**Session 2 Notes**

**Strings and Things**

*NOTE: Before undertaking this lesson please read through the notes on Variables and Functions. Pay particular attention to the various types of variables. In this course we will be using variables of the types String, Integer, Float, Boolean and List.*

Start up your Wing IDE. In the shell enter the following list:

list\_one = [1,'hello',['a','b','c'],True]

This creates a list called list\_one. The [] define the boundaries of the list, the elements are numbered from 0 to the end of the list. To access a variable you need the element number. In the shell type in:

list\_one[0]

When you press enter the shell will give you the contents of the 0 element in the list, which is the number 1. To understand how this works look at the table below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| List contents | 1 | Hello | [‘a’,’b’,’c’] | True |
| List element number | 0 | 1 | 2 | 3 |
| Data type | Number (integer) | String | List of characters | Boolean |

List element numbering always starts at 0, each element in the list can be any type of Python variable. To access the String hello you would type in list[1].

In Python Strings are also treated as lists. Define a variable in the shell:

a = ‘hello’

Now see what a[0] gives you.

This means that our list actually contains two lists, one a String at element 1 and the other at element 2. We can access those lists as well, by first accessing the element in list, then accessing the elements inside. Type in:

list\_one[2][1]

list\_one[1][3]

This can go on forever. You can bury lists inside other lists as much as you like, it just might get hard to find them.

**Activity One:**

Create a small program using our list above and print out the following:

The Boolean True

The word hello

The list of characters [‘a’,’b’,’c’]

The letter o in hello

The letter a

**Slicing**

Often we want more than just the element in a list, especially when we are dealing with Strings. Python has a way of easily finding parts of a list or String. Create the following string:

quote=”Don't worry about the world coming to an end today. It is already tomorrow in Australia”

This is a quote from Charles M. Schulz the author of the comic strip Peanuts (<http://www.brainyquote.com/quotes/keywords/australia_2.html>). Remember we need to use double quotes for this because there is a single quote inside.

Now let’s say we want the word worry from this quote. This is what slicing does. The format is similar to getting an element. The actual code for this is:

quote[6:11]

Let’s just look at how this works. The first part of quote is

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| D | o | n | ‘ | t |   | w | o | r | r | y |   | a |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

The top row is the start of the list that makes up the variable quote, underneath is the number of each element. Splicing works by first saying where you want to start, followed by a colon and then where you want to finish. In particular note that the last position is not displayed.

quote[6:11]

End position, which is not displayed.

Starting position

Variable name

**Activity Two:**

Create a small program using our quote above and print out the following:

1. about

2. an end today

3. Australia

4. Don’t worry about the world coming to an end

The element numbering of a list starts from 0 as the leftmost element and goes on towards the right. However it also starts at the rightmost element and works backwards. In this case the number uses negative numbers. Also the end points can be reached by leaving the numbers out in slicing. To see how this works lets create another String:

wolf = “What’s the time”

To get time we could use

wolf[11:15]

wolf[11:]

wolf[-4:]

Note there is no element number 15, so Python just sees this as the end of the list. You could use wolf[11:25] and it gets the same result. The negative numbering works like this:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| W | h | a | t | ‘ | s |   | t | h | e |   | t  | i | m | e |  |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |  |
| -15 | -14 | -13 | -12 | -11 | -10 | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 |  |

Each element has two numbers for its position, a positive number starting from 0 on the left and a negative number starting from -1 on the right.

There is one more feature of slicing. Sometimes you want to move through the list getting every second or third member. To do this with slicing you add a third parameter which shows how you want to count by. For example

wolf[::2] produces Wa' h ie, by starting at 0 going to the end and only printing every second element.

Comments

Comments are parts of the code which people read but Python skips over and does not read. They are created by first typing in a hash sign #, then everything else on the line is ignored. If you want multi-line comments you can use the triple quotes ‘’’ to start and finish the comment. The triple quotes can also be used for multi-line strings.

Functions

As we learnt last lesson you use the command def to create a function. However functions have three components, **inputs, processing and outputs**. The **inputs** to the function are declared at the start as a list of variables, the **processing** is the block of the function, however last lesson we did not discuss how to get **output**s. These are what the function creates for the program and can be used directly or to create variables.

For example say I want to create a function which takes a two letter word and creates a palindrome using those letters, eg lo becomes lol, da becomes dad etc. The function to do this would look like this (printing the function to test it):

def palindrome(text):
 value = text + text[0]
 return value

print(palindrome('da'))

The new command here is **return**. This sends a value back from the function (the output of the function). If you want to return more than one value you can use separate variable separated by comma, or you could use a list or other variable type. In the other functions you can use the returned value(s) either by assigning them to a variable or directly.

In this case the returned value is used by the print statement. Note the two closing brackets, one closes the print command and one closes the palindrome command.

In fact I could have made the palindrome function shorter, I did not need to create the variable value, I could have just written

return text + text[0]

It does the same thing.

There is more to learn about function (recursion, default parameters, keyword arguments, decorators, closure etc) but this provides you with the basic way they work. This course will not go any further with functions and for 90% of your programming this will be all that you need.