Ayush Goyal 190905522
CSE D 62

Distributed Systems Week 1: Lab 1: Python Basic Practice-I

1. Assigning Values to Variables

```
counter = 100 # An integer assignment
miles = 1000.0 # A floating point
name = "John" # A string
print (counter)
print (miles)
print (name)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
100
1000.0
John
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

2. Multiple Assignment

Python allows you to assign a single value to several variables simultaneously.
For example:

```
a = b = c = 1
print(a,b,c)
a, b, c = 1, 2, "john"
print(a,b,c)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
1 1 1
1 2 john
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

3. Standard Data Types

Python has five standard data types:
Numbers
String
List
Tuple
Dictionary

4. Python Numbers

```
a = 5 # integer assignment
b= 4.56 #floating point assignment
#mathematical operations with scalar variables
print (5*a) #would give the result 25
print (a/2) #would give the result 2.5
print(a**2) #is the power (squaring operation) would give the result 25
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
25
2.5
25
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

5. import numpy as np

numpy includes various scientific functions like log10, log2, natural log, exp, floor, ceil, rounding and all statistical summary functions.

6. Python Strings

```
str = 'Hello World!'
print (str) # Prints complete string
print (str[0]) # Prints first character of the string
print (str[2:5]) # Prints characters starting from 3rd to 5th
print (str[2:]) # Prints string starting from 3rd character
print (str * 2) # Prints string two times
print (str + "TEST") # Prints concatenated string
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Hello World!
H
llo
llo World!
Hello World!Hello World!
Hello World!TEST
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

7. Updating a string

```
var1 = 'Hello World!'
print ("Updated String :", var1[:6] + 'Python')
```

Ans: Updated String :Hello Python

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Updated String : Hello Python
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

8. String formatting operator

One of Python's coolest features is the string format operator %. This operator is unique to strings and makes up for the pack of having functions from C's printf() family. Following is a simple example:

```
print( "My name is %s and weight is %d kg!" % ('Anitha', 55))
```
My name is Anitha and weight is 55 kg!

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
My name is Anitha and weight is 55 kg!
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

9. Built-in String methods

capitalize() , the first character of the string is converted to upper case.

str = "this is string example....wow!!!";
print (str.capitalize())

Ans: This is string example....wow!!!

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
This is string example....wow!!!
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

count(), counts the number of times a specific substring , occurrence in the main string

str = "this is string example....wow!!!";
print(str.count('s'))

Ans: 3 # three times s , appears in str.

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
3
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

find() , will locate the position of searching substring , (index)

print(str.find('example'))

Ans: 15 #at 15th index, the substring ‘example’ is placed.

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
15
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

lower(), returns a copy of the string in which all case-based characters have been lowercased.

str = "THIS IS STRING EXAMPLE....WOW!!!";
print (str.lower())

this is string example....wow!!!

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
this is string example....wow!!!
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

replace(), this method returns a copy of the string with all occurrences of substring old replaced by new.

str = "this is string example....wow!!! this is really string";
print (str.replace("is", "was"))

Ans: thwas was string example....wow!!! thwas was really string

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
thwas was string example....wow!!! thwas was really string
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

swapcase(), this method returns a copy of the string in which all the case-based characters have had their case swapped.

str = "this is string example....wow!!!";
print (str.swapcase())

Ans: THIS IS STRING EXAMPLE....WOW!!!"

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
THIS IS STRING EXAMPLE....WOW!!!
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

title(), returns a copy of the string in which first characters of all the words are capitalized.

str = "this is string example....wow!!!";
print (str.title())

Ans: This Is String Example....Wow!!!

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
This Is String Example....Wow!!!
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

10. Python LIST

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (*) is the repetition operator. For example:

list = ['abcd', 786 , 2.23, 'john', 70.2]
tinylist = [123, 'john']
print (list) # Prints complete list
print (list[0]) # Prints first element of the list
print (list[1:3]) # Prints elements starting from 2nd till 3rd
print (list[2:]) # Prints elements starting from 3rd element
print (tinylist * 2) # Prints list two times
print (list + tinylist) # Prints concatenated lists

Output will be like this

['abcd', 786, 2.23, 'john', 70.200000000000003]
abcd
[786, 2.23]
[2.23, 'john', 70.200000000000003]
[123, 'john', 123, 'john']

['abcd', 786, 2.23, 'john', 70.200000000000003, 123, 'john']

Note: LIST is mutable data type

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
['abcd', 786, 2.23, 'john', 70.2]
abcd
[786, 2.23]
[2.23, 'john', 70.2]
[123, 'john', 123, 'john']
['abcd', 786, 2.23, 'john', 70.2, 123, 'john']
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

11. Functions & Methods in LIST

list = ['physics', 'chemistry', 1997, 2000];
list.append('maths')
print(list)

Ans: ['physics', 'chemistry', 1997, 2000, 'maths'];

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
['physics', 'chemistry', 1997, 2000, 'maths']
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

To delete an element in a list

del list[2] #will remove 1997 record from the list.
print(list)

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
['physics', 'chemistry', 2000, 'maths']
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

To check a data in a list,
'physics' in list #will return 'True'
'english' in list #will return 'False'
len(list) #will return total items in list
list.count('physics')

Ans: 1

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
True
False
4
1
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

list.pop () #will **remove and return the last item from the list.**

list = ['physics', 'chemistry', 1997, 2000];
list.pop()
print(list)

Ans: ['physics', 'chemistry', 1997]l

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
['physics', 'chemistry', 1997]
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

ist.insert() #will **insert an item in the specified index**

list = ['physics', 'chemistry', 1997, 2000];
list.insert (2, 'maths')
print(list)

Ans: ['physics', 'chemistry', 'maths', 1997, 2000];

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
['physics', 'chemistry', 'maths', 1997, 2000]
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

list = ['physics', 'chemistry', 1997, 2000];
list.remove('chemistry') #will **remove the item specified.**
print(list)

Ans: = ['physics', 1997, 2000];

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
['physics', 1997, 2000]
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

list = ['physics', 'chemistry', 1997, 2000];
list.reverse() #will **reverse the objects of the list in place.**
print(list)

Ans: [2000, 1997, 'chemistry','physics']

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
[2000, 1997, 'chemistry', 'physics']
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

12. Python TUPLE

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.
The main differences between lists and tuples are: Lists are enclosed in brackets ([ ]) and their elements and size can be changed, while tuples are enclosed in parentheses (( )) and cannot be updated. Tuples can be thought of as read-only lists.

```
tuple = ( 'abcd', 786 , 2.23, 'john', 70.2 )
list = [ 'abcd', 786 , 2.23, 'john', 70.2 ]
print(tuple)
print(list)
tuple[2] = 1000 # Invalid syntax with tuple
list[2] = 1000 # Valid syntax with list
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
('abcd', 786, 2.23, 'john', 70.2)
['abcd', 786, 2.23, 'john', 70.2]
Traceback (most recent call last):
  File "script.py", line 5, in <module>
    tuple[2] = 1000 # Invalid syntax with tuple
TypeError: 'tuple' object does not support item assignment
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

13. Looping & Conditional Branches in Python

Eg.1

```
num=float(input('Enter a number:'))
if num>0:
        print('pos number')
elif num==0:
        print('zero')
else:
        print('Neg number')
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:34
pos number
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:-12
Neg number
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.2

```
x=float(input('Enter a number:'))
if x<10:
        print('smaller')
if x>20:
        print('bigger')
print('Finished')
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:5
smaller
Finished
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:56
bigger
Finished
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.3

```
x=5
print('Before 5')
if x==5:
        print ('this is 5')
        print('still 5')
print('After 5')
print('Before 6')
if x==6:
        print('this is 6')
print('After 6')
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Before 5
this is 5
still 5
After 5
Before 6
After 6
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.4: which will never print?

```
x=float(input('Enter a number:'))
if x<20:
        print('Below 20')
elif x<10:
        print('Below 10')
else:
        print('something else')
```

Ans: Below 10

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:15
Below 20
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:5
Below 20
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.5: Nested Decisions

```
x=42
if x>1:
        print('above one')
if x<100:
        print('less than 100')
print('All done')
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
above one
less than 100
All done
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.6: Ternary operator

```
age=15
b=('kid' if age<18 else 'adult')
print(b) #this will print 'kid'
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
kid
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

14. Usage of For-loop

Eg.1

```
for val in [5,4,3,2,1]:
        print(val)
print ('Done')
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
5
4
3
2
1
Done
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.2

```
stud=['Ram','Vijay','Nithya','Anu','Ramesh','suja']
for k in stud:
        print('Hello:', k)
print('done')
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Hello: Ram
Hello: Vijay
Hello: Nithya
Hello: Anu
Hello: Ramesh
Hello: suja
done
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.3

```
for i in range(5):
        print(i)
        if i>2:
                print('Bigger than 2')
        print('Done with i',i)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
0
Done with i 0
1
Done with i 1
2
Done with i 2
3
Bigger than 2
Done with i 3
4
Bigger than 2
Done with i 4
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.4: Calculate factors of a number

```
x=int(input('Enter a number:'))
for i in range(1,x+1):
        if x%i ==0:
                print(i)
```

x=10
1,2,5,10

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:5
1
5
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
Enter a number:10
1
2
5
10
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.5: Calculate largest number in an array

```
from math import *
x= [9, 41, 12, 3, 74, 15]
Largest=-inf
for i in x:
        if i>Largest:
                Largest=i
print(Largest)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
74
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.6: Calculate smallest number in an array

```
from math import *
x= [9, 41, 12, 3, 74, 15]
smallest=inf
for i in x:
        if i<smallest:
                smallest=i
print(smallest)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
3
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.7: Calculate the count, sum and average of numerical array

```
x = [9, 41, 12, 3, 74, 15]
count=sum=avg=0
for i in x:
        count=count+1
        sum=sum+1
avg=sum/count
print(count)
print(sum)
print(avg)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
6
6
1.0
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.8: Filtering in a loop (print all numbers >20)

```
x = [9, 41, 12, 3, 74, 15]
for i in x:
        if i>20:
                print (i)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
41
74
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.9: For the above problem, instead of printing the result, store the elements in a variable (object)

```
x = [9, 41, 12, 3, 74, 15]
res=[]
for i in x:
        if i>20:
                res.append(i)
print(res)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
[41, 74]
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

Eg.10: For the above x, replace all elements <20 into zero. Store the result in different variable (object)

```
y=np.zeros(len(x))
for i in range(len(x)):
        if x[i]>20:
                y[i]=x[i]
print(y)
```

```
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$ python3 script.py
[ 0. 41.  0.  0. 74.  0.]
student@dslab-12:~/Desktop/DSLab/AyushGoyal190905522/Week_1$
```

THE END