Strings in Python

Strings in Python

- Strings are amongst the most popular types in Python.
- Strings are identified as a contiguous set of characters represented using the quotation marks.
- Python allows for either pairs of single or double quotes.
- Subsets of strings can be taken using the slice operator ([ ] and [:]) with indexes starting at **0** in the beginning of the string and working their way to **end -1**.
- The plus (+) sign is the string concatenation operator and the asterisk (*) is the repetition operator.

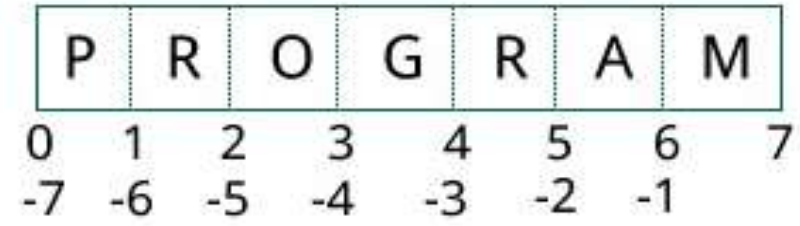
```
str = 'Hello World'
print(str)              # Prints complete string
print(str[0])           # Prints first character of the
string
print(str + 'Bye')      # Prints concatenated string
print(str * 2)          # Prints string two times
```

Strings in Python

Accessing Values in Strings

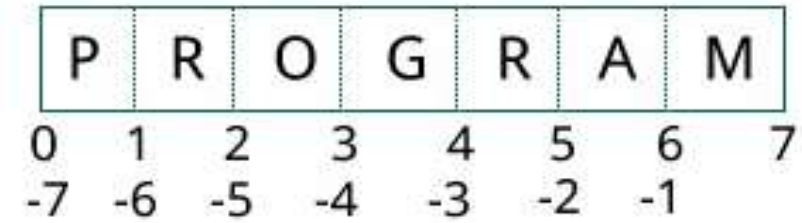
To access substrings, use the square brackets for slicing along with the index or indices to obtain your substring. Indexing allows negative address references to access characters from the end of the String, e.g. -1 refers to the last character, -2 refers to the second last character and so on.

```
str =  'PROGRAM'
print(str[0])        #output P
print(str[5])        #output A
print(str[-1])       #output M
print(str[-3])       #output R
print(str[-7])       #output P
```



Strings in Python

String Slicing: To access a range of characters in the String, method of slicing is used. Slicing in a String is done by using a Slicing operator [:] (colon).

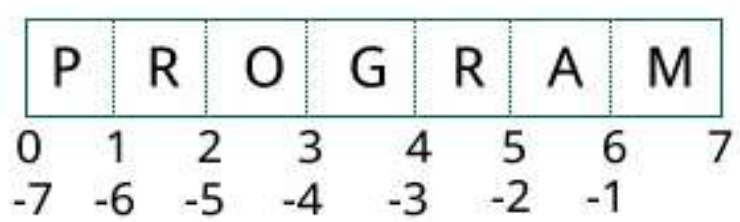



```
str =  'PROGRAM'
print(str[1:4])        #output ROG
print(str[2:])         #output OGRAM
print(str[:])          #output PROGRAM
print(str[2:-2])       #output OGR
print(str[-6:-2])      #output ROGR
print(str[-2:-6])      #No output
print(str[::-1])       #output MARGORP
```

Strings in Python

Deleting/Updating a String:

- In Python, updation or deletion of characters from a String is not allowed.
- This is because Strings are **immutable**, hence elements of a String cannot be changed once it has been assigned.
- Only new strings can be reassigned to the same name.
- Although deletion of entire String is possible with the use of a built-in del keyword.




```
str1 = 'PROGRAM‘
str1[1]='A'                 #Error
str1='PRAGRAM'              #This is valid
str1=str1[0:3]+str1[4:7]    # PRARAM
del str1                    # deletion of string
print(str1)                 #str1 is not defined
```

Strings in Python

Escape Characters:

A list of escape or non-printable characters that can be represented with backslash notation.

```
print ("RCC\nIIT");      #output RCC then IIT in next line
print ("RCC\tIIT");      #output RCC  IIT
print ("RCC\\IIT");      #output RCC\IIT
```

Special Operators on String

```
name='RCCIIT‘
addr='Kolkata‘
result=name+addr          # Concatenation   Output = RCCIITKolkata
result=name*2             # Repetition      Output = RCCIITRCCIIT
member='I' in name        #Membership
print(member)             # Output=True
member='I' not in name    #Non-membership
print(member)             # Output=False
```

Strings in Python

String Formatting in Python using %: The % format is similar to that of 'printf' in C programming language.

```
name='RCCIIT'
print("College name is %s." %(name))            #String
roll=20
print("Roll number is %d." %(roll))             #Integer
wt=62.5
print("Weight is %f KG." %(wt))                 #Float
number=45
print("Octal value is %o." %(number))          #Octal
print("Hexadecimal value is %x." %(number))  #Hexadecimal
```

Strings in Python

String Functions and Methods

len(str1) # Returns the length of the string.

---

str1.isdigit() # Checks whether the string consists of digits only.

---

str1.count(sub, start, end)

Returns no. of occurrences of substring **sub** in the range [start, end].

---

str1.find(sub, start, end)

Return index if substring **sub** occurs in string, otherwise return -1.

---

str1.isalpha()

Returns true if all characters in the string are alphabets.

---

str1.replace(old, new[, max])

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.

THANK YOU