Test your knowledge.

Answer the following questions

Write (or just say out loud to yourself) a brief description of all the following Object Types and Data Structures we've learned about. You can edit the cell below by double clicking on it. Really this is just to test if you know the difference between these, so feel free to just think about it, since your answers are self-graded.

Double Click HERE to edit this markdown cell and write answers.

Numbers: Numbers are immutable type. When the value of a number is changed a new object is alloted they also allows scalar storage and direct acces.

Strings: String can be defined as a sequance of characters representation in a quotation mark

Lists: List can be contain different data types. The items are stored in the list afre separated by comma in square braces. List are mutable.

Tuples: Tuple contains that are used to data items. They are similar to list allows a elements to store.Tuple are represent in paranthesis it is not mandatory.

Dictionaries: Dictionary refers to in python mapping type. they are similar to associative arrays or hash in perl. The syntax of dictionary is(key : value)

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 [1]:
```
(60 + (10 ** 2) / 4 * 7) - 134.75
```
Out[1]:
```
100.25
```

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

In [2]:
```
4 * (6 + 5)
```
Out[2]:
```
44
```

In [3]:
```
4 * 6 + 5
```
Out[3]:
```
29
```

In [4]:
```
4 + 6 * 5
```
Out[4]:
```
34
```

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

Floating point real number

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

In [5]:
```
# Square root:
100 ** 0.5
```
Out[5]:
```
10.0
```

In [6]:
```
# Square:
10 ** 2
```
Out[6]:
```
100
```

Strings

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

In [7]:
```
s = 'hello'
# Print out 'e' using indexing

s[1]
```
Out[7]:
```
'e'
```

Reverse the string 'hello' using slicing:

In [8]:
```
s ='hello'
# Reverse the string using slicing

s[::-1]
```
Out[8]:
```
'olleh'
```

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

In [9]:
```
s ='hello'
# Print out the 'o'

# Method 1:
s[-1]
```
Out[9]:
```
'o'
```

In [10]:
```
# Method 2:
s[4]
```
Out[10]:
```
'o'
```

Lists

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

In [11]:
```
# Method 1:
[0]*3
```
Out[11]:
```
[0, 0, 0]
```

In [12]:
```
# Method 2:
list2 = [0,0,0]
list2
```
Out[12]:
```
[0, 0, 0]
```

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

In [13]:
```
list3 = [1,2,[3,4,'hello']]
```

In [14]:
```
list3[2][2] = 'goodbye'
```

In [15]:
```
list3
```
Out[15]:
```
[1, 2, [3, 4, 'goodbye']]
```

Sort the list below:

In [16]:
```
list4 = [5,3,4,6,1]
```

In [17]:
```
sorted(list4)
```
Out[17]:
```
[1, 3, 4, 5, 6]
```

In [18]:
```
list4.sort()
list4
```
Out[18]:
```
[1, 3, 4, 5, 6]
```

Dictionaries

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

In [19]:
```
d = {'simple_key':'hello'}
# Grab 'hello'
d['simple_key']
```
Out[19]:
```
'hello'
```

In [20]:
```
d = {'k1':{'k2':'hello'}}
# Grab 'hello'
d['k1']['k2']
```
Out[20]:
```
'hello'
```

In [21]:
```
# Getting a little trickier
d = {'k1':[{'nest_key':['this is deep',['hello']]}]}

#Grab hello
d = {'k1':[{'nest_key':['this is deep',['hello']]}]}
```

In [22]:
```
d['k1'][0]['nest_key'][1][0]
```
Out[22]:
```
'hello'
```

In [55]:
```
# This will be hard and annoying!
d = {'k1':[1,2,{'k2':['this is tricky',{'tough':[1,2,['hello']]}]}]}
```

In [25]:
```
d['k1'][2]['k2'][1]['tough'][2][0]
```
Out[25]:
```
'hello'
```

Can you sort a dictionary? Why or why not?

It is not possible to sort a dictionary only to get a representation of a dictionary that is sorted. Dictionaries are inherently orderless but other types such as lists and tuples are not. So you need an ordered data type to represent sorted values which will be a list probably a list of tuples.

Tuples

What is the major difference between tuples and lists?

list is mutable, whereas a tuple is immutable. This means that lists can be changed, and tuples cannot be changed. So, some operations can work on lists, but not on tuples.List can represent in square brackets and tuples are paranthesis.

How do you create a tuple?

In [45]:
```
t = (1,2,3)
```

Sets

What is unique about a set?

A set is an unordered collection of items. Every set element is unique no duplicates and must be immutable cannot be changed. However, a set itself is mutable. We can add or remove items from it.

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

In [46]:
```
list5 = [1,2,2,33,4,4,11,22,3,3,2]
```

In [47]:
```
set(list5)
```
Out[47]:
```
{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 [48]:
```
# Answer before running cell
2 > 3
```
Out[48]:
```
False
```

In [49]:
```
# Answer before running cell
3 <= 2
```
Out[49]:
```
False
```

In [50]:
```
# Answer before running cell
3 == 2.0
```
Out[50]:
```
False
```

In [51]:
```
# Answer before running cell
3.0 == 3
```
Out[51]:
```
True
```

In [52]:
```
# Answer before running cell
4**0.5 != 2
```
Out[52]:
```
False
```

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

In [60]:
```
# 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']
```
Out[60]:
```
False
```

Great Job on your first assessment!