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:

For the full answers, review the Jupyter notebook introductions of each topic!

Numbers:- There are different categories of numbers in Python. Mostly used are Integers and Floats.

Integers are numbers without decimal point. It can be either negative or positive. **For Example**: +15, -20 , +1000 . Floats are numbers with decimal point. **Example**: 15.01, -20.81 , 0.08. You can perform many operations: Addition, Subtraction, Multiplication, Power, and Division on Integers and Floats. Also, there are operations like Floor Division and Modulo. Floor division denoted as '//' returns only the integer part of the result after division. Modulo operation (%) returns the remainder after division

Strings:- Strings refers to the sequence of characters in Python. Strings are always written in quotation marks (single, double, or triple) and are Immutable. It means that the elements of string cannot be edited after definition. Different operations that can be performed on Strings are Indexing, Slicing, Concatenation, Repetition. Also, there are various in-built method in Strings such as upper(), lower(),split(),strip(),lstrip() ,rstrip() and many others that can be studied. Below are various **examples** and explanation of operations and methods in String with comments(denoted as '#' for understanding purpose).

Lists:- List is a collection of items containing data of different types. The indexing of List starts with zero. If a length of a list is three, indexes are 0,1,2. I have shared multiple **examples** of different List operations with output and comments for better understanding.

Tuples:- Tuples are immutable data types with a collection of different types of data. It does not support object assignment. They are used for passing objects without getting changed for data integrity. It is written with a small bracket. Let us see an **example** for clarity.

Note that Tuples operations are similar to the Lists. Except, Tuples are not mutable and defined with small brackets.

Dictionaries:- Dictionaries in Python are key-value pairs collection. Key and Value can be of any data type. There is a mapping of the key with the corresponding value. Keys are used to look-up for the values. Dictionaries are written within curly '{}' braces.

In the second **example** dictionary 'newd' is created and key 'India' and 'Indonesia' are added. Hence new elements can be added in the dictionary.

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

```
# Your answer is probably different
(20000 - (10 ** 2) / 12 * 34) - 19627.75
```

```
100.25
```

Explain what the cell below will produce and why. Can you change it so the answer is correct?

```
2/3
```

```
0
```

Answer: Because Python 2 performs classic division for integers. Use floats to perform true division. For example: 2.0/3

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

```
What is the value of the expression 4 * (6 + 5)

What is the value of the expression 4 * 6 + 5

What is the value of the expression 4 + 6 * 5
```

```
4 * (6 + 5)
```

```
44
```

```
4 * 6 + 5
```

```
29
```

```
4 + 6 * 5
```

```
34
```

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

Answer: Floating Point Number

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

```
100 ** 0.5
```

```
10.0
```

```
10 ** 2
```

```
100
```

Strings

Given the string 'hello' give an index command that returns 'e'. Use the code below:

```
s = 'hello'
# Print out 'e' using indexing
s[1]
```

```
'e'
```

Reverse the string 'hello' using indexing:

```
s ='hello'

# Reverse the string using indexing

s[::-1]
```

```
'olleh'
```

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

```
s ='hello'

# Print out the

s[-1]
```

```
'o'
```

```
s[4]
```

```
'o'
```

Lists

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

```
#Method 1
[0]*3
```

```
[0, 0, 0]
```

```
#Method 2
l = [0,0,0]
l
```

```
[0, 0, 0]
```

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

```
l = [1,2,[3,4,'hello']]
```

```
l[2][2] = 'goodbye'
```

```
l
```

```
[1, 2, [3, 4, 'goodbye']]
```

Sort the list below:

```
l = [5,3,4,6,1]
```

```
#Method 1
sorted(l)
```

```
[1, 3, 4, 5, 6]
```

```
#Method 2
l.sort()
l
```

```
[1, 3, 4, 5, 6]
```

Dictionaries

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

```
d = {'simple_key':'hello'}
# Grab 'hello'
```

```
d['simple_key']
```

```
'hello'
```

```
d = {'k1':{'k2':'hello'}}
# Grab 'hello'
```

```
d['k1']['k2']
```

```
'hello'
```

```
# Getting a little tricker
d = {'k1':[{'nest_key':['this is deep',['hello']]}]}
```

```
# This was harder than I expected...
d['k1'][0]['nest_key'][1][0]
```

```
'hello'
```

```
# This will be hard and annoying!
d = {'k1':[1,2,{'k2':['this is tricky',{'tough':[1,2,['hello']]}]}]}
```

```
# Phew
d['k1'][2]['k2'][1]['tough'][2][0]
```

```
'hello'
```

Can you sort a dictionary? Why or why not?

Answer: No! Because normal dictionaries are *mappings* not a sequence.

Tuples

What is the major difference between tuples and lists?

Tuples are immutable!

How do you create a tuple?

```
t = (1,2,3)
```

Sets

What is unique about a set?

Answer: They don't allow for duplicate items!

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

```
l = [1,2,2,33,4,4,11,22,3,3,2]
```

```
set(l)
```

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

Booleans

For the following quiz questions, we will get a preview of comparison operators:

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

```
# Answer before running cell
2 > 3
```

```
False
```

```
# Answer before running cell
3 <= 2
```

```
False
```

```
# Answer before running cell
3 == 2.0
```

```
False
```

```
# Answer before running cell
3.0 == 3
```

```
True
```

```
# Answer before running cell
4**0.5 != 2
```

```
False
```

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

```
# 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
```

Check for Leap year? 2000 is leap 2004 leap 1900 not leap year

```
def checkLeapYear(year):

#importing the calendar module

    import calendar

#returning boolean value based on given year is leap year or not
```

```
    return(calendar.isleap(year))



# Driver Code for the leap year program in python

#taking input (year) from the user

year = int(input("Enter the year: "))

if (checkLeapYear(year)):

#if checkLeapYear function returns true

    print("Leap Year")

else:

#if checkLeapYear function returns false

    print("Not a Leap Year")
```

OUTPUT:

```
Enter the year: 2000

Leap Year

Enter the year: 1900

Not a Leap Year
```

Great Job on your first assessment!