Strings

Topics covered

- Defining String
- Indexing, Slicing
- Various methods
  - strip
  - split
  - lower, upper
  - find
  - join
  - count
  - replace

Defining string

- Created by enclosing characters in quotes - single, double or triple quotes.
- Python treats single quotes the same as double quotes

In [1]:
```
string_1 = "Hello Python. "
string_2 = 'Welcome to programming'
```

String Operators

- " + " : Concatenation - Adds multiple strings
- " * " : Repetition - repeat multiple string
- Membership check operator - check if character or substring is present in the string
  - "in": Return `True` if character/substring exsist
  - "not in": Return `False` if character/substring exsist

In [2]:
```
# Concatenation
string_1 = "Hello Python."
string_2 = 'Welcome to programming'

print (string_1 + string_2)
print (string_1+ " " + string_2 + " world.")
```

```
Hello Python.Welcome to programming
Hello Python. Welcome to programming world.
```

In [3]:
```
# Repetition
string_1 = 4*"Hello"

print (string_1)
```

```
HelloHelloHelloHello
```

In [4]:
```
# Membership
string_1 = "Hello Python "

print ("Hello" in string_1) #Remember that it is case sensitive
print ("hello" in string_1)
print ("hell" in string_1)
```

```
True
False
False
```

Accessing string using slicing and indexing

- use the square brackets for slicing along with the index or indices to obtain your substring.
- Indexing starts from 0.
- variable name[a:b:c]
  - here a is starting index and ends at b-1 index (note index starts from 0) c is step size, default is 1
- negative indexing - last element is also index as -1 going through -2,-3.. as we go towards left

In [5]:
```
string_1 = "Hello Python"
print (string_1[0])        # print first character
print (string_1[-1])
print (string_1[0:2])       # print character at 0 and 1 index
print (string_1[-1:])       # printing from backward. print character at last index
print (string_1[1:-2])      #
print (string_1[::2])       # printing every alternate digit
print (string_1[::-1])      # printing reverse string
print (type(string_1[0]))
```

```
H
n
He
n
ello Pyth
HloPto
nohtyP olleH
<class 'str'>
```

String Methods

- `capitalize()` - return string with first letter in upper case
- `upper()` - return string with all letter in upper case
- `lower()` - return string with all letter in lower case
- `count(str)` - count occurrence of 'str' in string
- `find(str)` - find 'str' in string. If found: return start index of first occurrance str, else return -1
- `index(str)` - same as find, but raises exception when str is not found
- `join(seq)` - join string represented as sequence into a string with a separator string between each string
- `len(str)` - returns length of string
- `replace()` - returns a copy of the string in which the occurrences of old have been replaced with new, optionally restricting the number of replacements to max.
- `split(str)` - return a list which has string of words separated by str
- `strip(str)` - remove occurrence of str from the beginning and the end of the string

In [6]:
```
# Capitalize string - Convert first character to upper case and other to lower case
string_1 = "hello Python, the string handling features are awesome"
print ("Actual string: ", string_1)

print ("Capitalize: ",string_1.capitalize())           # First character of string is changed to upper case
```

```
Actual string:  hello Python, the string handling features are awesome
Capitalize:  Hello python, the string handling features are awesome
```

In [7]:
```
# Convert to upper case
string_1 = "Hello Python, the string handling features are awesome"
print ("Actual string: ", string_1)

print ("Upper Case: ",string_1.upper())    # All characters are changed to upper case
# print string_1
```

```
Actual string:  Hello Python, the string handling features are awesome
Upper Case:  HELLO PYTHON, THE STRING HANDLING FEATURES ARE AWESOME
```

In [8]:
```
# Convert to lower case
string_1 = "Hello Python, the string handling features are awesome"
print ("Actual string: ", string_1)

print ("Lower case: ",string_1.lower())    # All characters are changed to lower case
# print string_1
```

```
Actual string:  Hello Python, the string handling features are awesome
Lower case:  hello python, the string handling features are awesome
```

In [9]:
```
# Count occurrances
string_1 = "Hello Python, the string handling features are awesome"
print ("Actual string: ", string_1)

print ("Count of 'P': ",string_1.count('P'))      # Count occurrances of character 'P' in string
print ("Count of 'th': ",string_1.count('th'))    # Count occurrances of string 'th' in string
```

```
Actual string:  Hello Python, the string handling features are awesome
Count of 'P':  1
Count of 'th':  2
```

In [10]:
```
# Index of occurrance using find
string_1 = "Hello Python, the string handling features are awesome"
print ("Actual string: ", string_1)

print ("Index of occurrance of 'lo': ",string_1.find('lo'))
print ("Index of occurrance of 'th': ",string_1.find('th')) # Index of first occurrance
#If python is not able to find the string mentioned it will return -1
```

```
Actual string:  Hello Python, the string handling features are awesome
Index of occurrance of 'lo':  3
Index of occurrance of 'th':  8
```

In [11]:
```
# Index of occurrance using index
string_1 = "Hello Python, the string handling features are awesome"
print ("Actual string: ", string_1)

print ("Index of occurrance of 'lo': ",string_1.index('lo'))
print ("Index of occurrance of 'th': ",string_1.index('th')) # Index of first occurrance
#If python is not able to find the index of the string mentioned it will throw an exception in this case
```

```
Actual string:  Hello Python, the string handling features are awesome
Index of occurrance of 'lo':  3
Index of occurrance of 'th':  8
```

In [12]:
```
# Joining each element of string

letter = "e"
word = "lvl"
new_word = letter.join(word)
print (new_word)

print ("Join characters with '-': "+ "-".join("world"))
print ("Join characters with ',': "+ ",".join("world"))
```

```
level
Join characters with '-': w-o-r-l-d
Join characters with ',': w,o,r,l,d
```

In [13]:
```
# length, minimum value and maximum value
string_1 = "Hello Python,the string handling features are awesome"
print ("Actual string: ", string_1)

print ("length of string_1 is: ",len(string_1))   # returns length of string
print ("Maximum value is: ",max(string_1))        # returns largest element of string
print ("Minimum value is (space which is not visible): ",min(string_1)) #returns space since it has lowest ASCII value i
#Remove the spaces in the string to see what happens.
```

```
Actual string:  Hello Python,the string handling features are awesome
length of string_1 is:  53
Maximum value is:  y
Minimum value is (space which is not visible):
```

In [14]:
```
# Replace
string_1 = "Hello Python, the string handling features are the best"
print ("Actual string: ", string_1)

#string.replace returns a copy of the string and doesn't alter the orignal string.
print ("Replaced occurrance of 'th': ", string_1.replace('th','!!',1))   # replace only first occurrance
print ("Replaced occurrance of 'th': ", string_1.replace('th','!!',3))   # replace all occurrances
print (string_1) #The orignal string remains the same
```

```
Actual string:  Hello Python, the string handling features are the best
Replaced occurrance of 'th':  Hello Py!!on, the string handling features are the best
Replaced occurrance of 'th':  Hello Py!!on, !!e string handling features are !!e best
Hello Python, the string handling features are the best
```

In [15]:
```
# split string
string_1 = "Hello Python, the string handling features are awesome"
print ("Actual string: ", string_1)

string_to_list = (string_1.split())
print (string_to_list)
print (len(string_to_list))

string_to_list = string_1.split(" ",0) #The list contains only the zeroth element
print (string_to_list)
print (len(string_to_list))
print (string_1)
```

```
Actual string:  Hello Python, the string handling features are awesome
['Hello', 'Python,', 'the', 'string', 'handling', 'features', 'are', 'awesome']
8
['Hello Python, the string handling features are awesome']
1
Hello Python, the string handling features are awesome
```

In [16]:
```
# strip unwanted characters from beginning and end of string
string_1 = "0 00Hello Python, the string handling features are awesome00"
print ("Actual string: ", string_1)
print (len(string_1))
print (string_1.strip())
print (len(string_1.strip()))
print (string_1.strip("0"))
print (len(string_1.strip("0")))
print (string_1.strip("0 "))
#Note the addition of a space after '0' in the last strip statement. Also, the orignal string is not at all altered duri
#print (string_1)
```

```
Actual string:  0 00Hello Python, the string handling features are awesome00
60
0 00Hello Python, the string handling features are awesome00
60
 00Hello Python, the string handling features are awesome
57
Hello Python, the string handling features are awesome
```