The print() function

Till now we have learned that we can use the print() function to print something to the screen. There are many ways to use the print() function which we are going to cover here.

Note: We are using the word *"function"* to describe print(). You may not know what this means in the context of programming. We will introduce the concept of functions later in the course. For now, you can understand a function in a way that it performs certain actions and provides the desired output.

Now, let's explore various ways the print() function can be used.

Basic usage

The print() function prints the specified message to the screen or other standard output device.

Example

```
message = 'Learning Python and chilling'

# print the string message
print(message)

# Output: Learning Python and chilling
```

Using the end parameter

While using the print() function, we can use certain parameters to modify the output. A parameter is a value which can be provided to a function to modify the behaviour of that function. One of the parameters of the print function is **"end"**.

By default, the print function ends with a new line. The example is shown below:

```
print("Hello")
print("World")
```

This will output:

```
Hello
World
```

But, what if we want the output in the same line separated by a space, then we can use the end parameter like this:

```
print("Hello", end = " ")
print("World")
```

The output will be:

```
Hello World
```

We can use any value inside the end operator.

Using the sep parameter

Like the “end” parameter, one other parameter is “sep”. This can be used when multiple outputs are printed in the same print() method. Let’s say we want to print the date and want to separate the day, month and year by “-”, the print function can be used like this:

```
day = '07'
month = '04'
year = '2022'

print(day, month, year, sep='-')
```

Output:

```
07-04-2022
```

Insights

1. To output a string using the print function we use quotes. But in some cases, there can be a conflict when the same quotes are used inside the string. An example is shown below:

   ```
   print('She is Ana's sister')
   ```

   If you write the above script and run, it will throw an error, because we are using the single quotes inside the string also.
   One possible solution for this problem is to use double quotes for the string

   ```
   print("She is Ana's sister")
   ```

   But this can also cause a problem for a more elaborated case like

```
print("She is Ana's sister. Last night she said "Ana helps me with my homework.".")
```

The above script will also produce an error because now we are using both single quote and double quotes inside the string.

Using the backslash (\)

To overcome this, we can use the **“\”** (backslash) operator while using single quotes or double quotes inside the string.

```
print('She is Ana\'s sister')
```

```
print("She is Ana's sister. Last night she said \"Ana helps me with my homework.\".")
```

The backslash helps the print function to understand that the quotes are a part of the string.

2. We can also use **triple quotes (”””)** to print the output. This can also be used to print multiple lines.

```
print("""She is Ana's sister
         She is very intelligent.
           Last night she said to me "Ana helps me in doing my homework".""")
```

We can use double quotes or single quotes inside the triple quotes without using the backslash.