C2 - Python for Data Science

C-DAT-100

Python Data Structure

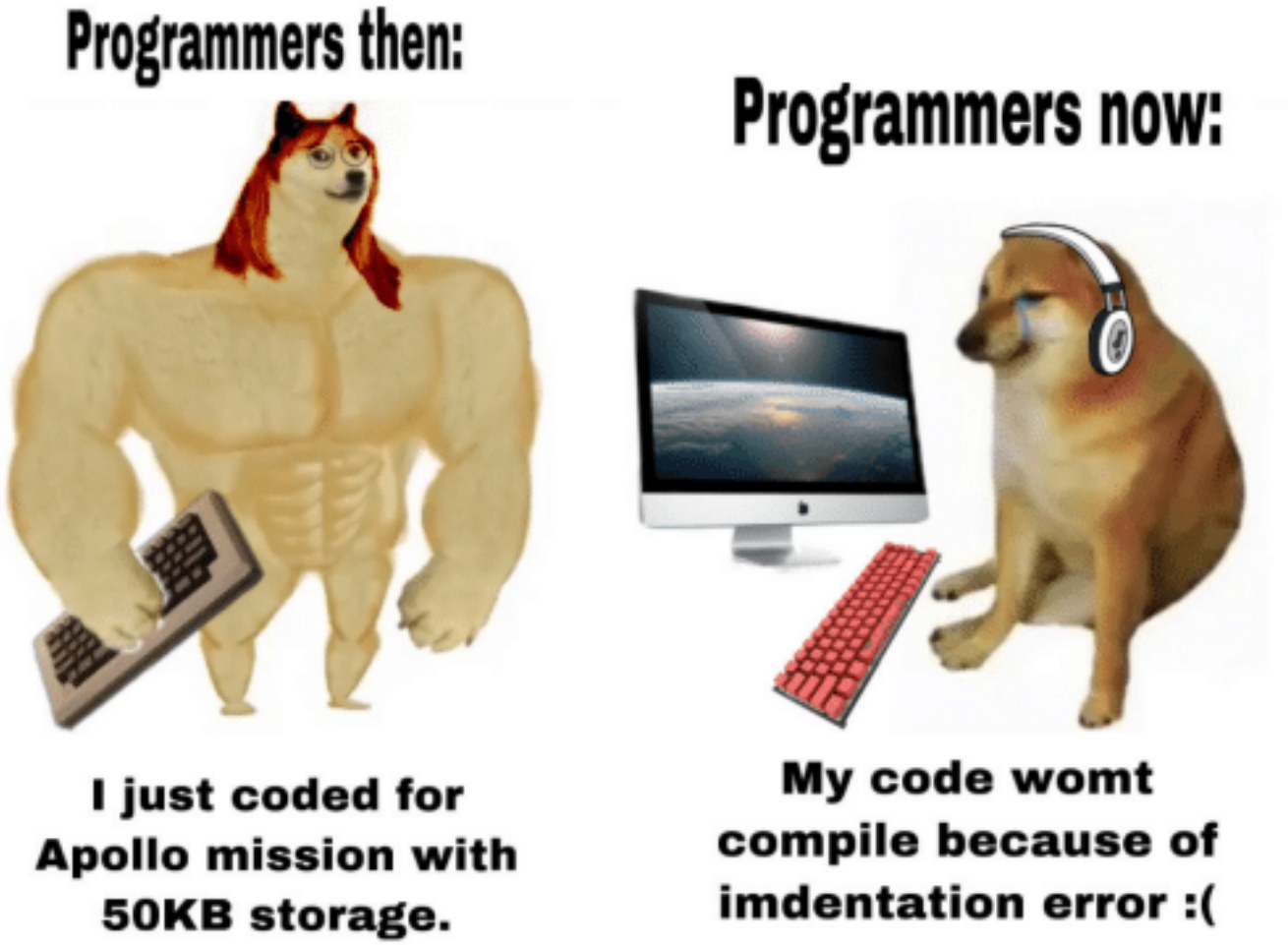
Is this the Monty Python ?

1.5

Python Data Structure

delivery method: py01 on Github
language: python

Before doing Data Science, you must first learn **Python syntaxt**. It's a general purpose and high programming language used by all Data Science / Machine Learning frameworks such as Scikit-learn, PyTorch and Tensorflow. You should coding using at least **PEP8** Python coding convention, ideally **PEP484** too.



EXERCISE 01

Create a function **show_transactions** that takes a list as parameter and displays each of its elements

EXAMPLE

```
transactions = [512, 42.08, -12]
show_transactions(transactions)
```

Terminal

```
~/C-DAT-100> python3 ex_01.py
You received 512 euros
You received 42.08 euros
You spent 12 euros
```

EXERCISE 02

Create a class called **Budget** with **_ transactions** as class attribute.

Adapt your previous function **show_transactions** as a method (class function) using **_ transactions**

Create **add_transaction** that takes only numerical values as parameter and raise an Exception overwise. The method appends the value to **_transactions**

EXAMPLE

```
transactions = [512, 42.08, -12]
wallet = Budget()
wallet.add_transactions(transactions)
wallet.show_transactions()
```

```
Terminal
~/C-DAT-100> python3 ex_02.py
You received 512 euros
You received 42.08 euros
You spent 12 euros
```

EXERCISE 03

Make the class **Budget** takes an optionnal parameter when created. This parameter is a file path (.json) that contains history of all transactions.

The json is structured as such

```
{
   "transactions" : [
         {
            "value" : -42,
            "category" : "transport"
         },
         {
            "value" : 1234,
            "category" : "salary"
         }
    ]
}
```

As you see, there is a new **category** field. **_transactions** must now be of type **Dict** where each **key** is the name of the category and the corresponding **value** is a list containing all transactions for this category

Create a function named **get_category** that returns all categories.

EXAMPLE

```
wallet = Budget("~/data.json")
for category in wallet.get_category() :
    print(category)
    wallet.show_transactions(category)
```

Terminal

```
~/C-DAT-100> python3 ex_03.py
Transport
You spend 42.08 euros
Salary
You received 1234 euros
```

EXERCISE 04

Create a package called **Finance** containing your Budget class. The exemple must be working.

EXAMPLE

```
from Finance import Budget

wallet = Budget()
```

See _ _ init _ _.py

EXERCISE 05

Create a **C.L.I** (Command Line Interface) as a very simple budget app :
- when started, it opens previous transactions records
- the user can add a new transaction, specifying both the category and the value
- the user can see its financial balance
- when closed, it saves data as json into a file

EXAMPLE

```
Terminal
~/C-DAT-100> ./budget.py
Choose between :
1 - consult my balance
2 - add a new transaction
3 - consult your transactions history
4 - quit
```