Tutorial 1

1 - Write Python code that prints Hello World!

In [1]:

```
print("Hello World!")
```

```
Hello World!
```

2 - Write Python code to print out the number of items in a list, B.

In [2]:

```
B = ["apple", "banana", "cherry", 100, 3.14]
print(len(B))
```

```
5
```

3 - Write Python code to construct a list, L, with the following components in order: a string: 'Hello', a real number: 0.345, two integer numbers: 111333 and 44.

In [3]:

```
L = ['Hello',0.345, 111333, 44]
```

4 - Given some list, A:

a. Write Python code to make L2 a list containing only the first to the 3rd element of A.

b. Write Python code to make L3 a list containing only the last 3 elements of A.

In [4]:

```
A = [1,2,3,4,5,6]
L2 = A[:3] # or L2 = A[0:3]
print(L2)
L3 = A[-3:]
print(L3)
```

```
[1, 2, 3]
[4, 5, 6]
```

5 - A list, L, contains the following components in order: 'hello',23.45, 45, 'Brown', and [756.45, 34.5]. Write Python code to:

a. Modify L by adding 111 to the integer number component of the list (that is, the 45).

b. Modify L by inserting the elements 'new' and 21.5 between the 'Hello' and the 23.45 of L.

c. Modify L by inserting the list ['old', 33] before the element 'Brown'. Note, this is not inserting two elements but inserting one element which is a list.

In [5]:

```
L = ['hello',23.45, 45, 'Brown', [756.45, 34.5]]

L[2] += 111
print(L)

L[1:1] = ['new',21.5]
print(L)

L[5:5] = [['old', 33]]
print(L)
```

```
['hello', 23.45, 156, 'Brown', [756.45, 34.5]]
['hello', 'new', 21.5, 23.45, 156, 'Brown', [756.45, 34.5]]
['hello', 'new', 21.5, 23.45, 156, ['old', 33], 'Brown', [756.45, 34.5]]
```

6 - List the ways a string such as ’Hello’ can be represented.

In [6]:

```
'hello', "hello", """hello""", '''hello'''
```

Out[6]:

```
('hello', 'hello', 'hello', 'hello')
```

7 - Write Python code to:

a. Make a new string S with the concatenation of the two strings ’Tony’ and ’Vignaux’.

b. Do the same as in part (a) but including a space between the two words of the string.

c. Print out the first letter and the last letter of the new string.

In [7]:

```
S = 'Tony' 'Vignaux' #a
print(S)

S = 'Tony' + 'Vignaux'
print(S)

S = 'Tony' ' ' 'Vignaux' #b
print(S[0], S[-1])
```

```
TonyVignaux
TonyVignaux
T x
```

8 - L is a a string which may have unwanted spaces at both ends. Write Python code to return a string, M, without those spaces.

In [8]:

```
L = " Vignaux "
M = L.strip()
print(M)
```

```
Vignaux
```

9 - Write Python code to print out a string, S, without the automatic newline after it.

In [9]:

```
# We can use print(S,) in Python 2.
# In Python 3, use the code below to suppress the normal newline printed after the string i
S = 'test string'
print(S, end = " ")
print("and the next string")
```

```
test string and the next string
```

10 - You have a dictionary: d={'name':'Joe Blogps','weight':81}. Write Python code to:

a. Correct the name to 'Joe Bloggs',
b. Add a new entry for a height of 1.90 metres,
c. Print out all the entries in the dictionary

In [10]:

```
d = dict(name='Joe Blogps',weight=81)
d['name']='Joe Bloggs' #a
d['height'] = 1.90 #b
kys = d.keys() #c
for k in kys:
    print(k, d[k])

# to get the list with the keys in alphabetic order:
kys = sorted(d.keys())
for k in kys:
    print(k, d[k])
```

```
name Joe Bloggs
weight 81
height 1.9
height 1.9
name Joe Bloggs
weight 81
```

11 - I have an integer, i ≥ 0. Write Python code that will print out 'zero' if it is zero, 'one' if it is 1, and 'more than one' if it is more than 1.

In [11]:

```
i = 2
if (i==0):
    print('zero')
elif (i==1):
    print('one')
else:
    print('more than one')
```

```
more than one
```

12 - Write Python code to print out the numbers from 0 to 9. Use `range()` .

In [12]:

```
for i in range(10):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
```

13 - Write Python code to print out the numbers from 10 to 1 in decreasing order.

In [13]:

```
for i in range(10,0,-1):
    print(i)
```

```
10
9
8
7
6
5
4
3
2
1
```

14 -

a. Write Python code to construct a list L of n random numbers. Use n = 30 and print out the list L. Use the random seed of 111.
b. Adding to your Python code in part (a), calculate (and print out) the total of the values in the list L. Comment on this value, i.e., what would you expect it to be (approximately).

In [14]:

```
import random
random.seed(111)
n = 30
L = []
total = 0.0
for k in range(n):
    r = random.random()
    L.append(r)
    total += r
print(L)
print(total)
```

```
[0.827170565342314, 0.21276311517617263, 0.9425194436011797, 0.4939197167322
6975, 0.3975871534419906, 0.6171646774684565, 0.16962941460111325, 0.1936121
7094892524, 0.42164500474248356, 0.22970448570930002, 0.6704298068806634, 0.
41623153404520497, 0.38353495832202344, 0.2777463300387698, 0.22078858171623
96, 0.716191356812786, 0.9025521335945361, 0.3502399586135774, 0.80404143060
08918, 0.31313469765916013, 0.760409926484713, 0.23877406038049276, 0.568711
2022313325, 0.4074439634684711, 0.8709178300582484, 0.7612946965408907, 0.10
7791820862428, 0.09929772718869623, 0.4600412614040167, 0.2152795518990661]
14.050568576566413
```

Comment - We would expect the total of 30 U(0; 1) random numbers to be approximately 30 x 0.5 = 15 so this looks fine.

15 - Knuth algorithm

Let L be a list of n numbers. We can produce a random permutation of L by a sequence of random interchanges (Knuth shuffle), as follows:

Start with any permutation (for example, the identity permutation), and then go through the positions 0 through n − 2 (we use a convention where the first element has index 0, and the last element has index n − 1), and for each position i swap the element currently there with a randomly chosen element from positions i through n − 1 (the end), inclusive. It's easy to verify that any permutation of n elements will be produced by this algorithm with probability exactly 1/n!, thus yielding a uniform distribution over all such permutations.

Write Python code to construct a list L=[0,1,2,...,n-1] and then form a random permultation of L. You will find the function `random.randint` useful.

In [15]:

```
import random

random.seed(1)
L = [*range(10)]   # or L = list(range(10))
n = len(L)
for i in range(n-1):
    k = random.randint(i, n-1)
    t = L[i]
    L[i] = L[k]
    L[k] = t

print(L)
```

```
[2, 0, 6, 3, 7, 8, 9, 1, 4, 5]
```

16 - Very briefly, who is Guido van Rossum?

Guido van Rossum is the inventor of Python and a leading developer. He is known as python's "Benevolent Dictator for Life" (BDFL) and worked for Google and Dropbox. See http://en.wikipedia.org/wiki/Guido_van_Rossum (http://en.wikipedia.org/wiki/Guido_van_Rossum)

In [ ]:

```
```