import pandas as pd

import matplotlib.pyplot as plt

import requests

import json

import json

import requests

# URL da API

url = "https://api-publica.datajud.cnj.jus.br/api\_publica\_tjce/\_search"  # URL do TJSP

api\_key = "APIKey cDZHYzlZa0JadVREZDJCendQbXY6SkJlTzNjLV9TRENyQk1RdnFKZGRQdw=="  # Chave pública

# Array com os códigos de assunto

codigos\_assunto = [293]  # Adicione mais códigos conforme necessário

# Construindo a consulta com uma cláusula 'bool' e 'should' para múltiplos valores

payload = json.dumps({

  "size": 10000,

  "query": {

    "bool": {

      "should": [

        {"match": {"classe.codigo": codigo}} for codigo in codigos\_assunto

      ],

      "minimum\_should\_match": 1  # Pelo menos um deve coincidir

    }

  },

  "sort": [{"dataAjuizamento": {"order": "desc"}}]  # ou asc

})

# Cabeçalhos da solicitação

headers = {

  'Authorization': api\_key,

  'Content-Type': 'application/json'

}

# Enviando a solicitação

response = requests.request("POST", url, headers=headers, data=payload)  # <Response [200]>

dados\_dict = response.json()  # <class 'dict'>

# Imprimindo o número de resultados

print(f"Número de resultados: {len(dados\_dict.get('hits', {}).get('hits', []))}")

import json

import requests

import pandas as pd

# URL da API

url = "https://api-publica.datajud.cnj.jus.br/api\_publica\_tjce/\_search"  # URL do TJSP

api\_key = "APIKey cDZHYzlZa0JadVREZDJCendQbXY6SkJlTzNjLV9TRENyQk1RdnFKZGRQdw=="  # Chave pública

# Array com os códigos de assunto

codigos\_assunto = [293]  # Adicione mais códigos conforme necessário

# Construindo a consulta com uma cláusula 'bool' e 'should' para múltiplos valores

payload = json.dumps({

  "size": 10000,

  "query": {

    "bool": {

      "should": [

        {"match": {"classe.codigo": codigo}} for codigo in codigos\_assunto

      ],

      "minimum\_should\_match": 1  # Pelo menos um deve coincidir

    }

  },

  "sort": [{"dataAjuizamento": {"order": "desc"}}]  # ou asc

})

# Cabeçalhos da solicitação

headers = {

  'Authorization': api\_key,

  'Content-Type': 'application/json'

}

# Enviando a solicitação

response = requests.request("POST", url, headers=headers, data=payload)  # <Response [200]>

dados\_dict = response.json()  # <class 'dict'>

# Criando uma lista para armazenar os dados dos processos

processos = []

# Extraindo os dados necessários

for processo in dados\_dict['hits']['hits']:

    numero\_processo = processo['\_source'].get('numeroProcesso', None)

    grau = processo['\_source'].get('grau', None)

    classe = processo['\_source'].get('classe', {}).get('nome', None)

    assuntos = processo['\_source'].get('assuntos', [])  # Pode ter mais de um

    data\_ajuizamento = processo['\_source'].get('dataAjuizamento', None)

    ultima\_atualizacao = processo['\_source'].get('dataHoraUltimaAtualizacao', None)

    formato = processo['\_source'].get('formato', {}).get('nome', None)

    codigo = processo['\_source'].get('orgaoJulgador', {}).get('codigo', None)

    orgao\_julgador = processo['\_source'].get('orgaoJulgador', {}).get('nome', None)

    municipio = processo['\_source'].get('orgaoJulgador', {}).get('codigoMunicipioIBGE', None)

    sort = processo.get('sort', [None])[0]

    try:

        movimentos = processo['\_source'].get('movimentos', [])

    except KeyError:

        movimentos = []

    # Adicionando os dados extraídos à lista de processos

    processos.append([numero\_processo, classe, data\_ajuizamento, ultima\_atualizacao, formato, \

                      codigo, orgao\_julgador, municipio, grau, assuntos, movimentos, sort])

# Criando um DataFrame a partir da lista de processos

df = pd.DataFrame(processos, columns=['numero\_processo', 'classe', 'data\_ajuizamento', 'ultima\_atualizacao', \

                                      'formato', 'codigo', 'orgao\_julgador', 'municipio', 'grau', 'assuntos', 'movimentos', 'sort'])

# Exibindo uma amostra de 5 registros do DataFrame

df.sample(5)