**Objects and Data Structures Assessment Test**

**Test your knowledge.**

**Answer the following questions**

Write a brief description of all the following Object Types and Data Structures we've learned about:

Numbers: Consist of positive negative integer, float

Strings: sequence of character (case sensitive)

Lists: ordered

Tuples: immutable

Dictionaries:

**Numbers**

Write an equation that uses multiplication, division, an exponent, addition, and subtraction that is equal to 100.25.

Hint: This is just to test your memory of the basic arithmetic commands, work backwards from 100.25

In [ ]:

Answer these 3 questions without typing code. Then type code to check your answer.

What is the value of the expression 4\*(6 + 5)=44

What is the value of the expression 4\*6 + 5 = 29

What is the value of the expression 4 + 6 \* 5=34

In [ ]:

44,29,34

What is the *type* of the result of the expression 3 + 1.5 + 4? Float

What would you use to find a number’s square root, as well as its square?

In [x\*\*(1/2)]:

*# Square root:*

In [x\*\*2]:

*# Square:*

**Strings**

Given the string 'hello' give an index command that returns 'e'. Enter your code in the cell below:

In [ s*[1]* ]:

s = 'hello'

*# Print out 'e' using indexing*

Reverse the string 'hello' using slicing:

In [ *s[-4]* ]:

s ='hello'

*# Reverse the string using slicing*

Given the string hello, give two methods of producing the letter 'o' using indexing.

In [ ]:

s ='hello'

*# Print out the 'o'*

*# Method 1: s[-1::]*

In [ ]:

*# Method 2: s[4::]*

**Lists**

Build this list [0,0,0] two separate ways.

In [ ]:

*# Method 1: list1 = [0,0,0]*

In [ ]:

*# Method 2: list2 = [‘0’,’0’,’0’]*

Reassign 'hello' in this nested list to say 'goodbye' instead:

In [ ]:

list3 = [1,2,[3,4,'hello']]

list3[2][2] = ‘goodbye’

Sort the list below:

In [ ]:

list4 = [5,3,4,6,1]

list4.sort()

**Dictionaries**

Using keys and indexing, grab the 'hello' from the following dictionaries:

In [ ]:

d = {'simple\_key':'hello'}

*# Grab 'hello'*

*d[‘simple\_key’]*

In [ ]:

d = {'k1':{'k2':'hello'}}

*# Grab 'hello'*

*d[‘k1’][‘k2’]*

In [ ]:

*# Getting a little tricker*

d = {'k1':[{'nest\_key':['this is deep',['hello']]}]}

d['k1']['nest\_key']['this is deep']

*#Grab hello*

d['k1'][0]['nest\_key'][1]

In [ ]:

*# This will be hard and annoying!*

d = {'k1':[1,2,{'k2':['this is tricky',{'tough':[1,2,['hello']]}]}]}

d['k1'][2]['k2'][1]['tough'][2]

Can you sort a dictionary? Why or why not?

No, because by convention dictionaries are unordered mappings for storing objects.

**Tuples**

What is the major difference between tuples and lists?

Tuples are immutable.

How do you create a tuple?

my\_tuple = (‘first string’, 1, 4.5)

**Sets**

What is unique about a set?

Elements in sets are must be distinct.

Use a set to find the unique values of the list below:

In [ ]:

list5 = [1,2,2,33,4,4,11,22,3,3,2]

{1, 2, 3, 4, 11, 22, 33}

**Booleans**

For the following quiz questions, we will get a preview of comparison operators. In the table below, a=3 and b=4.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| == | If the values of two operands are equal, then the condition becomes true. | (a == b) is not true. |
| != | If values of two operands are not equal, then condition becomes true. | (a != b) is true. |
| > | If the value of left operand is greater than the value of right operand, then condition becomes true. | (a > b) is not true. |
| < | If the value of left operand is less than the value of right operand, then condition becomes true. | (a < b) is true. |
| >= | If the value of left operand is greater than or equal to the value of right operand, then condition becomes true. | (a >= b) is not true. |
| <= | If the value of left operand is less than or equal to the value of right operand, then condition becomes true. | (a <= b) is true. |

What will be the resulting Boolean of the following pieces of code (answer fist then check by typing it in!)

In [F ]:

*# Answer before running cell*

2 > 3

In [ F]:

*# Answer before running cell*

3 <= 2

In [ F]:

*# Answer before running cell*

3 == 2.0

In [ T]:

*# Answer before running cell*

3.0 == 3

In [f ]:

*# Answer before running cell*

4\*\*0.5 != 2

Final Question: What is the boolean output of the cell block below?

In [ ]:

*# two nested lists*

l\_one = [1,2,[3,4]]

l\_two = [1,2,{'k1':4}]

*# True or False?*

l\_one[2][0] >= l\_two[2]['k1']

False

**Great Job on your first assessment!**