Data Science & ML Course

Lesson #3 [Part #2] Python Introduction

Ivanovitch Silva

September, 2018

Agenda

- String operations
- Date operations

global_rankings.csv

Spotify®

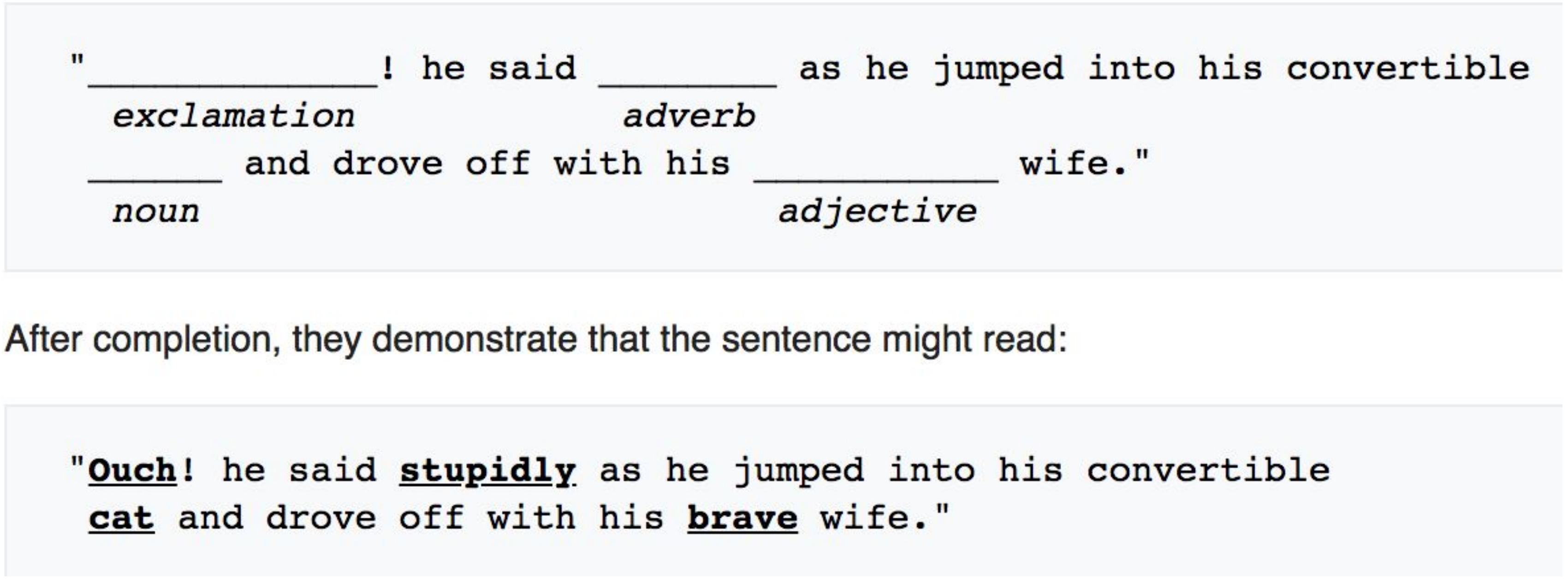
Update from repository

```
git clone https://github.com/ivanovitchm/datascience2machinelearning.git
```

Or ….

```
git pull
```

String operations - Mad Libs



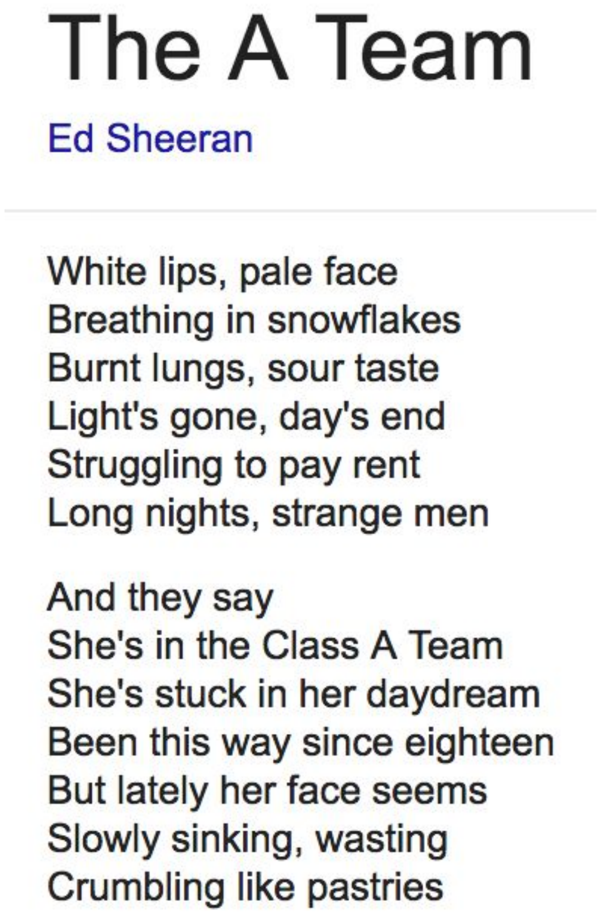
```
"______________! he said ________ as he jumped into his convertible
 exclamation              adverb
______ and drove off with his ___________ wife."
 noun                          adjective
```

After completion, they demonstrate that the sentence might read:

"**Ouch**! he said **stupidly** as he jumped into his convertible
cat and drove off with his **brave** wife."

Ed Sheeran

String operations - Mad Libs



The A Team

Ed Sheeran

White lips, pale face
Breathing in snowflakes
Burnt lungs, sour taste
Light's gone, day's end
Struggling to pay rent
Long nights, strange men

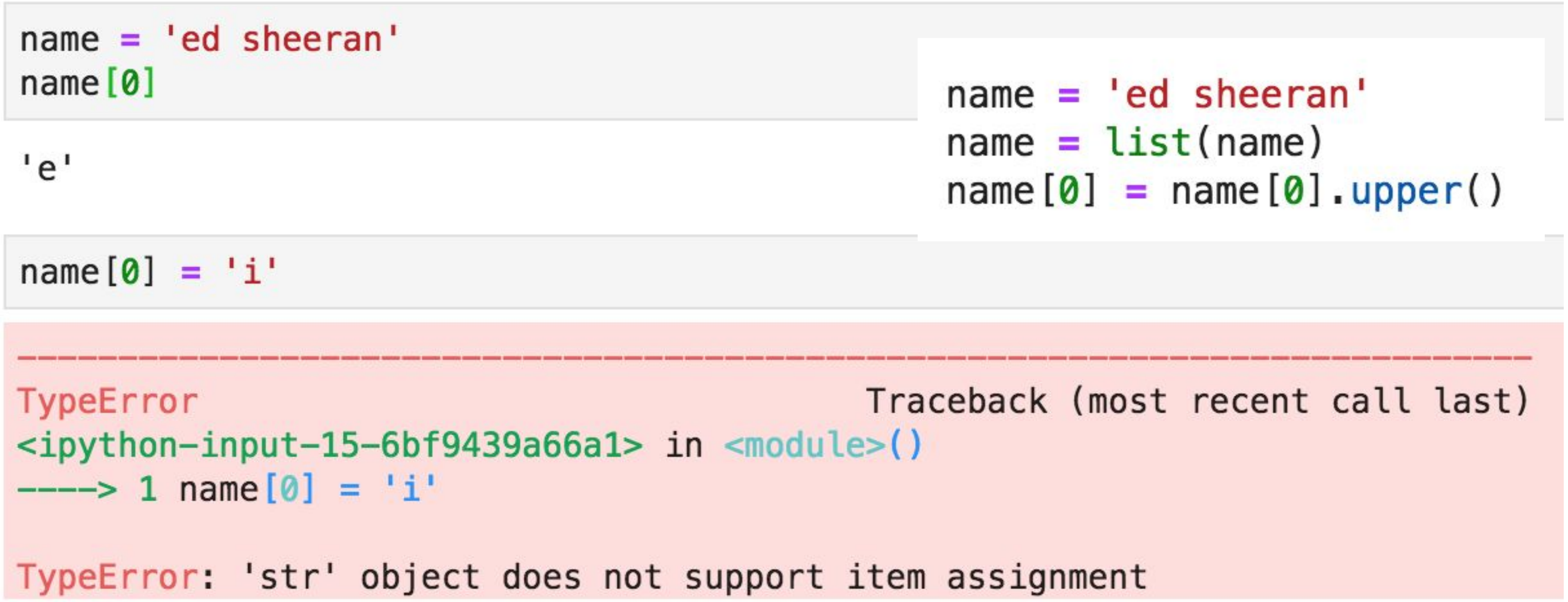
And they say
She's in the Class A Team
She's stuck in her daydream
Been this way since eighteen
But lately her face seems
Slowly sinking, wasting
Crumbling like pastries

In this section, we'll taking Ed Sheeran's lyrics and transforming his lyrics into a mad libs game. We'll write a program that:

- Detects the nouns, verbs and adjectives in his lyrics.
- Replaces these nouns, verbs and adjectives with placeholders.
- Then, we'll replace these placeholders with our own words.

Pseudocode	Drawing
	`'\n'` ↓
1. Insert a new line for each line in the lyrics.	`'white lips, pale face breathing in snowflakes'` ↓
2. Split our string into a list of lists.	`[['white lips, pale face], [breathing in snowflakes']]` ↓
3. To change the first letter to uppercase, lowercase, break the strings into a list of characters.	`[['W','h','i','t','e',' ','l','i','p's'..... ]]` ↓
4. After changing the first letter, turn the list of characters back into a string.	`[['White lips, pale face], [Breathing in snowflakes']]` ↓
5. Replace words with different parts of speech.	`[['ADJ lips, pale face], [VERB in snowflakes']]` ↓
6. Replace the blanks with specified words.	`[['Hard lips, pale face], [Running in snowflakes']]`

Mutable vs Immutable Objects (Python)



```
name = 'ed sheeran'
name[0]
```

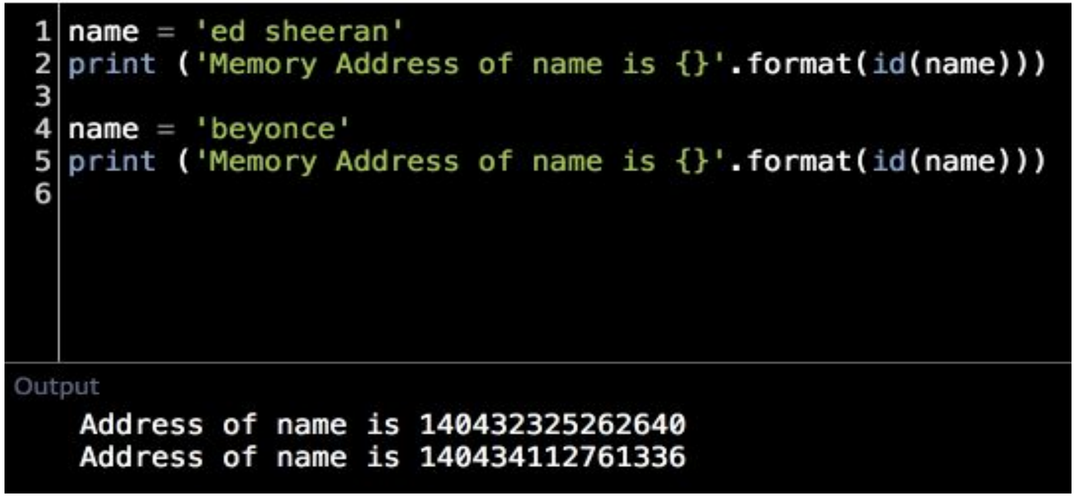
```
'e'
```

```
name[0] = 'i'
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-15-6bf9439a66a1> in <module>()
----> 1 name[0] = 'i'

TypeError: 'str' object does not support item assignment
```

```
name = 'ed sheeran'
name = list(name)
name[0] = name[0].upper()
```

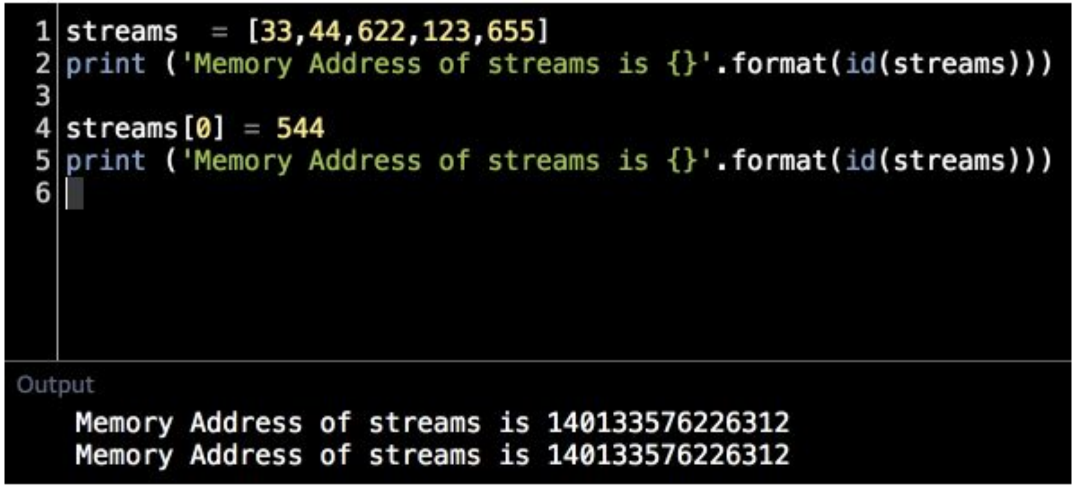
Immutable	Mutable
int	list
float	dict
decimal	set
complex	bytearray
bool	user-defined classes
string	
tuple	
range	
frozenset	
bytes	



```
name = 'ed sheeran'
print ('Memory Address of name is {}'.format(id(name)))

name = 'beyonce'
print ('Memory Address of name is {}'.format(id(name)))

```

Output

```
Address of name is 140432325262640
Address of name is 140434112761336
```

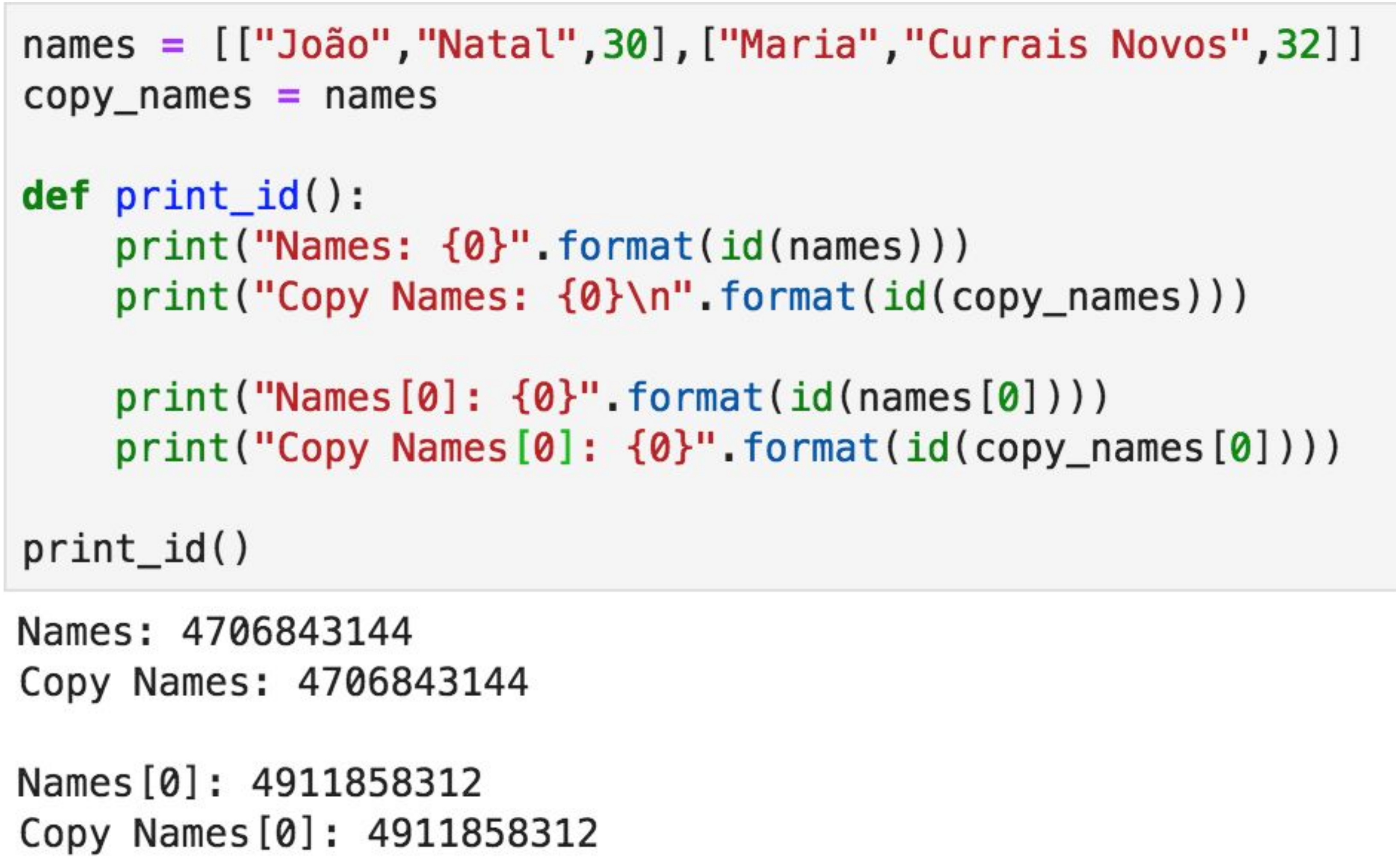


```
streams  = [33,44,622,123,655]
print ('Memory Address of streams is {}'.format(id(streams)))

streams[0] = 544
print ('Memory Address of streams is {}'.format(id(streams)))

```

Output

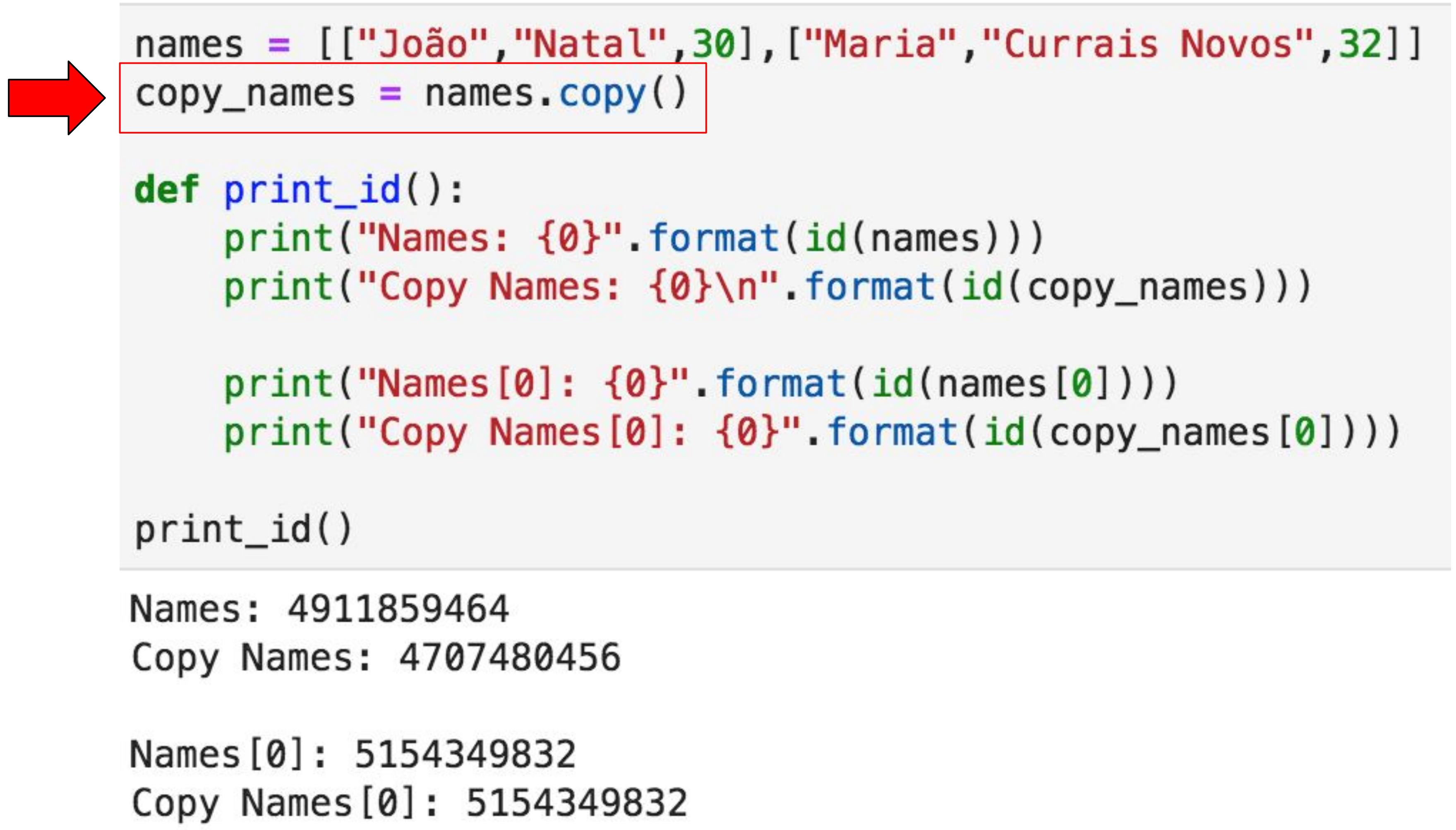
```
Memory Address of streams is 140133576226312
Memory Address of streams is 140133576226312
```

Rááááá! Pegadinha do Malandro!



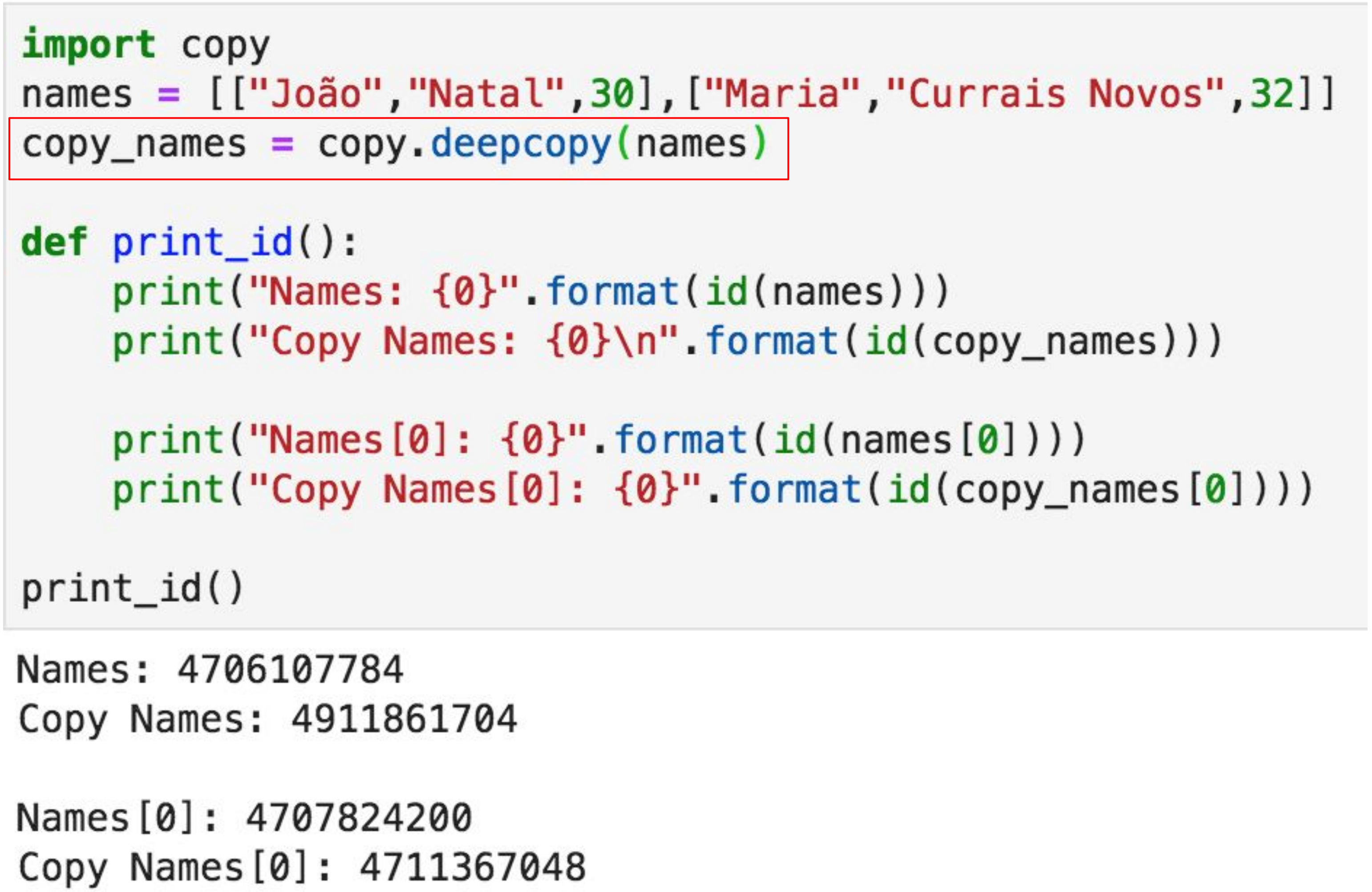
```python
names = [["João","Natal",30],["Maria","Currais Novos",32]]
copy_names = names

def print_id():
    print("Names: {0}".format(id(names)))
    print("Copy Names: {0}\n".format(id(copy_names)))

    print("Names[0]: {0}".format(id(names[0])))
    print("Copy Names[0]: {0}".format(id(copy_names[0])))

print_id()
```

```
Names: 4706843144
Copy Names: 4706843144

Names[0]: 4911858312
Copy Names[0]: 4911858312
```



```
names = [["João","Natal",30],["Maria","Currais Novos",32]]
copy_names = names.copy()

def print_id():
    print("Names: {0}".format(id(names)))
    print("Copy Names: {0}\n".format(id(copy_names)))

    print("Names[0]: {0}".format(id(names[0])))
    print("Copy Names[0]: {0}".format(id(copy_names[0])))

print_id()
```

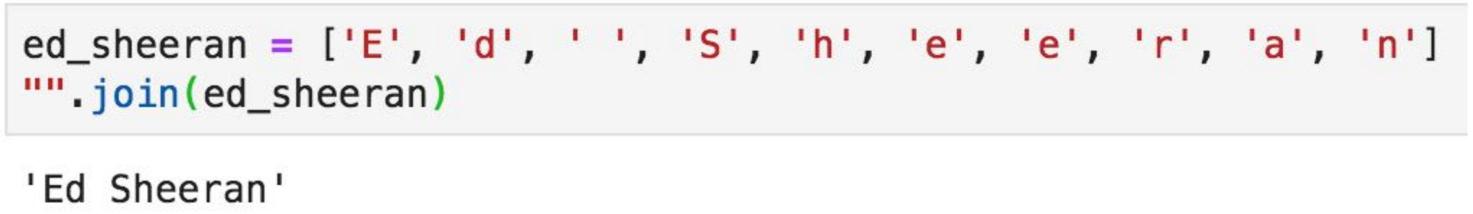
```
Names: 4911859464
Copy Names: 4707480456

Names[0]: 5154349832
Copy Names[0]: 5154349832
```



```python
import copy
names = [["João","Natal",30],["Maria","Currais Novos",32]]
copy_names = copy.deepcopy(names)

def print_id():
    print("Names: {0}".format(id(names)))
    print("Copy Names: {0}\n".format(id(copy_names)))

    print("Names[0]: {0}".format(id(names[0])))
    print("Copy Names[0]: {0}".format(id(copy_names[0])))

print_id()
```

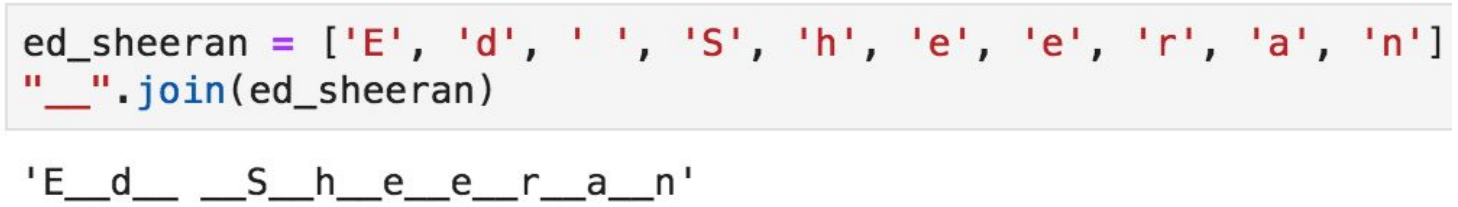
```
Names: 4706107784
Copy Names: 4911861704

Names[0]: 4707824200
Copy Names[0]: 4711367048
```

Joining a list of strings into one string



```
ed_sheeran = ['E', 'd', ' ', 'S', 'h', 'e', 'e', 'r', 'a', 'n']
"".join(ed_sheeran)
```

```
'Ed Sheeran'
```



```
ed_sheeran = ['E', 'd', ' ', 'S', 'h', 'e', 'e', 'r', 'a', 'n']
"__".join(ed_sheeran)
```

```
'E__d__ __S__h__e__e__r__a__n'
```

Replacing values in String

```
name = "ed sheeran"
name.replace("ed","od").replace("sh", "re")
```

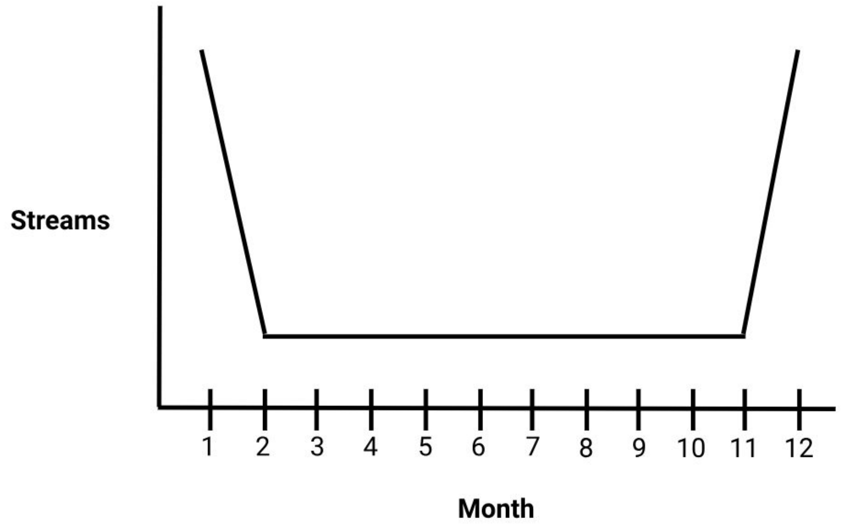
```
'od reeeran'
```

```
"hello {0}. I'm doing {1}".format("world", "well")
```

```
"hello {NOUN}. I'm doing {ADJ}".format(NOUN="world", ADJ="well")
```

Lesson_3_part_2.ipynb
Section 1.1 to 1.10

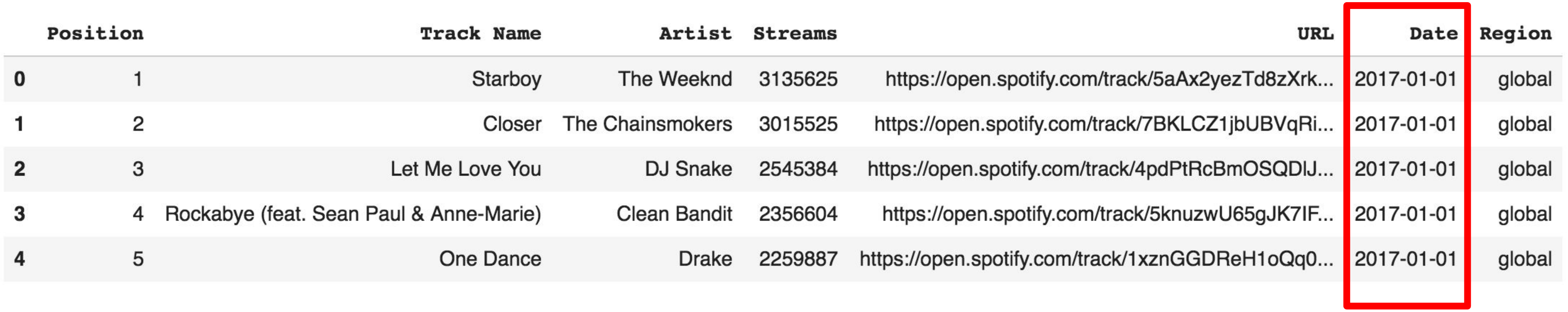
Date operations



Such as a Data Scientist, we should always maintain a healthy degree of skepticism towards our initial results.

Whenever we're performing an analysis, a common influence on our results is time.

Who is the dominant artist for each month of the year?



	Position	Track Name	Artist	Streams	URL	Date	Region
0	1	Starboy	The Weeknd	3135625	https://open.spotify.com/track/5aAx2yezTd8zXrk...	2017-01-01	global
1	2	Closer	The Chainsmokers	3015525	https://open.spotify.com/track/7BKLCZ1jbUBVqRi...	2017-01-01	global
2	3	Let Me Love You	DJ Snake	2545384	https://open.spotify.com/track/4pdPtRcBmOSQDlJ...	2017-01-01	global
3	4	Rockabye (feat. Sean Paul & Anne-Marie)	Clean Bandit	2356604	https://open.spotify.com/track/5knuzwU65gJK7IF...	2017-01-01	global
4	5	One Dance	Drake	2259887	https://open.spotify.com/track/1xznGGDReH1oQq0...	2017-01-01	global

str

datetime

Datetime class

- **time** - Represents time of day. To import:

```
from datetime import time
```

- **date** - Represents a date in an idealized calendar. To import:

```
from datetime import date
```

- **datetime** - Represents month, day, dayofweek, year etc. Combines both **time** class and **date** class. To import:

```
from datetime import datetime
```

- **timedelta** - Represents duration of time, difference between two dates. To import:

```
from datetime import timedelta
```

Creating a datetime based on a string

```
date = "01/01/2017"
datetime.strptime(date, "%m/%d/%Y")
```

```
date = "05-02-2017"
datetime.strptime(date, "%m-%d-%Y")
```

Finding the top artist for each group (m,d,y)

Separate data by month

Track	Artist	Streams	Month
A	Bob	100	1
B	Bill	200	1
C	Bob	300	1
D	Bill	400	2

Within each month, group by the artist

Track	Artist	Streams	Month
A	Bob	100	1
B	Bill	200	1
C	Bob	300	1

When grouping the artists, we'll need to take the sum of the "Streams" column

Track	Artist	Streams	Month
A	Bob	400	1
B	Bill	200	1

Lesson_3_part_2.ipynb
Section 2