2. Strings

12 October 2023 11:36

STRINGS

- Collection of characters; type of each character is also "str"
  → no such thing as "char" type in Python
- Iterable
- Indexable
- Immutable
- String functions do not modify the original string; they return new strings or other values

NOTE: chr() function
chr (ASCII value)
returns the character corresponding to specified ASCII value

NOTE: docstring
Right after the definition of a function, you can define a multiline string that will be taken as the docstring for that particular function. You can print only the docstring of a function without calling it like so:

```
print (func_name.__doc__)
```

Docstrings are used to store information about functions and especially helps when you're working on code that other people will work on as well.

Eg:

```
def myfunc():
        """This is a docstring.
    I can store info here." ""
    print (myfunc.__doc__)
```

OUTPUT

```
This is a docstring.
I can store info here.
```

Single line / Multiline

- a = "Hello world" → "Hello world"
- a = "Hello \
  world" → "Hello world"
  → used to type string into multiple lines, but actual value is still a single line string
- a = "Hello   world" → "Hello   world"
  → spaces and formatting are kept the same
- a = """Hello
  world""" → "Hello
  world"
  → triple quotes give multiline strings

STRING FUNCTIONS

Use help(str) in interpreter mode to get a list of all string functions

Use help(str) in interpreter mode to get a list of all string functions

① split()

str_name.split(value_to_split_at, maxsplit = n)

If nothing is specified, splits string at every blank space to form a list

If value to split at is specified, splits the string at every occurence of that value.

Specifying maxsplit sets the maximum number of splits to be done

② splitlines()

str_name.splitlines()

Splits multiline strings at line breaks

③ strip(), rstrip(), lstrip()

→ removes white spaces at left end of string

→ removes white spaces at right end of string

→ removes white spaces at both ends of string

④ join()

str_name.join(iterable)

Takes each element of the given iterable and join them with the called string in alternate.

Eg: ".".join(["ab", "bc", "cd"])

OUTPUT

"ab.bc.cd"

⑤ replace()

str_name.replace("old-value", "new_value", count = n)

Returns a copy of the string with all occurences of old_value replaced with new_value if count is not specified.

If specified, count is the no. of occurences from the start of the string that will be replaced

⑥ startswith("value", start, end) / endswith("value", start, end)

Checks if a string starts/ends with specified value.

Optional start and end index values can be given to check a specific range instead of a single value.