10 Days of Python Tips : Day-3

Inspired by the article by **Fatos Morina**.

Link: *https://towardsdatascience.com/100-helpful-python-tips-you-can-learn-before-finishing-your-morning-coffee-eb9c39e68958*

1. You can use the underscore character anywhere in the name of a variable

This means, anywhere you want, as many times as you want in the name of a variable:

In [2]:
```
a______b = "abcd"  # this works
```

In [3]:
```
_a_b_c_d = "abcd"  # this also works
```

I am not encouraging you to use it, but in case you see a weird naming of a variable like that, know that it is actually a valid name of a variable

2. You can separate big numbers with the underscore

This way it can be easier to read them.

In [6]:
```
print(1_000_000_000)
print(1_234_567)
```

```
1000000000
1234567
```

3. Reverse the ordering of a list

In [7]:
```
my_list = ['a', 'b', 'c', 'd']

my_list.reverse()

print(my_list)  # ['d', 'c', 'b', 'a']
```

```
['d', 'c', 'b', 'a']
```

4. Slice a string using a step function

In [8]:
```
my_string = "This is just a sentence"
print(my_string[0:5])  # This

# Take three steps forward
print(my_string[0:10:3])  # Tsse
```

```
This
Tssu
```

5. Reverse slicing

In [9]:
```
my_string = "This is just a sentence"
print(my_string[10:0:-1])  # suj si sih

# Take two steps forward
print(my_string[10:0:-2])  # sjs i
```

```
suj si sih
sjs i
```

6. Partial Slicing with just the beginning or ending index

Indices indicating the start and end of slicing can be optional.

In [10]:
```
my_string = "This is just a sentence"
print(my_string[4:])  # is just a sentence

print(my_string[:3])  # Thi
```

```
 is just a sentence
Thi
```

7. Floor division

In [11]:
```
print(3/2)  # 1.5
print(3//2)  # 1
```

```
1.5
1
```

8. Difference between == and “is”

“is” checks whether 2 variables are pointing to the same object in memory.

“==” compares the equality of values that these 2 objects hold.

In [12]:
```
first_list = [1, 2, 3]
second_list = [1, 2, 3]

# Is their actual value the same?
print(first_list == second_list)  # True

# Are they pointing to the same object in memory
print(first_list is second_list)
# False, since they have same values, but in different objects in memory


third_list = first_list

print(third_list is first_list)
# True, since both point to the same object in memory
```

```
True
False
True
```

9. Merge 2 dictionaries quickly

In [13]:
```
dictionary_one = {"a": 1, "b": 2}
dictionary_two = {"c": 3, "d": 4}

merged = {**dictionary_one, **dictionary_two}

print(merged)  # {'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

```
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

10. Check whether a string is larger than another string

In [14]:
```
first = "abc"
second = "def"
print(first < second)  # True
second = "ab"
print(first < second)  # False
```

```
True
False
```