

Trabalho Prático

Objetivo:

Avaliar entendimento sobre a sintaxe da linguagem python, utilização de instruções de controle de fluxo, estruturas de dados, declaração de funções, tratamento de exceções e conhecimento sobre módulos e pacotes.

Regras:

- Será necessário construir uma biblioteca que seja capaz de fazer a conversão de arquivos no formato **CSV** ↔ **JSON**. O formato do arquivo CSV possui o seguinte formato:

```
apn,zip_code,property_type,lot,sqft,beds,baths,price
06037-5782020007,91007,residential,20492,2991,4,2.5,2168000
06037-5436013020,90039,residential,6752,1533,3,2,898000
06037-2549013007,91040,residential,38757,3146,4,4,998999
06059-03318106,92831,residential,6375,1364,3,1,549000
06059-93279051,92637,residential,1700,1592,3,2,699999
```

- A primeira linha é composta pelo nome de cada coluna. No exemplo dado acima cada coluna é separada por um delimitador de colunas que é o símbolo “,” (vírgula). No entanto, o conversor de CSV para JSON deve ser capaz de aceitar os seguintes delimitadores:

virgula	ponto e virgula	dois pontos	tabulacao
,	;	:	\t

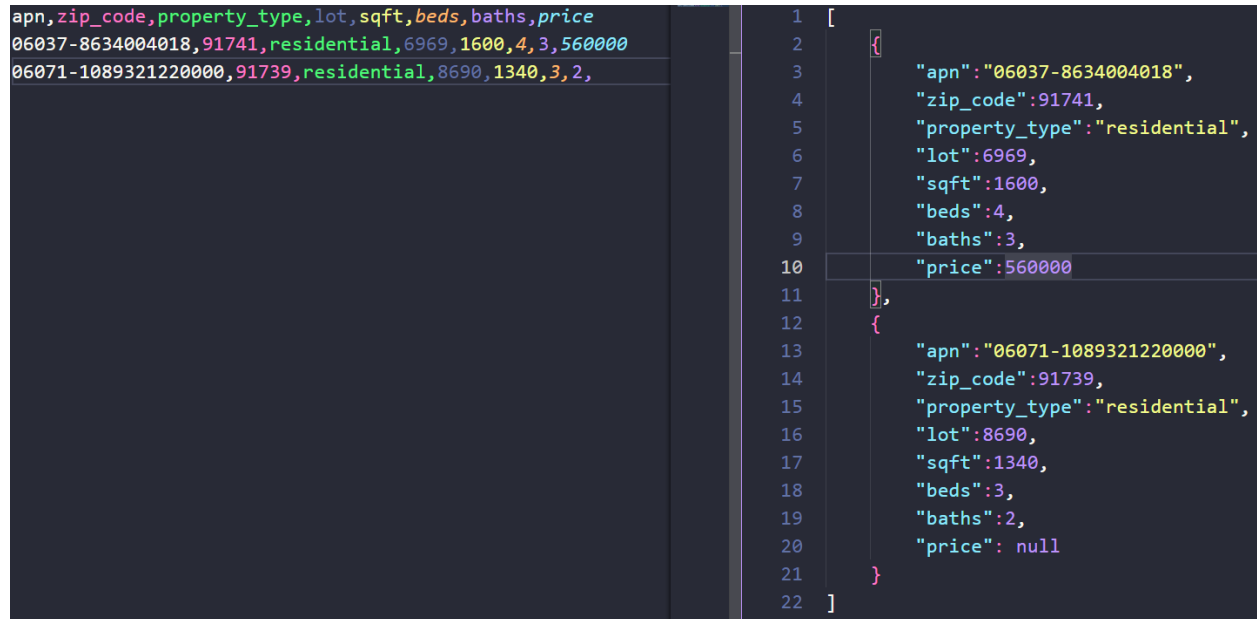
- Lembrem-se, o caracter de tabulação não é exibido diretamente no arquivo, ele é um caracter especial que faz com que haja um espaçamento tabular nos dados. Exemplo:

apn	zip_code	property_type	lot	sqft	beds	baths	price	
06037-5782020007	91007	residential	20492	2991	4	2.5	2168000	
06037-5436013020	90039	residential	6752	1533	3	2	898000	

- Cada coluna possui um tipo especifico, por exemplo, colunas do tipo string devem ser convertidas para string quando mapeadas para JSON. Colunas do tipo inteiro e float

devem ser mapeadas para floats no JSON. Colunas que não tiverem alguma informação, devem ser mapeadas no JSON como **null**.

- O arquivo JSON convertido deve ser uma lista, no qual cada objeto dentro da lista é um dicionário mapeando cada coluna a informação daquela linha. Veja o exemplo na imagem abaixo.



```
apn,zip_code,property_type,lot,sqft,beds,baths,price
06037-8634004018,91741,residential,6969,1600,4,3,560000
06071-1089321220000,91739,residential,8690,1340,3,2,

1 [
2   {
3     "apn": "06037-8634004018",
4     "zip_code": 91741,
5     "property_type": "residential",
6     "lot": 6969,
7     "sqft": 1600,
8     "beds": 4,
9     "baths": 3,
10    "price": 560000
11  },
12  {
13    "apn": "06071-1089321220000",
14    "zip_code": 91739,
15    "property_type": "residential",
16    "lot": 8690,
17    "sqft": 1340,
18    "beds": 3,
19    "baths": 2,
20    "price": null
21  }
22 ]
```

- Conforme pode ser observado, a primeira linha é considerado como o nome das colunas e as duas linhas que se seguem são convertidas para uma lista com 2 dicionários, sendo cada dicionário uma representação direta do nome da coluna e a informação daquela coluna. Conforme pode ser visto na representação JSON, o nome das colunas são mapeadas entre os caracteres aspas duplas ("nome_coluna"). Valores do tipo string são mapeados para strings entre "aspas_duplas", valores inteiros e flutuantes são mapeados para flutuantes e valores faltantes são representados como **null**.
- A conversão de **JSON** para **CSV** deve seguir as mesmas instruções, quando for salvar o arquivo csv deve-se salvar usando um dos 4 possíveis delimitadores de colunas mostrado anteriormente.
- Como o objetivo do projeto é entender conceitos básicos da linguagem, somente será permitido a utilização dos seguintes módulos:
 - [logging](#) para impressão de informações via terminal
 - **Path** da biblioteca [pathlib](#) para criação de endereçamento de arquivos.
 - [typing](#) para informação de tipagem.
 - [click](#) para criação de menus iterativos no terminal
 - [threading](#) ou [concurrent.futures](#) caso queiram fazer uso de paralelismo
- O projeto deve ser upado para o [Pypi](#), para isso vocês devem criar uma conta e fazer o deploy do pacote conforme mostrado em sala de aula.
- O projeto deve ter uma configuração determinada no arquivo **pyproject.toml**, neste arquivo é possível informar configurações do projeto necessárias pelo poetry. Verifique

os possíveis parâmetros da ferramenta poetry deste arquivo nesta [página](#). As configurações mostradas em sala já são suficientes.

- Este arquivo deve conter uma sessão de dependências do projeto e outra sessão de dependências de desenvolvimento.
- Como dependência do projeto só deve constar a versão do python e o módulo click.
- Como dependência de desenvolvimento é sugerido as seguintes bibliotecas (não é obrigatório):
 - [Flake8](#)
 - [mypy](#)
 - [black](#)
 - [isort](#)
 - [ipython](#)
- O projeto deve ser capaz de ser instalado via pip com o comando pip uma vez que esteja publicado no pypi, exemplo de instalação: **pip install nome_do_projeto**
- O projeto deve funcionar via chamada no terminal e ter uma função de ajuda conforme mostrado em sala de aula por meio do comando click. Exemplo:

```
> csv_converter --help
Usage: csv_converter [OPTIONS]

Options:
  -i, --input TEXT      Path where to find files to be loaded for conversion.
  -o, --output TEXT     Path where the converted files will be saved.
  -d, --delimiter TEXT  Separator used to split the files.
  -p, --prefix TEXT     Prefix used to prepend to the name of the converted
                        file saved on disk. The suffix will be a number
                        starting from 0. ge: file_0.json.
  --help               Show this message and exit.
```

- No exemplo mostrado acima temos as opções de utilização de conversão de CSV para JSON, no entanto devemos ter também opções de JSON para CSV. Pensem como inserir estas opções, além de controlar os possíveis caracteres que serão utilizados como delimitadores mencionados anteriormente.
- O projeto deve ser capaz de converter um único arquivo JSON para CSV ou CSV para JSON caso seja passado o path de um arquivo, no entanto caso seja diretório ele deve converter todos os arquivos dentro daquele diretório. Assumam que o diretório contendo os arquivos sejam todos do tipo arquivos JSON ou CSVs.
- Como neste projeto estamos lidando com operações de IO, seria interessante fazermos utilização de paralelismo no projeto. Existem 2 excelentes documentações de utilização nestes links:
 - [Speed up your python program with Concurrency](#)
 - [An intro to concurrency](#)