Python-Getting Started ¶

Slide 9: Assigning single value to a variable…

In [1]:

```
a = 10
name = 'Victor'
salary = 2000.23
print (a)
print (name)
print (salary)
```

```
10
Victor
2000.23
```

In [5]:

```
# Python code t get difference of two lists


# compare Code
li1 = [2,4,5,6,6,7]
li2 = [23,5,6,3,6,7]
print(Diff(li1, li2))

```

```
[2, 4]
```

In [ ]:

```

```

In [1]:

```
a = 10
name = 'Victor'
salary = 2000.23
print (a)
print (name)
print (salary)
```

```
10
Victor
2000.23
```

In [ ]:

```

```

Slide 10 : Assigning multiple value to a variable…

In [2]:

```
a=b=c=10
x,y,z=10,20,30
print(y)
print(a)


```

```
20
10
```

In [3]:

```
#will throw an error
x,y,z=10,20
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-3-fab2237f1a2e> in <module>()
      1 #will throw an error
----> 2 x,y,z=10,20

ValueError: not enough values to unpack (expected 3, got 2)
```

Slide 14:String Literals

In [4]:

```
name1="John"
name2='James'
print(name1)
print(name2)
text1='hello\
World'
text1
print(text1)
multiline='''st1
...st2
...st3'''
print(multiline)
```

```
John
James
helloWorld
st1
...st2
...st3
```

Slide 17: Special Literals: None Works like a null value, or empty value

In [5]:

```
val1 = 10
val2 = None
val1,val2
print(val2)
```

```
None
```

Operators

Slide : 19 Arithmetic Operator +, -, *, /, %

In [6]:

```
1+2
```

Out[6]:

```
3
```

In [7]:

```
1-2
```

Out[7]:

```
-1
```

In [8]:

```
2%1
```

Out[8]:

```
0
```

Slide : 20 Assignment Operator =,+=, -=, *=

In [9]:

```
a=10
a*=10 #a=10*10
print(a)
```

```
100
```

Slide 21: Comparison Operator <, >, <=, >=, !=

In [10]:

```
a = 10
b = 20
a > b
```

Out[10]:

```
False
```

In [11]:

```
Slide 21: Comparison Operator <, >, <=, >=, !=
```

```
  File "<ipython-input-11-c926d40bf81a>", line 1
    Slide 21: Comparison Operator <, >, <=, >=, !=
           ^
SyntaxError: invalid syntax
```

In [12]:

```
a = 10
b = 20
a != b
```

Out[12]:

```
True
```

Slide 21: More Operators =, +=, -=, *=

In [13]:

```
a = 10
a*=10#a=10*10
print(a)
```

```
100
```

Logical Operator: and, or , not

In [14]:

```
a=10<10 and 2>-1
print(a)
```

```
False
```

Bitwise Operator &,|,>>,<<,~

In [15]:

```
7 | 5

```

Out[15]:

```
7
```

In [16]:

```
7 & 5
```

Out[16]:

```
5
```

Identity Operator is, is not

In [17]:

```
x = 10
x is 10
```

Out[17]:

```
True
```

In [18]:

```
x = 10
x is not 10
```

Out[18]:

```
False
```

Membership Operator in, not in

In [19]:

```
pets=['dog','cat','wolf']
'lion' in pets
'wolf' in pets
'me' in 'appointment'
```

Out[19]:

```
True
```

Datatypes in Python

Number

In [20]:

```python
message = "Hello World"
num = 1234
pi = 13.6
print(type(message)) #return a string
print(type(num)) #return an integer
print(type(pi)) #return a float
```

```
<class 'str'>
<class 'int'>
<class 'float'>
```

Strings

In [21]:

```python
var1 = 'Hello-World!'
var2 = 'PythonTutorial'

print(var1[0]) # prints the first character in the string `H`
print(var2[1:5]) # prints the substring 'ytho'
```

```
H
ytho
```

find()

In [22]:

```python
#find() - Returns the position of the string
str='Attachment'
str.find('Me')
```

Out[22]:

```
-1
```

replace()

In [23]:

```python
#replace() - Replaces one character/string with other
str='Replacement'
str.replace('ed','e')
```

Out[23]:

```
'Replacement'
```

split()

In [24]:

```
#split() - Creates a split on the basis of a character
str='word1,word2,word3'
str.split(',')
```

Out[24]:

```
['word1', 'word2', 'word3']
```

Tuples

In [25]:

```
myGroup = ('a', 'b', 'c', 'd')
```

concatenation

In [26]:

```
#Concatenation - Adds two string/character
myGroup += ('f',)
print(myGroup)

```

```
('a', 'b', 'c', 'd', 'f')
```

repetation

In [27]:

```
#Repetition -  Duplicate string/character by given number of times
myGroup * 2
```

Out[27]:

```
('a', 'b', 'c', 'd', 'f', 'a', 'b', 'c', 'd', 'f')
```

indexing

In [28]:

```
#Indexing - Shows the indexed character/string
myGroup = ('a', 'b', 'c', 'd', 'f')
myGroup[2]
```

Out[28]:

```
'c'
```

slicing

In [29]:

```python
#Slicing - Shows a specific set of indexed character/string
myGroup = ('a', 'b', 'c', 'd', 'f')
myGroup[1:4]
```

Out[29]:

```
('b', 'c', 'd')
```

List

In [36]:

```python
myGroup = ('a', 10, 7.12, 'data')
```

concatenation

In [38]:

```python
#Concatenation - Add elements to the list
myList = ['a', 1, 3.14, 'python']
myList+=['d',]
print(myList)
```

```
['a', 1, 3.14, 'python', 'd']
```

repetition

In [39]:

```python
#Repetition
myList = ['a', 1, 3.14, 'python']
myList*2
```

Out[39]:

```
['a', 1, 3.14, 'python', 'a', 1, 3.14, 'python']
```

slicing

In [41]:

```python
#Slicing
myList = ['a', 1, 3.14, 'python']
myList = myList[0:4]
myList
```

Out[41]:

```
['a', 1, 3.14, 'python']
```

append

In [42]:

```python
#append(value)
myList = ['a', 1, 3.14, 'python']
myList.append('d')
myList
```

Out[42]:

```
['a', 1, 3.14, 'python', 'd']
```

extend

In [43]:

```python
#extend(list)
myList = ['a', 1, 3.14, 'python']
myList.extend (['c', 'd'])
myList
```

Out[43]:

```
['a', 1, 3.14, 'python', 'c', 'd']
```

insert

In [44]:

```python
#insert(index, value)
myList = ['a', 1, 3.14, 'python']
myList.insert(2, 'd')
myList
```

Out[44]:

```
['a', 1, 'd', 3.14, 'python']
```

Dictionary

Example

In [45]:

```
myDict = { 1: 'John' , 2: 'Bob', 3: 'Alice' }
```

Empty dictionary

In [46]:

```
#empty dictionary
myDict = {}
```

Integer keys

In [47]:

```
#dictionary with integer keys
myDict = {1: 'apple', 2: 'ball'}
```

In [48]:

```
#dictionary with mixed keys
myDict = {'name': 'John', 1: [2, 4, 3]}
```

mixed keys

In [49]:

```
#dictionary with mixed keys
myDict = {'name': 'John', 1: [2, 4, 3]}

```

In [50]:

```
#paring
#from sequence having each item as a pair
myDict = dict([(1,'apple'), (2,'ball')])
```

In [53]:

```
#accessing dictionary
myDict = {1: 'word1', 2: 'word2'}
myDict[2]
```

Out[53]:

```
'word2'
```

In [54]:

```
#len()
myDict = {1: 'word1', 2: 'word2'}
len(myDict)
```

Out[54]:

```
2
```

In [55]:

```
#key()
myDict = {1:'apple', 2:'ball'}
len(myDict)
myDict.keys()

```

Out[55]:

```
dict_keys([1, 2])
```

In [56]:

```
#values()
myDict = {1: 'apple', 2: 'ball'}
myDict.values()
['apple', 'ball']

```

Out[56]:

```
['apple', 'ball']
```

Sets

Example

In [ ]:

```
mySet = {1, 2, 3}
```

In [57]:

```
#Creating set with duplicate values
mySet = {1, 2, 3, 3}
print (mySet)
```

```
{1, 2, 3}
```

In [72]:

```python
#Union of set
mySet1 = {1, 2, 'c'}
mySet2 = {1, 'b', 'c'}
mySet1|mySet2
```

Out[72]:

```
{1, 2, 'b', 'c'}
```

In [59]:

```python
type(mySet1)
```

Out[59]:

```
set
```

In [61]:

```python
mySet1.union(mySet2)
```

Out[61]:

```
{1, 2, 'b', 'c'}
```

In [69]:

```python
mySet1.intersection(mySet2)
```

Out[69]:

```
{1, 'c'}
```

In [71]:

```python
mySet1.difference(mySet2)
```

Out[71]:

```
{2}
```

File Handling

In [3]:

```
#File Handling in Jupyter using Pandas
import pandas as pd
#df1 = pd.read_csv("C:/PythonHow/income_data.csv")
df1 = pd.read_csv("C:/Users/HP/Downloads/Intelli/IntelliDataSciencePython/PythonDataSc
print(df1)
```

```
    HP        MPG  VOL          SP         WT
0   49  53.700681   89  104.185353  28.762059
1   55  50.013401   92  105.461264  30.466833
2   55  50.013401   92  105.461264  30.193597
3   70  45.696322   92  113.461264  30.632114
4   53  50.504232   92  104.461264  29.889149
5   70  45.696322   89  113.185353  29.591768
6   55  50.013401   92  105.461264  30.308480
7   62  46.716554   50  102.598513  15.847758
8   62  46.716554   50  102.598513  16.359484
9   80  42.299078   94  115.645204  30.920154
10  73  44.652834   89  111.185353  29.363341
11  92  39.354094   50  117.598513  15.753535
```

Read

In [4]:

```
#Example
#f = open("demofile.txt", "r")
f = open("Cars.csv", "r")
print(f.read())

```

```
HP,MPG,VOL,SP,WT
49,53.70068138,89,104.1853528,28.7620589
55,50.01340115,92,105.4612635,30.46683298
55,50.01340115,92,105.4612635,30.19359657
70,45.69632238,92,113.4612635,30.63211391
53,50.50423183,92,104.4612635,29.88914864
70,45.69632238,89,113.1853528,29.59176832
55,50.01340115,92,105.4612635,30.30847957
62,46.71655428,50,102.5985128,15.84775807
62,46.71655428,50,102.5985128,16.35948352
80,42.29907817,94,115.6452041,30.92015417
73,44.65283424,89,111.1853528,29.36334142
92,39.3540941,50,117.5985128,15.75353468
```

In [5]:

```
#Reading parts of file
#f = open("demofile.txt", "r")
f = open("Cars.csv", "r")
print(f.read(5)) #Return the 5 first characters of the file

```

```
HP,MP
```

In [6]:

```
#Reading first line
#f = open("demofile.txt", "r")
f = open("Cars.csv", "r")
print(f.readline()) #readline() is used to return one line

```

```
HP,MPG,VOL,SP,WT
```

In [7]:

```
#using readline() twice prints first two line
#f = open("demofile.txt", "r")
f = open("Cars.csv", "r")
print(f.readline())
print(f.readline()) # using readline() twice prints first two line

```

```
HP,MPG,VOL,SP,WT

49,53.70068138,89,104.1853528,28.7620589
```

In [8]:

```
#Loop through the file
#Read the file line by line
#f = open("demofile.txt", "r")
f = open("Cars.csv", "r")
for x in f:
  print(x)

```

```
HP,MPG,VOL,SP,WT

49,53.70068138,89,104.1853528,28.7620589

55,50.01340115,92,105.4612635,30.46683298

55,50.01340115,92,105.4612635,30.19359657

70,45.69632238,92,113.4612635,30.63211391

53,50.50423183,92,104.4612635,29.88914864

70,45.69632238,89,113.1853528,29.59176832

55,50.01340115,92,105.4612635,30.30847957

62,46.71655428,50,102.5985128,15.84775807

62,46.71655428,50,102.5985128,16.35948352

80,42.29907817,94,115.6452041,30.92015417

73,44.65283424,89,111.1853528,29.36334142

92,39.3540941,50,117.5985128,15.75353468
```

Write/Create

In [9]:

```
#Example: Append
f = open("demofile.txt", "a")
f.write("Now the file has one more line!")

```

Out[9]:

```
31
```

In [10]:

```
#Example: Overwrite
f = open("demofile.txt", "w")
f.write("Woops! I have deleted the content!")
```

Out[10]:

```
34
```

In [12]:

```
# Create a new file. If file exists it will give an error
f = open("myfile1.txt", "x")

```

```
---------------------------------------------------------------------------
FileExistsError                           Traceback (most recent call last)
<ipython-input-12-3b58aa0e3571> in <module>
      1 # Create a new file. If file exists it will give an error
----> 2 f = open("myfile1.txt", "x")

FileExistsError: [Errno 17] File exists: 'myfile1.txt'
```

Delete

In [13]:

```
# Deleting the file
import os
os.remove("demofile.txt")
```

```
---------------------------------------------------------------------------
PermissionError                           Traceback (most recent call last)
<ipython-input-13-4566a23ffd32> in <module>
      1 # Deleting the file
      2 import os
----> 3 os.remove("demofile.txt")

PermissionError: [WinError 32] The process cannot access the file because it
is being used by another process: 'demofile.txt'
```

In [ ]:

```

```

In [ ]:

```

```