Tutorial_Session2

March 17, 2018

1 String and List

1.1 Strings

- A string is a sequence of characters.
- A string may be specified by placing the member characters of the sequence within quotes (single, double or triple).
- Triple quotes are typically used for strings that span multiple lines.

```
In [1]: message = 'Hello Gita'
```

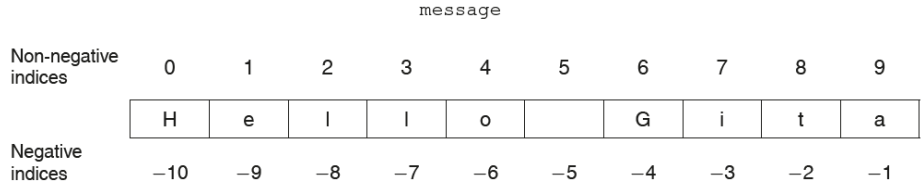
1.1.1 Computing Length using len function

```
In [2]: print(len(message))
```

```
10
```

1.1.2 Indexing

- Individual characters within a string are accessed using a technique known as indexing.




```
In [3]: index = len(message) - 1
        print(message[0], message[6], message[index], message[-1])
```

```
H G a a
```

```
In [4]: print(message[15])
```

```
---------------------------------------------------------------------------
```

```
IndexError                                Traceback (most recent call last)

<ipython-input-4-a801df50d8d1> in <module>()
----> 1 print(message[15])


IndexError: string index out of range
```

1.1.3 Slicing

- In order to extract the substring comprising the character sequence having indices from start to end-1, we specify the range in the form start:end.
- Python also allows us to extract a subsequence of the form *start:end:inc*.

```
In [89]: message = 'Hello Sita'
         print(message[0:5], message[-10:-5])
```

```
Hello Hello
```

```
In [90]: print(message[0:len(message):2])
         print(message[:])
```

```
HloSt
Hello Sita
```

1.1.4 Membership Operator in

- Python also allows us to check for membership of the individual characters or substrings in strings using in operator.

```
In [91]: 'h' in 'hello'
```

```
Out[91]: True
```

```
In [92]: 'ell' in 'hello'
```

```
Out[92]: True
```

```
In [93]: 'h' in 'Hello'
```

```
Out[93]: False
```

1.2 Built-in Functions on Strings

1.2.1 Function: count

- For counting number of occurrences of a substring.

```
In [94]: 'Encyclopedia'.count('c')
```

```
Out[94]: 2
```

```
In [95]: vowels = 'AEIOUaeiou'
         vowelCount = 0
         for ch in vowels:
             vowelCount += 'Encyclopedia'.count(ch)
         print(vowelCount)
```

```
5
```

1.2.2 Functions find and rfind

- Function **find**: Returns the index of the first occurrence of a string.
- Function **rfind**: Returns the index of the last occurrence of a string.

```
In [96]: colors = 'green, red, blue, red, red, green'
         print("colors.find('red'): ", colors.find('red'))
         print("colors.rfind('red'): ", colors.rfind('red'))
         print("colors.find('orange'): ", colors.find('orange'))
```

```
colors.find('red'):  7
colors.rfind('red'):  23
colors.find('orange'):  -1
```

1.2.3 Functions capitalize, title, lower, upper, and swapcase

- Function **capitalize**: converting the first letter of a string to uppercase character and converting the remaining letters in the string to lowercase.
- Function **title**: Capitalize the first letter of each word in a string and change the remaining letters to lowercase.
- Function **lower**: Convert all letters in a string to lowercase.
- Function **upper**: Convert all letters in a string to uppercase.

```
In [97]: 'python IS a Language'.capitalize()
```

```
Out[97]: 'Python is a language'
```

```
In [98]: 'python IS a PROGRAMMING Language'.title()
```

```
Out[98]: 'Python Is A Programming Language'
```

```
In [99]: emailId1 = 'geek@gmail.com'
         emailId2 = 'Geek@gmail.com'
         emailId1.lower() == emailId2.lower()
```

```
Out[99]: True
```

1.2.4 Function swapcase

```
In [4]: 'AnilKumar'.swapcase()
```

```
Out[4]: 'aNILkUMAR'
```

1.2.5 Functions islower, isupper, isalpha, isdigit, and isalnum

```
In [101]: 'python'.islower()
```

```
Out[101]: True
```

```
In [102]: 'Python'.isupper()
```

```
Out[102]: False
```

```
In [103]: '9953799924'.isdigit()
```

```
Out[103]: True
```

```
In [104]: 'Nikhil Kumar'.isalpha()
```

```
Out[104]: False
```

```
In [105]: password = 'Kailash107Ganga'
          password.isalnum()
```

```
Out[105]: True
```

1.2.6 Function replace

- It allows to replace part of a string by another string.
- It takes two arguments as inputs. The first argument is used to specify the substring that is to be replaced. The second argument is used to specify the string that replaces the first string.

```
In [106]: message = 'Amey my friend, Amey my guide'
```

```
In [107]: message.replace('Amey', 'Vihan')
```

```
Out[107]: 'Vihan my friend, Vihan my guide'
```

1.2.7 Functions strip, lstrip, and rstrip

- The functions **lstrip** and **rstrip** remove whitespaces from the beginning and end, respectively.
- The function **strip** removes whitespaces from the beginning as well as the end of a string.

```
In [108]: '       Hello How are you!      '.lstrip()
```

```
Out[108]: 'Hello How are you!      '
```

```
In [109]: '       Hello How are you!      '.rstrip()
```

```
Out[109]: '       Hello How are you!'
```

```
In [110]: '       Hello How are you!      '.strip()
```

```
Out[110]: 'Hello How are you!'
```

1.2.8 Functions split and partition

- The function **split** enables us to split a string into a list of strings based on a delimiter.
- The function **partition** divides a string S into two parts based on a delimiter and returns a tuple comprising string before the delimiter, the delimiter itself, and the string after the delimiter

```
In [111]: colors = 'Red, Green, Blue, Orange, Yellow, Cyan'
          colors.split(',')

Out[111]: ['Red', ' Green', ' Blue', ' Orange', ' Yellow', ' Cyan']

In [112]: 'Hello. How are you?'.partition('.')

Out[112]: ('Hello', '.', ' How are you?')
```

1.2.9 Function join

- The function **join** returns a string comprising elements of a sequence separated by the specified delimiter.

```
In [123]: ' > '.join(['I', 'am', 'ok'])

Out[123]: 'I > am > ok'

In [124]: ' '.join(('I', 'am', 'ok'))

Out[124]: 'I am ok'
```

1.3 Problem: Reversing a string

```
In [1]: def reverse(str1):
            '''
            Objective: To reverse a string
            Input Parameter: str1 - string
            Return Value: reverse of str1 - string
            '''
            reverseStr = ''
            for i in range(len(str1)):
                reverseStr = str1[i] + reverseStr
            return reverseStr

        def main():
            '''
            Objective: To reverse the string provided as input
            Input Parameter: None
            Return Value: None
            '''
            myString=input('Enter a string to be reversed:')
            reverseStr=reverse(myString)
```

```
        print('The reverse is:' + reverseStr)
    if __name__ == '__main__':
        main()
```

```
Enter a string to be reversed:PYTHON
The reverse is:NOHTYP
```

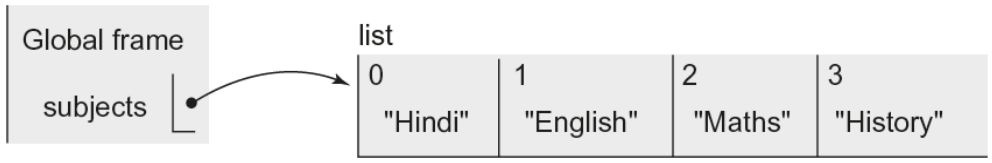
```
In [2]: reverse('Python')
```

```
Out[2]: 'nohtyP'
```

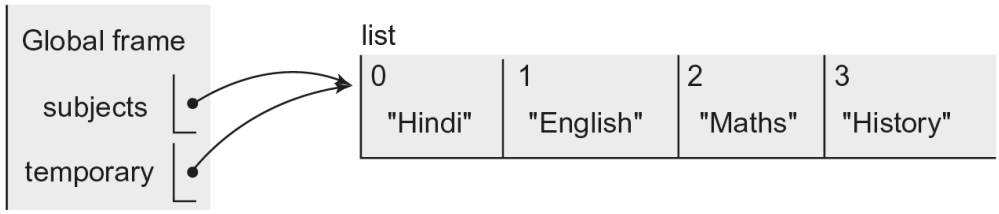
1.4 Lists

- A list is an ordered sequence of values.
- Values stored in a list can be of any type such as string, integer, float, or list.
- Note!! Elements of a list are enclosed in square brackets, separated by commas.
- Unlike strings, lists are mutable, and therefore, one may modify individual elements of a list.

```
In [127]: subjects=['Hindi', 'English', 'Maths', 'History']
```




```
In [128]: temporary = subjects
```




```
In [129]: temporary[0] = 'Sanskrit'
          print(temporary)
          print(subjects)
```

```
['Sanskrit', 'English', 'Maths', 'History']
['Sanskrit', 'English', 'Maths', 'History']
```

```
In [130]: subjectCodes = [['Sanskrit', 43], ['English', 85] , ['Maths', 65], ['History', 36]]
          subjectCodes[1]
```

```
Out[130]: ['English', 85]
```

```
In [131]: print(subjectCodes[1][0],subjectCodes[1][1])
```

```
English 85
```

1.4.1 Heterogeneous List

```
In [132]: details = ['Megha Verma', 'C-55, Raj Nagar, Pitam Pura, Delhi - 110034', 9876543210]
```

1.5 List Operations

```
In [156]: list1 = ['Red', 'Green']
          list2 = [10, 20, 30]
```

1.5.1 Multiple Operator *

```
In [157]: list2 * 2

Out[157]: [10, 20, 30, 10, 20, 30]
```

1.5.2 Concatenation Operator +

```
In [158]: list1 = list1 + ['Blue']
          list1

Out[158]: ['Red', 'Green', 'Blue']
```

1.5.3 Length Operator len

```
In [159]: len(list1)

Out[159]: 3
```

1.5.4 Indexing & Slicing

```
In [160]: list2[-1]

Out[160]: 30

In [161]: list2[0:2]

Out[161]: [10, 20]

In [162]: list2[0:3:2]

Out[162]: [10, 30]
```

1.5.5 Function min & max

```
In [163]: min(list2)

Out[163]: 10

In [164]: max(list1)

Out[164]: 'Red'
```

1.5.6 Function sum

```
In [165]: sum(list2)
```

```
Out[165]: 60
```

1.5.7 Membership Operator: in

```
In [166]: 40 in list2
```

```
Out[166]: False
```

```
In [167]: students = ['Ram', 'Shyam', 'Gita', 'Sita']
          for name in students:
              print(name)
```

```
Ram
Shyam
Gita
Sita
```

1.5.8 Function list

- The function list takes a sequence as an argument and returns a list.

```
In [168]: vowels = 'aeiou'
          list(vowels)
```

```
Out[168]: ['a', 'e', 'i', 'o', 'u']
```

1.6 Built-in Functions on Lists

1.6.1 Function append

- The function append insert the object passed to it at the end of the list.

```
In [170]: a = [10, 20, 30, 40]
          a.append(35)
          print(a)
```

```
[10, 20, 30, 40, 35]
```

1.6.2 Function extend

- The function extend accepts a sequence as an argument and puts the elements of the sequence at the end of the list.

```
In [171]: a = [10, 20, 30]
          a.extend([35,40])
          print(a)
```

```
[10, 20, 30, 35, 40]
```

1.6.3 Function:count

- The function count returns the count of the number of times the object passed as an argument appears in the list.

```
In [172]: a = [10, 20, 30, 10, 50, 20, 60, 20, 30, 55]
          print(a.count(20))
```

```
3
```

1.6.4 Function pop

- The function pop returns the element from the list whose index is passed as an argument, while removing it from the list.

```
In [173]: a = [10, 20, 30, 10, 50, 20, 60, 20, 30, 55]
          a.pop(3)
          print(a)
```

```
[10, 20, 30, 50, 20, 60, 20, 30, 55]
```

1.6.5 Function remove

- The function remove takes the value to be removed from the list as an argument, and removes the first occurrence of that value from the list.

```
In [174]: a.remove(20)
          print(a)
```

```
[10, 30, 50, 20, 60, 20, 30, 55]
```

1.6.6 del Operator

- The del operator is used to remove a subsequence of elements (start:end:increment) from a list.

```
In [175]: a = [10, 20, 30, 20, 60, 20, 30, 55]
          del a[2:6:3]
          print(a)
```

```
[10, 20, 20, 60, 30, 55]
```

1.6.7 Function insert

- The insert function can be used to insert an object at a specified index. This function takes two arguments: the index where an object is to be inserted and the object itself.

```
In [176]: names = ['Ram', 'Sita', 'Gita', 'Sita']
          names.insert(2, 'Shyam')
          print(names)
```

```
['Ram', 'Sita', 'Shyam', 'Gita', 'Sita']
```

1.6.8 Function reverse

- The function reverse reverses the order of the elements in a list.

```
In [178]: names = ['Ram', 'Sita', 'Sita', 'Anya']
          names.reverse()
          print(names)
```

```
['Anya', 'Sita', 'Sita', 'Ram']
```

1.6.9 Function sort

- The function sort can be used to arrange the elements in a list in ascending order.

```
In [179]: names = ['Ram', 'Sita', 'Sita', 'Anya']
          names.sort()
          print(names)
```

```
['Anya', 'Ram', 'Sita', 'Sita']
```

```
In [180]: names = ['Ram', 'Sita', 'Sita', 'Anya']
          names.sort(reverse = True)
          print(names)
```

```
['Sita', 'Sita', 'Ram', 'Anya']
```

1.7 Problem: List of n terms of fibonacci series

```
In [3]: def fib(n):
            '''
            Objective: To return list of n terms of fibonacci series.
            Input Parameter: n - numeric
            Return Value: numeric
            '''
            '''
```

```
        Approach:
            Create a list with 0 and 1 as first two numbers of fibonacci series.
            For each subsequent number, append sum of previous two numbers to
            the list.
        '''
        if n <=0:
            return None
        elif n ==1:
            return [0]
        elif n == 2:
            return [0, 1]
        else:
            resList = []
            a, b = 0, 1
            resList.append(a)
            resList.append(b)
            count = 3
            while  count <= n:
                c = a + b
                resList.append(c)
                a = b
                b = c
                count += 1
            return resList

    def main():
        '''
        Objective: To print n terms of fibonacci series based on user input
        Input Parameter: None
        Return Value: None
        '''
        num = int(input('Enter no. of terms:'))
        result = fib(num)
        print(result)

    if __name__ == '__main__':
        main()
```

```
Enter no. of terms:6
[0, 1, 1, 2, 3, 5]
```