PYTHON TUTORIAL FOR BEGINNERS

Source: www.youtube.com/@RishabhMishraOfficial

Chapter - 12

Strings in Python (Part-1)

- Strings and Examples
- Formatted Strings
- Escape Characters
- String Operators

Strings in Python

A string is a sequence of **characters**. In Python, strings are enclosed within single (') or double (") or triple (""") quotation marks.

Examples:

```
print('Hello World!')      # use type() to check data type
print("Won't Give Up!")
print('''"Quotes" and 'single quotes' can be tricky.''')
print("\"Quotes\" and 'single quotes' can be tricky.")
```

Types of Function Arguments

A formatted string in Python is a way to **insert variables** or expressions inside a string. It allows you to format the output in a readable and controlled way.

There are multiple ways to format strings in Python:

1. Old-style formatting (% operator)
2. str.format() method
3. F-strings (formatted string literals)

Formatted String - % Operator

Old-style formatting (% operator)

This approach uses the **%** operator and is similar to string formatting in languages like C.

Syntax: `"string % value"`

Example:

```
name = "Madhav"
age = 16
print("My name is %s and I'm %d." % (name, age))
# %s, %d are placeholders for strings and integers
```

Formatted String - str.format()

str.format() method

In Python 3, the **format**() method is more powerful and flexible than the old-style **%** formatting.

Syntax: `"string {}".format(value)`

Example:

```
name = "Madhav"
age = 16
print("My name is {} and I'm {}.".format(name, age))
# You can also reference the variables by index or keyword:
print("My name is {0} and I'm {1}.".format(name, age))
print("My name is {name} and I'm {age}.".format(name="Madhav",
age=28))
```

Formatted String – F-strings

F-strings (formatted string literals)

In Python 3.6, F-strings are the most concise and efficient way to format strings. You prefix the string with an f or F, and variables or expressions are embedded directly within curly braces **{}**.

Syntax: `f"string {variable}"`

Example:

```
name = "Madhav"
age = 16
print(f"My name is {name} and I'm {age}.")
# You can also perform expressions inside the placeholders:
print(f"In 5 years, I will be {age + 5} years old.")
```

Escape Characters

Escape characters in Python are **special** characters used in strings to represent whitespace, symbols, or control characters that would otherwise be difficult to include. An escape character is a **backslash ** followed by the character you want to insert.

Examples:

```
print('Hello\nWorld!')     # \n for new line
print('Hello\tWorld!')     # \t for tab
print("\"Quotes\" and 'single quotes' can be tricky.") # print single and double quotes
```

String Operators

Operator	Description	Example
+	Concatenation - Adds values on either side of the operator	a + b -- will give HelloPython
*	Repetition - Creates new strings, concatenating multiple copies of the same string	a*2 -- will give HelloHello
[]	Slice - Gives the character from the given index	a[1] -- will give e
[:]	Range Slice - Gives the characters from the given range	a[1:4] -- will give ell
in	Membership - Returns true if a character exists in the given string	H in a -- will give 1
not in	Membership - Returns true if a character does not exist in the given string	M not in a -- will give 1
r/R	Raw String - Suppresses actual meaning of Escape characters.	print(r'\n') -- \n
%	Format - Performs String formatting	% (value)

Python Tutorial Playlist: **Click Here**

https://www.youtube.com/playlist?list=PLdOKnrf8EcP384Ilxra4UlK9BDJGwawg9