Thuisproject dec 2020

Het thuisproject is een volledige zelfstandige oefening waarbij je alle aangeleerde skills uit het labo toepast. Het is een verplichte opgave. Er staat geen score op. Jouw oplossing moet op GitHub staan. Voer regelmatig een 'commit & push' uit. Geef telkens een gepaste boodschap mee.

Start

Om je repository in onze Github-classroom aan te maken, klik je op volgende link: https://classroom.github.com/a/GtFwZ67q

Klik op de juiste repository. Clone en open de repository in visual studio.

Werkwijze:

1: Analyseer de json file aandachtig in de map doc. Voor deze oefening gebruik je één bestand (het andere bestand is qua structuur gelijkaardig en kan je even testen op het einde van de oefening). Je zal onmiddellijk opmerken dat het leesteken op de eerste (en laatste) lijn anders is. Deze maal staat er een { }.

Het resultaat van de methode __inlezen_local_json_file zal deze maal een dictionary zijn. Uit de json-file volgt dat

- De value van de key ***competition*** in deze dictionary is opnieuw een nieuwe **dictionary** {}.
- De value van de key ***teams*** is een **list** []. Elk **element** van deze list bestaat op zijn beurt uit een dictionary {}

2: In de submap 'model' maken we volgende twee dataklasses aan:

2a: klasse '**`Team`**' met de attributen *name, shortname, founded, colors* en *venue*.

- Voorzie de klasse van een constructor die alle attributen opvult.
  - De constructor heeft volgende parameters: *name, shortname, founded, colors* en *venue*
- Getter- en setter-properties voor elk attribuut
  - Bij de setter *founded* controleer je natuurlijk of het een integer is en groter dan 0 is. Bij een fout geef je een ValueError met passende boodschap terug.
  - Controleer in de andere setters of het type een string is en niet leeg is. Bij een fout geef je een ValueError met passende boodschap terug.
- tostring-methode
- repr-methode
- eq-methode
  - Een team is gelijk aan een ander team als de *shortnames* gelijk zijn.
- lt-methode (zodat er later kan gesorteerd worden).
  - Alfabetisch op *shortname*.

2b: klasse '**`Competition`**' met de attributen *id*, *name*, *code, area en teams*.

- De klassieke methodes__init__() met parameters *id*, *name*, *code, area*
  - Voor de teams maak je een lege list aan in zijn attribuut. Je voorziet hiervoor enkel een getter-property.
  - Het klaarzetten van de lege list teams gebeurt in de constructor
- Getter- en setter-properties
  - *Name* en *code hebben een setter en getter property*
    - Controleer in elke setter of het type een string is en niet leeg. Bij een fout geef je een ValueError.
  - Id en area hebben enkel een getter property
- Voeg een methode '**voeg_team_toe'** toe: deze methode heeft één parameter, nl een object van de klasse Team. Deze methode voegt het object aan de list teams toe.
  - Controleer in de methode of de parameter weldegelijk een object van de klasse Team is
  - Controleer het team niet reeds eerder aan de list toegevoegd is. (*welke methode uit de klasse Team wordt hiervoor achter de schermen gebruikt?)*
- Programmeer de methode __str__(): deze geeft terug:
  "Competion : *id name code area* aantal teams *xxxx*"

Test nu reeds je klasses uit: Als je **test_voetbal_zonder_json.py** uitvoert moet je volgende output krijgen

```
Competition :1007 Howest competition HWST West-Vlaanderen - aantal teams 2
        Teams in list: [TEAM: MIT, TEAM: MCT]
```

Test nu reeds je klasses uit: Wat gebeurt er indien je verkeerde parameters (lege strings, negatieve jaartallen,…) doorgeeft in de constructor? Krijg je je eigen foutboodschap te zien?

Doe commit en push naar GitHub.

3: Voeg een nieuw bestand CompetitionRepository.py in de map 'model' toe. Maak hierin een nieuwe klasse **CompetitionRepository** aan.

- Voorzie de klasse van een private klasse-attributte "__filename" die het pad naar het bronbestand bijhoudt. Bijvoorbeeld:

```
__filename = "doc/uefa.json"
```

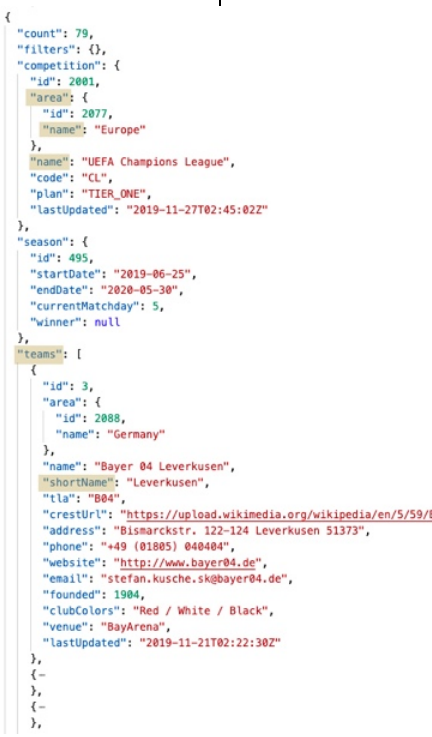
- Voeg een static methode '**load_competition**' toe die in staat is om het bronbestand in te lezen en de competition terug te geven.

  Om een lokaal json-file in te lezen maak je opnieuw gebruik van de private hulpmethode.

  Vervolledig onderstaande code.
  - Schenk extra aandacht hoe je de naam van de area opvraagt in de voorbeeldcode.
  - Waarom staat de *temp_comp_naam* en *temp_comp_area_name* BUITEN de for-lus? Kijk hiervoor even terug naar de json-file.
  - Met welk deel uit de json-file komt de inhoud van de for-lus overeen???

  Opgelet: niet elke vermelde waarde in de json is correct. Vervolledig daarom de methode load_competition met exception handling.

```python
# imports?!?!?
class CompetitionRepository:

    __filename = "doc/uefa.json"


    @staticmethod
    def load_competition():
        dict_json =

    CompetitionRepository._read_local_json_file(CompetitionRepository.__filename)
        dict_comp = dict_json["competition"]
        temp_comp_naam = dict_comp["name"]
        temp_comp_code = dict_comp["code"]
        temp_comp_id = dict_comp["id"]

        # nieuwe dict die in area zit
        dict_comp_area = dict_comp["area"]
        temp_comp_area_name = dict_comp_area["name"]
        geladen_comp = Competition(temp_comp_id, temp_comp_naam,
                       temp_comp_code, temp_comp_area_name)

        # alle teams opvragen
        dict_teams = dict_json["teams"]
        for team in dict_teams:
                temp_short_name = team["shortName"]
                ……
                temp_team = Team(…… ,temp_short_name, …)
                geladen_comp.voeg_team_toe(temp_team)
        return geladen_comp

    @staticmethod
    def __inlezen_local_json_file(bestandsnaam):
        fo = open(bestandsnaam)
        response_json = fo.read()
        fo.close()
        return json.loads(response_json)
```



4: Voeg een static methode '**search_team_by_founded'** toe met twee parameters, nl. een object van de klasse *Competition* en een *jaartal*.

- Vraag binnen deze methode de teams property van de competition op, doorloop deze list.
- Controleer het jaartal.
- Return enkel de teams die in het opgegeven jaartal zijn opgericht.

4: Voeg een static methode '**search_team_by_clubcolor'** toe met twee parameters, nl. een object van de klasse *Competition* en een *kleur* (string).

- Vraag binnen deze methode de teams property van de competition op, doorloop deze list.
- Controleer of de doorgegeven kleur (string) voorkomt in de colors propertie (string).
- Return enkel de teams waarvan de kleur in de string van clubcolors voorkomen.

Doe commit en push naar GitHub

Test uit: Als je test_voetbal_met_json.py uitvoert moet je volgende output krijgen

```
start laden...
Foutmelding: Geen geldige colors!
Foutmelding: Geen geldige colors!
Foutmelding: Geen geldige colors!
...laden is gedaan

print info competition
Competition :2001 UEFA Champions League CL Europe - aantal teams 76

print de ploegen uit de competition
Leverkusen                      Red / White / Black
Dortmund                        Black / Yellow
Bayern M                        Red / White / Blue
Chelsea                         Royal Blue / White
Liverpool                       Red / White
Man City                        Sky Blue / White
Tottenham                       Navy Blue / White
Atleti                          Red / White / Blue
Barça                           Red / Navy Blue / Orange
…(knip)
Partizani                       Red / Yellow
Sl. Bratislava                  Light Blue / White
Feronikeli                      Green / Black / White
Tre Penne                       White / Blue
Saburtalo                       Red / White
HB                              Red / Black

zoek ploegen, founded in 1904
Leverkusen                      Red / White / Black
Atalanta                        Black / Blue
Napoli                          Sky Blue / White
PSG                             Red / Blue / White
SL Benfica                      Red / White
HB                              Red / Black

zoek ploegen, met rood in de clubkleuren
Dortmund                        Black / Yellow
Galatasaray                     Red / Yellow
```

Maribor	Violet / Yellow
BATE	Blue / Yellow
APOEL	Yellow / Blue
Young Boys	Yellow / Black
Sheriff	Yellow / Black
AIK Fotboll	Black / White / Yellow
Maccabi TA	Yellow / Blue
Partizani	Red / Yellow

sorteer de ploegen alfabetisch met **rood** in de clubkleuren

AIK Fotboll	Black / White / Yellow
APOEL	Yellow / Blue
BATE	Blue / Yellow
Dortmund	Black / Yellow
Galatasaray	Red / Yellow
Maccabi TA	Yellow / Blue
Maribor	Violet / Yellow
Partizani	Red / Yellow
Sheriff	Yellow / Black
Young Boys	Yellow / Black

Doe commit en push naar GitHub om in te dienen