sandeepsuryaprasad / **python_tutorials** Private

Code Issues Pull requests Actions Projects Security Ins

master python_tutorials / 1_Strings / **_strings.py** / Go to file ...
Jump to

Sandeep Suryaprasad fixed typo Latest commit 03cc4cc 1 minute ago **History**

1 contributor

216 lines (169 sloc) | 6.92 KB Raw Blame

```python
# Working with Strings.
"""
All Variables should in Lower Case. If there are more than one word in the Va
then we separate with under scores. And this is PYTHON CONVENTION
"""
# ================================================================
# Difference ways of constructing a string object
word = 'Hello World'
word = str('Hello world')
print(word)
word = ""   # Zero Length string or an empty string

"""
We can use both single and Double Quotes for the Strings.
If you have single Quotes in the actual String, we can represent the original
If the String actual String contains Double Quotes, we can use single Quotes
"""

message = "Welcome to Python's world"
print(message)

message = 'Welcome to Pythons"s world'
print(message)

# Both single and double quotes in single sting
message = """ Hello world! "Hi" and 'Bye' """
print(message)
```

```python

message = ''' Hello world! "Hi" and 'Bye' '''
print(message)

# We can use Escape Charater as well
message = 'Welcome to Python\'s world'
print(message)

message = "Welcome to Python\"s world"
print(message)

# We can use either double backslash or prefix 'r' which stands for raw strin
print("C:\\testing\\newfolder")
print(r"C:\testing\newfolder")

# use Triple Quotes to represent a Multi-Line String
multi_line_string = '''Hello There..
Welcome to Python tutorials'''
print(multi_line_string)

multi_line_string = """Hello There..
Welcome to Python tutorials"""
print(multi_line_string)

# =========================================================
my_message = 'Hello World'
# ========================================================
# type is an inbuilt function, which returns the datatype of the
# variable or an object
print(type(my_message))
"""
1. my_message is of type str and its value is "Hello world"
2. my_message is an instance of class str
3. my_message is a string object with value "Hello world"
4. Every object has "identity", "type" and "value".
5. my_message is a label which points to a string object.
"""

"""
dir is an inbuilt function, which returns a list of attributes
that are attached to the object.
"""
print(dir(my_message))

"""
```

```
we can get information about a function using in-built function help()
e.g. help("hello".upper)
help("hello".split)
"""

# String Functions
# NOTE: ALL STRING FUNCTIONS RETURNS A NEW STRING AND WILL NOT MODIFY OR MUTA

print(len(my_message))      # Prints the Length of the String. Index starts f
print(my_message.upper())   # Prints the String in Upper Case
print(my_message.lower())   # Prints the String in Lower Case
print(my_message.count('l'))    # Prints number of occurances of the letter '
print(my_message.count('Hello'))    # Prints number of occurances of the word
print(my_message.find('l'))     # Prints the index of first occurance of the
print(my_message.find('World'))     # Prints the index of first occurance of
print(my_message.find('Universe'))      # Prints -1.
print("today is beautiful day".rfind("day"))    # Prints 20
print(my_message.replace('World', 'Universe'))  # Prints 'Hello Universe'
print(my_message.split())   # Splits the string based on white space and retu

s = 'This is my string'
print(s.split('s'))

info = '560100,Bangalore,KA'
parts = info.split(',')

# space-delimit data
line = "Jun 03 22:58:18 farnsworth sshd[29386]: Failed password for invalid u
parts = line.split()

# comma-delimit data
record = "2020-01-03,IN,India,SEARO,0,0,0,0"
parts = record.split(",")

# pipe-delimit data
record = "2017-06-01T07:43:07.481Z|host1099-99.testnetwork.local|filebeat|log

print(my_message.startswith('Hello'))
print(my_message.endswith('World'))

my_string = '***************Hello world==================='
print(my_string.rstrip('='))    # prints ***************Hello world
print(my_string.lstrip('*'))    # prints Hello world===================
print(my_string.strip('=*'))    # Prints Hello world

```

```
message = 'hello'
'-'.join(message)   # Joins each character of the string using '-'
','.join(message)   # Joins each character of the string using ','

# len is an inbuilt method in python and its not an attribute of str class!
print(len(my_message))  # Prints the length of the string.

# using "in" operator to check if the character is present in the string
greeting = "hello world"
"d" in greeting # (returns True)
"y" in greeting # (returns False)
#===============================================================================

# String Slicing
# my_message[start:stop:step]
my_message = 'Hello World'

#  H    e     l     l    o       W    o    r    l    d
#  0    1     2     3    4    5   6    7    8    9   10
# -11  -10   -9    -8   -7   -6  -5   -4   -3   -2   -1

print(my_message[0])        # Prints the character present at the 0th index
print(my_message[10])       # Prints the character present at the 10th index
print(my_message[0:5])      # Prints Hello. Upto 5th character, but NOT INCLU
print(my_message[:5])       # Prints Hello.
print(my_message[6:])       # Prints World

# Negative Indexing
print(my_message[-1])      # Prints 'd'
print(my_message[-11])     # Prints 'H'
print(my_message[-4:])     # Prints 'World'
print(my_message[0:-6])    # Prints 'Hello'
print(my_message[2:-3])    # Prints 'llo Wo'

# Step
print(my_message[::2])      # Prints Every Alternate Characters
print(my_message[::-2])     # Prints Every Alternate Characters in reverse or
print(my_message[::-1])     # Prints the string in reversed order

# Print extension of the filename
name = 'Youtube.txt'
print(name[-3:])

# Print only filename
print(name[:-3])
```

```python

# Printing only protocol in url
url = 'https://google.com'
print(url[:5])

# Print only domain
print(url[7:])

# ==========================================================
# String Concatination
greeting = 'Hello'
name = 'Steve'
print(greeting, name)

print('Python '+str(2019))      # 2019 should be converted to String if using
print('Python' + ' 2019')
print('Python', 2019)       # Comma is used for concatinating two string of d

# '+' is used for concatinating two objects of same datatype
print(greeting+', '+name)

# Repeats the string 5 times
print('Hello ' * 5)

# String Conversions
x = 26
print(str(x))   # prints '26'

# String formatting.
name = "Steve"
age = 26
print("Hello {} you are {} years of age".format(name, age))

print("Hello {1} you are {0} years of age".format(name, age))

# using "f" strings
print(f"Hello {name} you are {age} years of age")

# Producing Structured Output
fname = "Steve"
lname = "Jobs"
pay = 2000

# Right Justification
print(f"{fname:>10} {lname:>10} {pay:>10}")
```

```python

# Left Justification
print(f"{fname:>5} {lname:>5} {pay:>10}")

# Center Justification
print(f'{fname:^10} {lname:^10} {pay:^10}')

# Printing the Headers
print(f'{"fname":>10} {"lname":>10} {"pay":>10}')
```