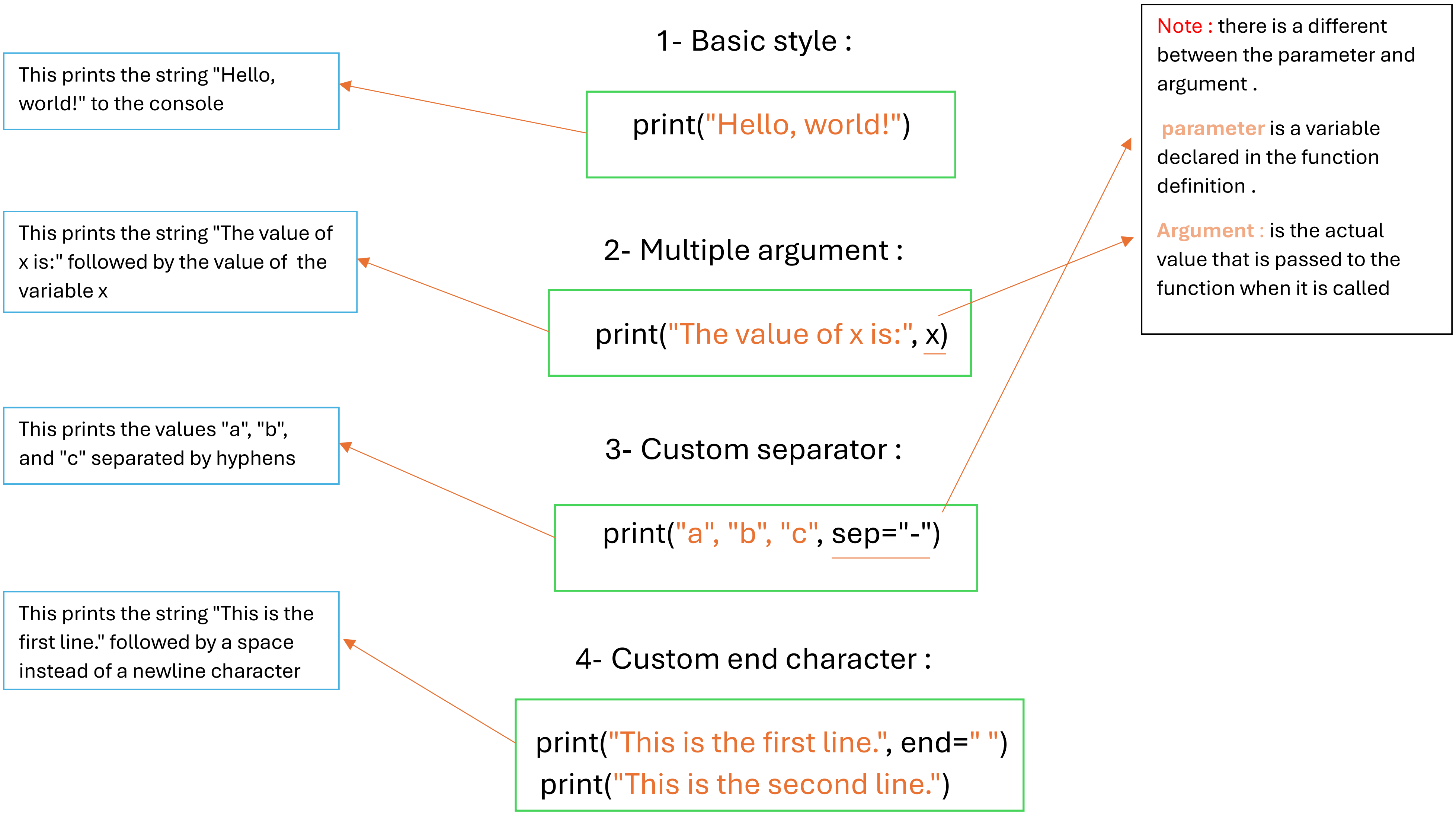
Python (self-learning)

- Printing in python :



1- Basic style :

```
print("Hello, world!")
```

This prints the string "Hello, world!" to the console

2- Multiple argument :

```
print("The value of x is:", x)
```

This prints the string "The value of x is:" followed by the value of the variable x

3- Custom separator :

```
print("a", "b", "c", sep="-")
```

This prints the values "a", "b", and "c" separated by hyphens

4- Custom end character :

```
print("This is the first line.", end=" ")
print("This is the second line.")
```

This prints the string "This is the first line." followed by a space instead of a newline character

Note : there is a different between the parameter and argument .

parameter is a variable declared in the function definition .

Argument : is the actual value that is passed to the function when it is called

5- Old style formatting :

```
name = "Hosam"
age = 30
 print("My name is %s and I am %d years old." % (name, age))
```

The %s and %d placeholders are replaced with the corresponding values from the tuple (name, age). %s is used for strings, and %d is used for integers

6- f-strings (formatted string literals):

```
name = "Hosam"
age = 30
print(f"My name is {name} and I am {age} years old.")
```

more modern and readable way to format strings. Expressions inside curly braces are evaluated and inserted into the string

7- Using the format() Method:

```
print("My name is {} and I am {} years old.".format(name, age))
```

This uses the format() method to format the string. The placeholders {} are replaced with the values from the arguments passed to format()

X

8- Using the join() Method :

```
items = ["apple", "banana", "orange"]
        print(", ".join(items))
```

This joins the elements of the list items into a string, using ", " as the separator