8/Dec/2023

Evergreen
Page No.
Date: / /20

Chapter-6

Strings in Python

Strings

In python, a consecutive sequence of characters, which are enclosed or surrounded by single (' ') or double (" ") quotes, is known as a string.

Creating Strings

String are among the most popular data types in python. We can create them by simply enclosing character in quotes (single, double, triple). Python treats single quotes the same as double quotes.

⇒ Sometimes, triple quotes can be used in python but they are generally used to represent multiline strings and doc strings.

- Creating strings is as simple as assigning a value to a variable.

For ex:- Str1 = "Hello world"
Str2 = "Python Programming"

Empty String

An Empty string is a string without any characters inside, having length zero. When we print an empty string a

blank space gets displayed.

Ex:- >>> str = " "
>>> print (str)

>>> str = ' '
>>> print (str)

>>>

Multiline strings

Multiline strings are represented using triple single(''' ''') or triple double (""" """) quotes.

Accessing Character (Indexing) in a string

We can access individual characters of the string by using the subscript or index value, which is known as Indexing.

⇒ Each character in a string has an index value. Indexing starts at 0 and must be an integer.

Ex:- Str = "hello world!"

h	e	l	l	o		w	o	r	l	d	!
0	1	2	3	4	5	6	7	8	9	10	11

- We can't use float or other types for index as this will result in Type Error.
- Trying to access out of index range will raise an Index Error.

→ We can start index value either from the left or from the right. Left or positive index starts from -1 to the left of the string.

for ex:- var1 = 'Hello World!'

Positive Index -	0	1	2	3	4	5	6	7	8	9	10	11
Var1 -	H	e	l	l	o		W	o	r	l	d	!
Negative Index -	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

(a) Iterating through string using for loop
This is done using the indexes and iterating over it results in displaying the elements of a string, one character at a time.

for ex:- Str = 'Computer Science'
for i in str:
    print(i)

C
o
m
p
u
t
e
r

S
C
I
E
N
C
E

Evergreen
Page No.
Date : / /20

(b) Iterating through String using while loop

Like for, while loop can also be used for String traversal. Each element of the string is accessed one by one and while loop iterates till the end of the inputted string, which can be determined using standard library function, len( ) function.

Special String Operations

String can be manipulated using operators like concatenation, repetition (*) & membership operators like in and not in.

Concatenation → String 1 + String 2
Repetition → String 1 * x
Membership → in, not in
Comparison → Str 1 > Str 2
Slicing → String (range)

→ Concatenation Strings
It refers to creating a new string by adding two strings. The + operator joins or concatenates strings written on both sides of the operator and creates a new string.
- To concatenate means To join.

Ex:- >>> 'Hello' + 'world'
'Hello World'

⇒ You cannot add 'str' and 'int' objects. To use '+' operator, data types should be the same.

Ex:- 6 + 3 = 9 → addition
'6' + '3' = '63' → Concatenation

Evergreen
Page No.
Date: / /20

→ Replicating Strings
The * operator creates a new string by repeating multiple copies of the same thing

ex:- 2 * '2'
'22'

Membership operators

Python offers two membership operators for checking whether a particular character exists in the given string or not.
- These operators are 'in' and 'not in'.

→ 'in' operator:- It returns true if a character / substring exists in the given string.
→ 'not in' operator:- It returns true if a character does not exist in the given string.

ex:- >>> 'H' in 'Hello'
True
>>> 'y' in 'Hello'
false

☆ To use membership operator in strings, it is required that both the operands used should be of string type.

⇒ Comparison operators
- You can use relational / Comparison operators (>, <, >=, <=, ==, !=) to compare two strings.
- python compares strings using ASCII, rather UNICODE value of the characters. Unicode values are also called

ordinal values.

- Python starts by comparing the first element from each sequence (character-by-character comparison). If they are equal, it goes on to the next element.
- Subsequent elements are not considered (even if they are big)

⇒ String slicing

- Slicing is used to retrieve a subset of values.
- This extracted substring is termed as 'slice'.
- Chunk of characters can be extracted from a string using 'slice operator with two indices in square brackets separated by colon([:]).

Syntax:

String_name [start : end: step]

Strings are Immutable:-

- Strings are immutable means that the content of the string cannot be exchanged after it is created.
- Python does not allow the programmer to change a character in a string.

String methods and Built-in Functions

- Python provides several built-in functions associated with strings.
- This function allows us to easily modify and manipulate strings
- This function are pre-defined in Python Programming langudge and are readically available for use.

Syntax:

String_object.functionName()

1. Len()- This method returns the length of the string
Syntax:
Len(str)
☆ Here, str is the string

2. Capitalize()- This method returns the exact copy of the strings with the first letter is uppercase.
Syntax:
Str.Capitalize()
Here, str is the string
Note:- If the string has its first character as capital, then it returns the original string.

3. Split()- The split() method breaks up a string at the specified separator and returns a list of substrings
Syntax:
str.split ([Separator, [max split]])
⇒ The split() method takes a maximum of 2 parameters:

- Separator (optional):- The separator is a delimiter. The string splits at the specified. If the separator is not specified, any whitespaces (space, newline, etc) string is a separator.

- maxsplit (optional) - The maxsplit defines the maximum no. of splits. The default value of maxsplit is -1, which means no limit on the no. of splits.
☆ If separator is not ~~defined~~ specified, so by default, space is a string separator.

Evergreen
Page No.
Date : / /20

4. replace( ) - This function replaces all the occurrences of the old string with the new - string.
Syntax:- Str.replace(old, new)

5. find( ) - This function is used to search the first occurence of the substring in the given string.
Syntax:-

Str. find (sub, start, end)

- Sub: the substring which needs to be searched in the given string
- Smart: Starting position where substrings is to be checked within the string
- End: End position is the index of the last value for specified range.

6. index( ) - This function is quite similar to find ( ) function it also searches the ~~function~~ first occurence and returns the lowest index of the substring if it is found in the given string but raises an exception if the substring is not present in the given string.
Syntax:

index (substring, start, end)

7. isalpha( ) - This function checks for alphabets in an inputted string. It returns True if the String contains only letters, otherwise returns false.
Syntax:

Str. isalpha( )

8. isalnum( )- The isalnum ( ) method returns True if all the characters are alphanumeric, i.e. alphabet letters (A-Z, a-z) and numbers (0-9).

☆ Examples of characters that are not alphanumeric: (space), !, #, %, &, ?, etc.

Syntax:

String.isalnum( )

9. isdigit( )- This function returns True if the string contains only digits, otherwise False.

Syntax:

Str.isdigit( )

10. title( )- This function returns the string with first letter of every word in the string in uppercase and rest in lowercase.

Syntax:

Str.title( )

11. count( )- This function returns the number of times substring str occurs in the given string.

* If we do not give start index and end index, then searching starts from index 0 and ends at length of the string.

Syntax:

Str.count(substring, start, end)

12. lower( ):- This function converts all the uppercase letters in the string into lowercase.

Syntax:

Str.lower( )

Evergreen
Page No.
Date : / /20

Convert uppercase letters only to lowercase. If already in lowercase, then it will simply returns the string.

13. islower( )- This function returns True if all the letters in the string are in lowercase.
Syntax:
Str. islower( )

14. upper( )- This function converts lowercase letters in the string into uppercase.
Syntax:
Str. upper( )

15. isupper( )- This function returns True if the string is in uppercase.
Syntax:
Str. isupper( )

16. lstrip( )- This function returns the string after removing the space(s) from the left of the string.
Syntax.
Str. lstrip( )
or
str. lstrip(chars)

⇒ Chars(optional)- a string specifying the set of characters to be removed from the left of the string until left character of the string mismatches.

17. ~~rstrip(self)~~ rstrip( )- This function removes all the trailing whitespaces from the right of the string.

Evergreen
Page No.
Date : / /20

Syntax:

rstrip()
or
Str.rstrip(chars)

Chars(optional) - a String specifying the set of characters to be removed from the right.

18. strip() - This function returns the string after removing the spaces both on the left & the right of the string.
Syntax:

Str.strip()

19. isspace(): - This function returns True if the string contains only whitespace characters, otherwise returns false.
Syntax:

Str.isspace()

20. istitle() - The istitle() function doesn't take any arguments. It returns True if the string is properly "title-cased"; else returns false if the String is not a "title-cased" string or an empty string.
Syntax:

Str.istitle()

21. join(sequence) - This function returns a string in which the string elements have been joined by a string separator.
Syntax:

Str.join(sequence)

Sequence - Join() takes an argument which is of sequence data types, capable of returning its element one at a time.

Evergreen
Page No.
Date: / /20

22. Swapcase:- This function converts and returns all uppercase characters into lowercase and vice versa of the given string. It does not take any argument.
Syntax:

```
Str.swapcase()
```

23. partition(separator) - partition function is used to split the given string using the specified separator and returns a tuple with three parts:
- substring before the separator;
- separator itself;
- a substring after the separator;

Syntax:

```
Str.partition(separator)
```

Separator: This argument is required to separate a string. If the separator is not found, it returns the string itself followed by two empty strings within parentheses as tuple.

24. endswith() - This function returns True if the given string ends with the specified else returns false.
Syntax:

```
Str.endswidth(substr)
```

25. startswith() - This function returns True if the given string starts with the specified substring, else returns false.
Syntax:

```
Str.startswith(substr)
```

Evergreen
Page No.
Date : / /20

Other functions

- python provides two functions for character encoding:

⇒ ord() - This function returns the ASCII / Unicode (ordinal) of the character.

⇒ chr() - This function returns the character represented by the inputted Unicode / ASCII number.

Ques - WAP to input a string and count uppercase letter, lowercase letters, space and other characters.

```
S = input("Enter a string")
Ca = 0
Cs = 0
Cd = 0
Csp = 0
for i in S:
    if(i.isalpha()):
        Ca += 1
    elif(i.isdigit()):
        Cd += 1
    elif(i.isspace()):
        Cs += 1
    else:
        Csp += 1
print("No. of Alphabets:", Ca/n,
      "No. of Spaces:", Cs/n,
      "No. of digits:", Cd/n,
      "No. of Special characters:", Csp)
```

Evergreen
Page No.
Date : / /20

Ques - WAP to input a string and concadinate all the vowels in another string.

```
S = input ("Enter a string:")
S1 = ''
for x in S:
    if x in "aeiou AEIOU":
        S1 += x
print ("Vowels in string:", S1)
```

Ques - WAP ~~to check~~ check whether the string is a palindrom or not.

```
S = input ("Enter a string:")
S1 = ""
l = len(S)
for i in range (l-1, -1, -1):
    S1 += S[i]
if (S == S1):
    print ("String is palindrom.")
else:
    print ("String is not palindrom.")
```

Ques - WAP to input a string and count the no. of words having first letter vowel.

```
S = input ("Enter a string")
L = S.split()
C = 0
for in in L:
    if (i[0] in "aeiouAEIOU"):
        C += 1
print ("No. of words with first letter vowel:", C)
```

Evergreen
Page No.
Date : / /20

Ques- WAP to input a string and find words with length less than equal four.

```
S = input ("Enter a string")
L = S.split()
C = 0
for i in L:
    if (len(i) <=4):
        Ct = 1
print ("No. of words with length less than 4", C)
```

Ques- WAP that reads a string and displays the longest substring of the given string.

```
Str1 = input ("Enter a string:")
word = Str1.split()
maxlength = 0
maxword = ' '
for i in word:
    x = len(i)
    if x > maxlength and i.isalpha() == True:
        print(i)
        maxlength = x
        maxword = i
print ("Substring with maximum length is:", maxword)
```

Evergreen
Page No.
Date : / /20

Ques - WAP to remove vowels from a string

```
Str1 = input ("Enter string")
Str2 = ""
for i in range (len (Str1)):
        if Str[i] not in "ieouaAOIUE":
            Str2 = Str2 + Str1[i]
print ("original string:", Str1)
print ("New string is:", Str2)
```