1.7 Introduction to Python

1.7.1 Declaring Variables

```
var1 = 2
var2 = 5.0
var3 = True
var4 = "Machine Learning"
```

```
print("Value of var1 :", var1)
print("Value of var2 :", var2)
print("Value of var3 :", var3)
print("Value of var4 :", var4)
```

```
Value of var1 : 2
Value of var2 : 5.0
Value of var3 : True
Value of var4 : Machine Learning
```

```
type( var1 )
```

```
int
```

```
type( var2 )
```

```
float
```

```
type(var3)
```

```
bool
```

```
type(var4)
```

```
str
```

1.7.2 Conditional Statements

```
# Checking a condition if the variable value is more than 1
if var1 > 1:
    print( "Bigger than 1" )
```

```
Bigger than 1
```

```
x = 10
y = 12

# if x is greater than y
if x > y:
    print ("x > y")
# if x is lesser than y
elif x < y:
    print ("x < y")
else:
    print ("x = y")
```

```
x < y
```

```
# Initialize
x = 5

# Assign True if x is more than 10 or assign False using ternary operator
isGreater = True if x > 10 else False
```

```
isGreater
```

```
False
```

1.7.3 Generating Sequence Numbers

```
# Initializing the sequence of numbers starting from 1
# and ending (not including) with 6
numbers = range( 1, 6 )

numbers
```

```
range(1, 6)
```

```
range?
```

1.7.4 Control Flow Statements

```
# Iterate through the collection
for i in numbers:
    print (i)
```

```
1
2
3
4
5
```

```
# Initialize the value of 1
i = 1

# check the value of i to check if the loop will be continued or not
while i < 5:

    print(i)
    # Increment the value of i.
    i = i+1

# print after the value of i
print('Done')
```

```
1
2
3
4
Done
```

1.7.5 Functions

```
def addElements( a, b ):
    return a + b
```

```
result = addElements( 2, 3 )

result
```

```
5
```

```
result = addElements( 2.3, 4.5 )

result
```

```
6.8
```

```
result = addElements( "python", "workshop" )

result
```

```
'pythonworkshop'
```

```
def addElements( a, b = 4 ):
    return a + b
```

```
addElements( 2 )
```

```
6
```

```
addElements( 2, 5 )
```

```
7
```

1.7.6 Working with Collections

1.7.6.1 List

```
## Create an empty list
emptyList = []
```

```
batsmen = ['Rohit', 'Dhawan', 'Kohli', 'Rahane', 'Rayudu', 'Dhoni']
```

```
batsmen[0]
```

```
'Rohit'
```

```
## Slicing an list
batsmen[0:2]
```

```
['Rohit', 'Dhawan']
```

```
## Accessing the last element
batsmen[-1]
```

```
'Dhoni'
```

```
# how many elements in the list
len( batsmen )
```

```
6
```

```
bowlers = ['Bumrah', 'Shami', 'Bhuvi', 'Kuldeep', 'Chahal']
```

```
all_players = batsmen + bowlers
```

```
all_players
```

```
['Rohit',
 'Dhawan',
 'Kohli',
 'Rahane',
 'Rayudu',
 'Dhoni',
 'Bumrah',
 'Shami',
 'Bhuvi',
 'Kuldeep',
 'Chahal']
```

```
'Bumrah' in bowlers
```

```
True
```

```
'Rayudu' in bowlers
```

```
False
```

Finding the index of an item in the list.

```
all_players.index( 'Dhoni' )
```

```
5
```

```
all_players.reverse()

all_players
```

```
['Chahal',
 'Kuldeep',
 'Bhuvi',
 'Shami',
 'Bumrah',
 'Dhoni',
 'Rayudu',
 'Rahane',
 'Kohli',
 'Dhawan',
 'Rohit']
```

1.7.6.2 Tuples

```
odiDebut = ( 'Kohli', 2008 )
```

```
odiDebut
```

```
('Kohli', 2008)
```

```
odiDebut[0]
```

```
'Kohli'
```

```
tup1[1] = 2009
```

```
---------------------------------------------------------------------
-------
NameError                                 Traceback (most recent cal
l last)
<ipython-input-38-9195c07b537c> in <module>()
----> 1 tup1[1] = 2009

NameError: name 'tup1' is not defined
```

```
players = tuple( all_players )
```

```
players
```

```
('Chahal',
 'Kuldeep',
 'Bhuvi',
 'Shami',
 'Bumrah',
 'Dhoni',
 'Rayudu',
 'Rahane',
 'Kohli',
 'Dhawan',
 'Rohit')
```

1.7.6.3 Set

```
setOfNumbers = {6,1,1,2,4,5}
```

```
setOfNumbers
```

```
{1, 2, 4, 5, 6}
```

```
wc2011 = {"Dhoni", "Sehwag", "Tendulkar", "Gambhir", "Kohli", "Raina", "Yuvraj",
"Yusuf"}
wc2015 = {"Dhoni", "Dhawan", "Rohit", "Rahane", "Kohli", "Raina", "Rayudu", "Jad
eja"}
```

```
wc2011.union( wc2015 )
```

```
{'Dhawan',
 'Dhoni',
 'Gambhir',
 'Jadeja',
 'Kohli',
 'Rahane',
 'Raina',
 'Rayudu',
 'Rohit',
 'Sehwag',
 'Tendulkar',
 'Yusuf',
 'Yuvraj'}
```

```
wc2011.intersection( wc2015 )
```

```
{'Dhoni', 'Kohli', 'Raina'}
```

```
wc2015.difference( wc2011 )
```

```
{'Dhawan', 'Jadeja', 'Rahane', 'Rayudu', 'Rohit'}
```

1.7.6.4 Dictionary

```
wcWinners = { 1975: "West Indies",
              1979: "West Indies",
              1983: "India",
              1987: "Australia",
              1991: "Pakistan",
              1996: "Srilanka",
              1999: "Australia",
              2003: "Australia",
              2007: "Australia",
              2011: "India"}
```

```
wcWinners[1983]
```

```
'India'
```

```
wcWinners.values()
```

```
dict_values(['Australia', 'Australia', 'Australia', 'Pakistan', 'Wes
t Indies', 'India', 'West Indies', 'Srilanka', 'Australia', 'Indi
a'])
```

```
set(wcWinners.values())
```

```
{'Australia', 'India', 'Pakistan', 'Srilanka', 'West Indies'}
```

```
wcWinners[2015] = 'Australia'
```

```
wcWinners
```

```
{1975: 'West Indies',
 1979: 'West Indies',
 1983: 'India',
 1987: 'Australia',
 1991: 'Pakistan',
 1996: 'Srilanka',
 1999: 'Australia',
 2003: 'Australia',
 2007: 'Australia',
 2011: 'India',
 2015: 'Australia'}
```

1.7.7 Dealing with Strings

```
string0 = 'python'
string1 = "machine learning"
```

```
string2 = """This is a
 multiline string"""
```

```
# Converting to upper case
string0.upper()
# Similarly string.lower() can be used to convert to lower case.
# string0.lower()
```

```
'PYTHON'
```

```
tokens = string1.split(' ')
tokens
```

```
['machine', 'learning']
```

1.7.8 Functional Programming

1.7.8.1 Example 1: Map

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

```
# Create an empty list.
squareList = []

# Loop through the intList, square every item and append to result list squareLi
st.
for x in intList:
    squareList.append( pow( x, 2 ) )

print( squareList )
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
def square_me( x ):
    return x * x
```

```
squareList = map( square_me, intList)
```

```
list(squareList)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
squareList = map(lambda x: x*x, intList)
list(squareList)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81]
```

1.7.8.2 Example 2: Filter

```
evenInts = filter( lambda x : x % 2 == 0, intList )
```

```
list( evenInts )
```

```
[2, 4, 6, 8]
```

1.7.9 Modules and Packages

```
import math

## Taking square root of a value
math.sqrt(16)
```

```
4.0
```

```
from random import sample
```

```
sample( range(0, 11), 3)
```

```
[0, 4, 3]
```

1.7.10 Other Features

```
import random

randomList = random.sample( range(0, 100), 20)

randomList
```

```
[23, 28, 76, 72, 3, 39, 63, 74, 99, 97, 57, 6, 33, 62, 24, 71, 50, 2
7, 22, 30]
```

```
from statistics import mean, median

def getMeanAndMedian( listNum ):
    return mean(listNum), median(listNum)
```

```
mean, median = getMeanAndMedian( randomList )
```

```
print( "Mean: ", mean, " Median: ", median)
```

```
Mean:  47.8  Median:  44.5
```