Python Programming - 30 May 2019

Date: 30 May 2019

Session Objectives

- To introduce jupyter notebooks
- To teach ***Markdown*** syntax
- To introduce **python** basics
    - Basic Syntax(variables, Assignment, Data Types)
    - Control Structures
        1. Conditional
        2. Repetitive

Link to Main Website (https://sites.google.com/view/srit-python-may-2019/about-the-program)

Python Basics

In [29]:

```
print("Hello world..!") #single parameter
print("Hello","world",end="  ") # Multiple paramters
print("Hello"+"world") #  using +

```

```
Hello world..!
Hello world  Helloworld
```

In [30]:

```
#This is comment

n = input("Enter a value:") # capture input and store it in a variable
print("Value is:",n)
```

```
Enter a value:25
Value is: 25
```

In [31]:

```
n = int(n) # type conversion to integer and reassign
n * n # expression evaluation using arithmetic operator
```

Out[31]:

```
625
```

In [41]:

```
a = b = c = n # multi variable assignment with same value

print(a, b, c)

d, e, f = 123, 234, 345 # multi varaibles are assigned with different values
print(d, type(d), e, type(e), f, type(f))

```

```
25 25 25
123 <class 'int'> 234 <class 'int'> 345 <class 'int'>
```

In [42]:

```
# Type Conversion
print(type(c))
c = str(c)
print(type(c))
c = float(c)
print(type(c))
```

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

Higher Order Computation

In [131]:

```
num = 987 ** 987
s = str(num)
l = len(s)
print(l)

```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-131-569ada085ae0> in <module>
      1 num = 987 ** 987
----> 2 s = str(num)
      3 l = len(s)
      4 print(l)

TypeError: 'str' object is not callable
```

String Slicing

In [57]:

```
s1 = "Python Programming"
print(s1[0]) # Access the first character
print(s1[-1]) # Access the last character
print(s1[0:6]) # Access the substring - first 6 characters
print(s1[:6]) # Access the substring - frist 6 characters
print(s1[7:len(s1)]) # Access second string
print(s1[7:]) # Access all characters from 7th index
```

```
P
g
Python
Python
Programming
Programming
```

In [62]:

```
print(s1[::-1]) # Reverse of a string
print(s1[-1:6:-1]) # Reverse of a substring - Programming
print(s1[5::-1]) # Reverse of a substring - Python
```

```
gnimmargorP nohtyP
gnimmargorP
nohtyP
```

In [72]:

```
s2 = '123456789'
print(s2[-1:2:-1]) # Reverse of last len(s2)-3
print(s2[5:2:-1]) # Reverse of characters from index 5 to index 3
```

```
987654
654
```

In [79]:

```
print(s2[0::2]) # Accessing alternate characters
print(s2[1::2]) # Accessing alternate characters
print(s2[-1::-2]) # Accessing alternate characters in reverse order
print(s2[-2::-2]) # Accessing alternate characters in reverse order
```

```
13579
2468
97531
8642
```

Functions in Python

In [81]:

```
# Function to Reverse a string
def reverseString(s):
    return s[::-1]
print(reverseString(s2))
```

```
987654321
```

In [86]:

```
# Function to Reverse a substring
def reverseSubString(s, i, j):
    sub = s[i:j+1]
    return sub[::-1]
print(reverseSubString(s2,2,5))
```

```
6543
```

Control Structures

Conditional Statements in Python

In [88]:

```
#Function to test divisibility by 9 and 11 and not 10
def divisibilityTest(n):
    if n % 9 == 0 and n % 11 == 0 and n % 10 != 0:
        return True
    else:
        return False

print(divisibilityTest(99))
print(divisibilityTest(990))
```

```
True
False
```

In [91]:

```
# Recursive Function for a power n
def powerN(a, n):
    if n == 0:
        return 1
    return a * powerN(a, n-1)


a = int(input("Enter base:"))
n = int(input("Enter exponent:"))
print(powerN(a,n))
```

```
Enter base:2
Enter exponent:10
1024
```

In [106]:

```
# Function to check a given string is palindrome or not
def palindrome(s1):
    s2 = s1[::-1]
    if s1 == s2:
        print("Palindrome")
    else:
        print("Not Palindrome")



str = input("Enter a string to check:")
str = str.lower()
palindrome(str)
```

```
Enter a string to check:Madam
Palindrome
```

Looping Statements in Python

In [112]:

```
for i in range(123, 126):
    print(i,end=" ")
```

```
123 124 125
```

In [114]:

```
# Function to print all numbers divisible by 7 in given range [lb, ub]
def divisible7(lb, ub):
    for i in range(lb, ub+1):
        if(i % 7 == 0):
            print(i, end=" ")



lb = int(input("Enter lower bound:"))
ub = int(input("Enter upper bound:"))
divisible7(lb,ub)

```

```
Enter lower bound:1
Enter upper bound:70
7 14 21 28 35 42 49 56 63 70
```

In [122]:

```
# Function to generate of prime numbers from 1 to n
def isFactor(dividend, divisor):
    if dividend % divisor == 0:
        return True
    return False

def isPrime(n):
    for i in range(2, n//2 + 1):
        if isFactor(n,i):
            return False
    return True

def genPrimes(k):
    for i in range(2,k+1):
        if isPrime(i):
            print(i,end=" ")

n=int(input("Enter a number:"))
genPrimes(n)
```

```
Enter a number:100
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

Lists in Python

In [155]:

```
li = [1, 2, 3, 4, 5, 6,'Python',5.6] # Any type of data can be kept in List

print(li)
print(li[-1])
print(li[-2:])
print(li[2:])
print(li[6][0])


li.append('sudheer') # to add(append) element at the end of the list
print(li)
li.append([12, 13, 14]) # to add(append) sublist in main list
print(li)


li.remove('sudheer') # to remove specified element from the list
print(li)


li[8].remove(13) # to remove element in a sublist
print(li)


li[8].insert(1,13) # to insert element at a particular index
print(li)


li.pop(3) #to remove selected index element
print(li)


numlist = [25, 12, 54, 14]
numlist.sort() # sort the list in ascending oder
print(numlist)


numlist.extend([21, 22, 23]) # merges two lists - means second list elements are merge
print(numlist)
numlist.sort()
print(numlist)


numlist.sort(reverse=True) # sorts the list in descending order
print(numlist)
```

```
[1, 2, 3, 4, 5, 6, 'Python', 5.6]
5.6
['Python', 5.6]
[3, 4, 5, 6, 'Python', 5.6]
P
[1, 2, 3, 4, 5, 6, 'Python', 5.6, 'sudheer']
[1, 2, 3, 4, 5, 6, 'Python', 5.6, 'sudheer', [12, 13, 14]]
[1, 2, 3, 4, 5, 6, 'Python', 5.6, [12, 13, 14]]
[1, 2, 3, 4, 5, 6, 'Python', 5.6, [12, 14]]
[1, 2, 3, 4, 5, 6, 'Python', 5.6, [12, 13, 14]]
[1, 2, 3, 5, 6, 'Python', 5.6, [12, 13, 14]]
[12, 14, 25, 54]
[12, 14, 25, 54, 21, 22, 23]
```

```
[12, 14, 21, 22, 23, 25, 54]
[54, 25, 23, 22, 21, 14, 12]
```

Python Programming - 31 May 2019

In [ ]:

```
1
```