Análise de dados - API

In [ ]:
```
# !pip install requests
```

http://www.linkedin.com/in/rosileneljusto (http://www.linkedin.com/in/rosileneljusto)

In [ ]:
```
import requests as r
```

In [ ]:
```
# importando dados de uma API
url = 'https://api.covid19api.com/dayone/country/Brazil'
resp = r.get(url)
resp.status_code
row_data= resp.json()

row_data[0]
```

In [ ]:
```
# Filtrando as informações necessárias
final_data = []
for obs in row_data:
    final_data.append([obs['Confirmed'],obs['Deaths'],obs['Recovered'],obs['Active'],obs['Date']])
```

In [ ]:
```
final_data
```

In [ ]:
```
# Colocando dentro de lista
final_data.insert(0, ['Confirmed','Deaths','Recovered','Active','Date'])

final_data
```

In [ ]:
```
#for i in range(1, len(final_data)):
    #final_data[i][Date]=final_data[i][Date][:10]
    # dt.datetime.strptime(final_data[i][Date], '%y-%m-%d')

    #final_data
```

In [ ]:
```
# Importando a biblioteca datetime
import datetime as dt
```

In [ ]:
```
# Formatação de Datetime
print(dt.time(12,6,21),'hora:minuto:segundo:microsegundo')
print('dt.time')
```

In [ ]:
```
print(dt.date (2020,4,25),'Ano_mês_dia')
print('dt.time')
```

In [ ]:
```
print(dt.datetime(2020,4,25), 'Ano_mês_dia')
print('dt.time')
```

In [ ]:
```
# Guardando os dados em arquivo csv
import csv
with open('Brazil_covid.csv','w')as file:
    writer = csv.writer(file)
    writer.writerows(final_data)
```

In [ ]:
```
# Transformando em uma Data
#for i in range(1,len(final_data)):
    #final_data[i][Date]=dt.datetime.strptime(final_data[i][Date],'%y-%m-%d')
```

Usando API QuickChart (buscar conhecer a documentação da API)

In [ ]:
```
# criando uma função

def get_datasets(y,labels):
    if type(y[0])==list:
        datasets=[]
        for i in range (len(y)):
            datasets.append({
                'label':labels[i],
                'data':y[i]
            })
            return datasets
        else:
            return[
                {
                    'label': labels[0],
                    'data':y
                }
            ]
```

In [ ]:
```
def set_title(title = ' '):
    if title != ' ':
        display = 'true'

    else:
        display = 'false'

    return {
    'title': title,
    'display': display
}
```

In [ ]:
```
def create_chart(x,y,labels,kind = 'bar', title = ' '):
    datasets =get_datasets(y,labels)
    options = set_title(title)
    chart = {
        'type': kind,
        'data': {
            'labels':x,
            'datasets': datasets

        },
        'options': options # responsavel pelo setup, também pode ser usada para identificação de eixos.
    }
    return chart
```

In [ ]:
```
# Aqui vai retornar a imagem em si. guardar ela junto com o projeto vai retornar a partir dessa função o 'contente'
# dessa requisição que é um valor binário que é como vai conseguir armazenar essa imagem dentro da máquina.
def get_API_chart (chart):
    url_base = 'https://quickchart.io/chart'
    resp = r.get(f'{url_base}?c = {str (chart)}')
    return resp.content
```

In [ ]:
```
# Salvar imagem
def save_image(path,content):
    with open (path, 'wb')as image: # wb de binario
        image.write(content)
```

In [ ]:
```
# Exibindo a imagem dentro do notebook
# !pip install Ipython
from PIL import Image
#from Ipython.display
import IPython.display as display

def display_image(path):
    img_pil = image.open(path)

    display(img_pil)
```

Criando Dados e Gerar os Gráficos

In [ ]:
```
# Criando dados e gerar os gráficos
#y_data1 = []

#for obs in final_data[1::10]:    # pular de 10 em 10
   # y_data1.append([obs[Confirmed]])
```

In [ ]:
```
# Criando dados e gerar os gráficos
#y_data2 = []
#for obs in final_data[1::10]:  # pular de 10 em 10
    #y_data2.append(obs[Recuperados])
```

In [ ]:
```
# Declarando os labels
labels = ['Confirmados', 'Recuperados']
```

In [ ]:
```
# Criando dados e gerar os gráficos
x = []
for obs in final_data[1:10]:  # pular de 10 em 10
    x.append(obs[Date].strftime('%d/%m/%y'))
```

In [ ]:
```
# Gráfico
chart = create_chart(x,[y_data1, y_data2], labels,
                     title = 'Gráficos Confirmados vs Recuperados')
chart_content=get_api_chart(chart)
save image('meu primeiro gráfico.png', chart_content)

display_image('Meu primeiro gráfico.png')
```

Gerar QR CODE com link direto p / o primeiro gráfico. Para compartilhamento

In [ ]:
```
from urllib.parse import quote
```

In [ ]:
```
# Criar uma função
def get_api_qrcode(link):
    text = quote(link)  #parsing do link p/url
    url_base = 'https://quickchart.io/chart'
    resp = r.get(f'{url_base}?text = {text}')
    return resp.content
```

In [ ]:
```
# Recuperar o link
url_base = 'https://quickchart.io/chart'
link = f'{url_base}?c={str(chart)}'
save_image('qr_code.png',get_api_qrcode(link))display_image('qr_code.png')
```

In [ ]:
```
# Primeiro abre o qrcode no app de fotos, dá um zoom na imagem, depois a ponta a câmera do celuar
```