individual elements of the tuple are indexed so that you can access them one at a time. They are indexed from 0, so a 3-tuple like `('a', 'b', 3)` has elements at indices 0, 1, and 2.

   It will always work out like that: if you have ***n*** values in a tuple, then the last one is indexed ***n-1***.

   You can retrieve individual values.

```
In []: some_values[0]
```

Copyright © 2017 Project Lead The Way, Inc. All rights reserved.

```
Out[]: 'a

In []: some_values[2]
Out[]: 3
```

- Predict, try, discuss, and explain the output for the following input.

  ```
  In []: some_values[1]
  In []: # 7a. (Discuss and explain.)
  ```

- Predict, try, discuss, and explain the output for the following input.

  ```
  In []: some_values[0:2]
  In []: # 7b.(Discuss and explain.)
  ```

8. Tuples are **immutable**, which means that you're not allowed to change the elements of a tuple. You can only assign a variable that was a tuple to a whole new value, which might be another tuple.

   ```
   In []: some_values[2]= '3'
   ---------------------------------------------------------
   TypeError Traceback (most recent call last)
   <ipython-input-14-62eb70a8fd4d> in <module>()
   ----> 1 some_values[2]= '3'
   TypeError: 'tuple' object does not support item assignment
   ```

   Tuples only are bound to data, never to other int or str variables. If you use a variable expression when assigning a tuple, the expression is evaluated and the value is stored in the tuple. That value won't change, even if the expression involving the int variable has a different value later.

   ```
   In []: a = 10
   In []: b = (9, a, 11) # b was defined by using a
   In []: b
   Out[]: (9, 10, 11)

   In []: a = 15
   In []: a              # a has changed
   Out[]: 15

   In []: b              # but b has not changed
   Out[]: (9, 10, 11)
   ```

   Predict the output for the following inputs. Then try it, discuss, and explain.

   ```
   In []: b[1] == 10
   In []: # 8a. (Discuss and explain.)
   In []: b = (9, a , 11)
   In []: b[1]
   In []: # 8b. (Discuss and explain.)
   ```

Copyright © 2017 Project Lead The Way, Inc. All rights reserved.

Part II: Lists

9. If you list elements inside parentheses, *Python* creates a tuple. If you list the elements inside square brackets, *Python* creates a list. Lists are similar to tuples, but their elements can be changed, inserted, and deleted.

   ```
   In []: values = ['a','b',3]
   In []: values
   Out []: ['a', 'b', 3]
   ```

   List elements can be accessed individually, just like tuple elements.

   ```
   In []: values[1]
   Out[]: 'b'
   ```

   Predict, try, discuss, and explain the output for this input:

   ```
   In []: values[1:]
   In []: # 9. (Discuss and explain.)
   ```

10. Unlike tuples, lists are **mutable***:* you can reassign the value of individual elements in a list. Note, however, that using an element in an expression does not change its value.

    ```
    In []: values[2] == 3
    Out[]: True

    In []: values[2] + 5
    Out[]: 8

    In []: values[2]
    Out[]: 3
    ```

    Predict, try, discuss, and explain the output for this input:

    ```
    In []: values[2] = '3' # Note the quotes!
    In []: values[2] == 3
    In []: # 10. (Discuss and explain.)
    ```

11. You can add onto a list with another list or append a new element. Predict, try, discuss, and explain the output for this input.

    ```
    In []: values = values + [4, 5]
    In []: values
    In []: # 11a. (Discuss and explain.)

    In []: values.append([6, 7])
    In []: values
    In []: # 11b. (Discuss and explain.)
    ```

    If you are looking for a leap ahead, look at the official *Python* tutorial #5 at

Copyright © 2017 Project Lead The Way, Inc. All rights reserved.

https://docs.python.org/2/tutorial/datastructures.html.

12. You cannot add onto a list using anything other than a list. Try this and explain:

```
In []: values = values + 5
In []:  # 12._(Explain why this doesn't work.)
```

13. There is a shorthand for adding something to a variable and putting the result back in the variable.

```
In []: values += [5,6] # values = values + [5,6]
In []: values
Out[]: ['a','b', >'3', 4, 5, 6]
```

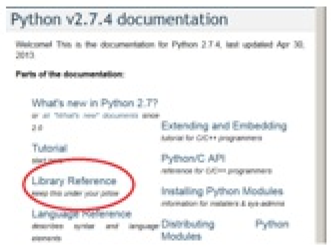
This shorthand saves valuable typing time! It works for many variable types and operations. Predict, try, discuss, and explain the output for this input:

```
In []: a = 6
In []: a *= 3
In []: a
In []:  # 13. (Discuss and explain.)
```

Part III: Lists and the random module

14. For a game, you might want a program to pick at random from a list. *Python* doesn't have a built-in way to get random or pseudo-random numbers. Pseudorandom numbers are unpredictable but not actually random. The computer can use the system clock and some fancy arithmetic to generate a pseudo-random number.

Fortunately, someone already wrote the code for generating pseudo-random numbers and shared it. You only need to `import` it from a library included with Canopy. The **module** containing these programs is called "random". It has many functions, as you can see in the "Library Reference" at http://docs.python.org/2.7. There is so much information there that it may be difficult to find what you're looking for quickly. A time-saving tip is to use the browser's "find" function to search for the keyword "random". You'll find documentation on the functions in the random module by following the link in category 9 as shown below, or just do a Google search for "python random" and choose the link to the official site.



Python v2.7.4 documentation

Welcome! This is the documentation for Python 2.7.4, last updated Apr 30, 2013.

Parts of the documentation:

What's new in Python 2.7? or all "What's new" documents since 2.0

Tutorial start here

Library Reference keep this under your pillow

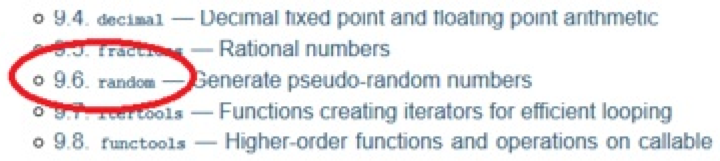
Language Reference describes syntax and language elements

Extending and Embedding tutorial for C/C++ programmers

Python/C API reference for C/C++ programmers

Installing Python Modules information for installers & sys-admins

Distributing Python Modules



- 9.4. decimal — Decimal fixed point and floating point arithmetic
- 9.5. fractions — Rational numbers
- 9.6. random — Generate pseudo-random numbers
- 9.7. itertools — Functions creating iterators for efficient looping
- 9.8. functools — Higher-order functions and operations on callable

```
In []: import random
```

Here are the three functions we want from the random module. Try each one several times to

Copyright © 2017 Project Lead The Way, Inc. All rights reserved.

view random output. Use the up arrow to enter them again.

```
In []: random.choice(values) # choose from a list
In []: random.randint(5,8) # choose int from [5,6,7,8]
In []: random.uniform(5,8) # choose float between 5 and 8
```

15. Define a function `roll_two_dice()` that simulates rolling two six–sided dice and returns the total. Pair program, strategizing first.

```
In []: roll_two_dice()
Out []: 7
```

16. Define a function `guess_letter()` that will pick one letter randomly from the alphabet.

Refer to your downloadable resources for this material. Interactive content may not be available in the PDF edition of this course.

Refer to your downloadable resources for this material. Interactive content may not be available in the PDF edition of this course.

Conclusion

1. Consider a string, tuple, and list of characters.

```
In []: a = 'abcde'
In []: b = ('a', 'b', 'c', 'd', 'e')
In []: c = ['a', 'b', 'c', 'd', 'e']
```

   The values of `a[3]`, `b[3]`, and `c[3]` are all the same. In what ways are `a`, `b`, and `c` different?

2. Why do computer programming languages almost always have a variety of variable types? Why can't everything be represented with an integer?

Copyright © 2017 Project Lead The Way, Inc. All rights reserved.