Python

Strings

Python String

- 1. Python String with String Functions and String Operators
- 2. Introduction to Python String
- 3. Declaring a Python String
- 4. Using quotes inside a Python string
- 5. Spanning a string across lines
- 6. Accessing a Python String
- 7. Python String Concatenation
- 8. Python String Formatters
- 9. Python Escape Sequences
- 10. Python String Functions
- 11. Python String Operations
  - a. Comparison
  - b. Arithmetic
  - c. Membership
  - d. Identity
  - e. Logical
- 12. Python String – Conclusion

Python String

- **Introduction to Python String**
- **A Python string is a sequence of characters.**
- **There is a built-in class ‘str’ for handling Python string.**
- **You can know the data type of this class with the type() function. >>> >>> type("kmit ngit")**
- **<class 'str'>**
- **>>> type('python welcome')**
- **<class 'str'>**
- **>>> a = "Welcome to KMIT"**
- **>>> PRINT(a)**
- **Traceback (most recent call last):**
- ** File "<stdin>", line 1, in <module>**
- **NameError: name 'PRINT' is not defined**
- **>>> print(a)**
- **Welcome to KMIT**
- **>>> a = 'Welcome to NGIT'**
- **>>> print(a);**
- **Welcome to NGIT**

Python String

- **Introduction to Python String**
- **As far as language syntax is concerned, there is no difference in**
- **single or**
- **double quoted string.**
- **Both representations can be used interchangeably.**
- **However, if single is a part of the string itself, then the string must be placed in double .**
- **Or double quote is a part of the string itself, then the string must be placed in single quotes .**

Python String

- Using quotes inside a Python string
- Since we delimit strings using quotes, there are some things you need to take care of when using them inside a string.
- >>> a=“Welcome to “KMIT""
- SyntaxError: invalid syntax
- If you need to use double quotes inside a string, delimit the string with single quotes.
- >>> a=‘Welcome to “KMIT“’
- >>> print(a)
- Welcome to “KMIT“
- And if you need to use single quotes inside a string, delimit it with double quotes.
- >>> a=“Welcome to ‘KMIT”
- >>> print(a)
- Welcome to ‘KMIT’

Python String

- https://data-flair.training/blogs/python-string/
- Spanning a string across lines
- >>> a="""hello
- ... kmit"""
- >>> a
- 'hello\nkmit'
- >>> print(a)
- Hello
- kmit

Python String

- >>> print("""hello
- ... NGIT""")
- hello
- NGIT
- To supress the new line use the back slash \
- >>> p = a=“””Hello \
- ... NGIT and KMIT"""
- >>> print(p)
- Hello NGIT and KMIT
- Without single back slash when we press enter it gives error.
- >>> "hello
-   File "<stdin>", line 1
-     "hello

Python String

- **Displaying a single character of a string**
- To display a single character from a string, put its index in square brackets.
- Indexing begins at 0.
- >>> a = "Bat“
- Index Range can be given
- >>> print(a[1:2])
- a
- >>> print(a[1:3])
- at
- >>> print(a[1:30])
- at
- >>> print(a[1:])
- at

Python String

- **Accessing a Python String**
- **Individual characters in a string are immutable; it can’t be changed.**
- **>>> a = "Bat"**
- **>>> print(a)**
- **Bat**
- **>>> a[0] = "P"**
- **Traceback (most recent call last):**
- ** File "<stdin>", line 1, in <module>**
- **TypeError: 'str' object does not support item assignment**

Python String

- >>> a = "Bat“
- >>> print(a[1:1])

- >>> print(a[2])
- t
- >>> print(a[:2])
- Ba
- >>> print(a[:3])
- Bat
- >>> print(a[:])
- Bat
- >>> print(a[-3:])
- Bat

Python String Concatenation

- **Python String Concatenation**
- **Concatenation is the operation of joining stuff together. Strings can be joined using the concatenation operator +.**
- **>>> a = "Welcome to"**
- **>>> b = " KMIT"**
- **>>> C = " & NGIT"**
- **>>> a+b+c**
- **Traceback (most recent call last):**
- ** File "<stdin>", line 1, in <module>**
- **NameError: name 'c' is not defined**
- **>>> print(a+b+c)**
- **Traceback (most recent call last):**
- ** File "<stdin>", line 1, in <module>**
- **NameError: name 'c' is not defined**
- **>>> a+b+C**
- **'welcome to KMIT & NGIT'**
- **>>> print(a+b+C)**
- **welcome to KMIT & NGIT**

Python String Concatenation

- Python String Concatenation
- `>>> a = 'Welcome to '`
- `>>> b = ' KMIT'`
- `>>> a+b`
- `'Welcome to  KMIT'`
- `>>> print(a+b)`
- `Welcome to  KMIT`
- `>>> c= ' & NGIT'`
- `>>> a+b+c`
- `'Welcome to  KMIT & NGIT'`

Python String

- Another example.
- >>> a = '10'
- >>> a*2
- '1010'
- >>> print(a*3)
- 101010
- >>> print(a/2)
- Traceback (most recent call last):
-   File "<stdin>", line 1, in <module>
- TypeError: unsupported operand type(s) for /: 'str' and 'int'

Python String

- >>> "10"+"20"
- '1020'
- >>> "30“ * 2
- '3030'
- >>> '10'+10
- Traceback (most recent call last):
-   File "<stdin>", line 1, in <module>
- TypeError: can only concatenate str (not "int") to str

Python String

- Python String Formatters
- Sometimes, you may want to print variables along with a string. You can either use commas, or use string formatters for the same.
- >>> city = 'Hyderabad'
- >>> print('Age ', 25,' City ',city)
- Age 25 City Hyderabad

Python String

- **f-strings:**
- The letter ‘f’ precedes the string, and
- the variables are mentioned in curly braces in their places.
- >>> name = 'Anand‘
- >>> print(**f** 'Hello **{**name**}** how are you?')
- Hello Anand how are you?

Python String

- **% operator-**
- The % operator is used to substitute the variables in a string.
- %s is for string.
- What follows the string is the operator and variables in parentheses.
- a = 'Ram'
- b = 'Hyderabad'
- print('Hi %s r u in %s' %(a,b))
- Save it in a file with extension .py
- Run: press F5
- Hi Ram r u in Hyderabad

Python String Functions

- Python provides us with a number of functions that we can apply on strings or to create strings.

a) len()- The len() function returns the length of a string.

   i. len(a)

- **str()-** This function converts any data type into a string.
- >>> str(2+3j)
- >>> str(‘red’+’pink’+’black’)
- **lower() and upper()-** These methods return the string in lowercase and uppercase, respectively.
- a.lower() or a.upper()

Python String Functions

- **strip()- It removes whitespaces from the beginning and end of the string.**
- **s ='   Book   '**
- **print(len(s))**
- **print(s.strip())**
- **print(len(s))**
- **print(len(s.strip()))**

- **isdigit()- Returns True if all characters in a string are digits.**
- **digit='777'**
- **print(' is ',digit, ' digit? ' ,digit.isdigit())**
- **digit='777a'**
- **print(' is ',digit, ' digit? ' ,digit.isdigit())**

- **isalpha()- Returns True if all characters in a string are characters from an alphabet.**

Python String Functions

- **isspace()-** Returns True if all characters in a string are spaces.
- >>> a=' '
- >>> a.isspace()

Python String Functions

- **startswith() -** It takes a string as an argument, and returns True is the string it is applied on begins with the string in the argument.
- str = ‘understand’
- >>> str.startswith('un')
- True
- **i. endswith()-** It takes a string as an argument, and returns True if the string it is applied on ends with the string in the argument.
- >>> a='therefore'
- >>> a.endswith('fore')
- True
- **j. find()-** It takes an argument and searches for it in the string on which it is applied. It then returns the index of the substring.
- >>> 'homeowner'.find('meow')
- 2
- If the string doesn’t exist in the main string, then the index it returns is 1.
- >>> 'homeowner'.find('wow')
- -1

Python String Functions

- **replace()-** It takes two arguments. The first is the substring to be replaced. The second is the substring to replace with.
- >>> 'banana'.replace('na','ha')
- ‘bahaha’
- **l. split()-** It takes one argument. The string is then split around every occurrence of the argument in the string.
- >>> 'No. Okay. Why?'.split('.')
- [‘No’, ‘ Okay’, ‘ Why?’]
- **m. join()-** It takes a list as an argument, and joins the elements in the list using the string it is applied on.
- >>> "*".join(['red','green','blue'])
- ‘red*green*blue’