Aluno: Felipe de Jesus Araujo da Conceição - DRE: 119180575

```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[ ] import seaborn as sns
    import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
```

Configuração da máquina (MacBook Pro M3 2023)

- Quantidade total de núcleos: 12
- Quantidade total de threads: 12

Informações sobre os dados na tabela

- Quantidade de threads (Em que 0 signfica sequencial)
- Dimensões das matrizes
- Tempo de inicialização
- Tempo de processamento (apenas multiplicação de matriz)
- Tempo de finalização

```
[ ] # Carrega o arquivo com tempos de execução
    nomes_das_colunas = ['Quantidade de threads', 'Dimensões das matrizes', 'Tempo de inicialização', "Tempo de Processamento", "Tempo de finalização"]
    df = pd.read_csv("/content/drive/MyDrive/ProgConc/temposExecucao.csv", header=None, names=nomes_das_colunas)
    df
```

	Quantidade de threads	Dimensões das matrizes	Tempo de inicialização	Tempo de Processamento	Tempo de finalização
0	0	500	0.001186	0.310951	0.002246
1	0	500	0.000241	0.324474	0.001136
2	0	500	0.000231	0.306925	0.001479
3	0	500	0.000253	0.332067	0.002086
4	0	500	0.000259	0.312965	0.000297
...	...	...	...	...	...
70	8	2000	0.007986	4.581733	0.004126
71	8	2000	0.005212	4.395898	0.003087
72	8	2000	0.003573	4.331167	0.003145
73	8	2000	0.005953	4.522541	0.002900
74	8	2000	0.003831	4.437305	0.003130

75 rows × 5 columns

Médias dos tempos de execução(em segundos) por threads e as dimensões

OBS.: Quantidade de threads 0 faz referencia ao sequencial

```
df_medias = df.groupby(["Quantidade de threads", "Dimensões das matrizes"]).median()
df_medias
```

		Tempo de inicialização	Tempo de Processamento	Tempo de finalização
Quantidade de threads	Dimensões das matrizes			
0	500	0.000253	0.312965	0.001479
	1000	0.000951	2.768062	0.001943
	2000	0.021734	24.796135	0.005035
1	500	0.000244	0.352430	0.000608
	1000	0.000983	2.843199	0.001008
	2000	0.022021	28.825046	0.005659
2	500	0.000259	0.181145	0.000592
	1000	0.001092	1.485681	0.001061
	2000	0.010535	13.670244	0.004734
4	500	0.000250	0.093091	0.001388
	1000	0.000941	0.741677	0.001163
	2000	0.006034	6.989204	0.003694
8	500	0.000268	0.062336	0.000795
	1000	0.001163	0.513900	0.001200
	2000	0.005212	4.437305	0.003130

Graficos

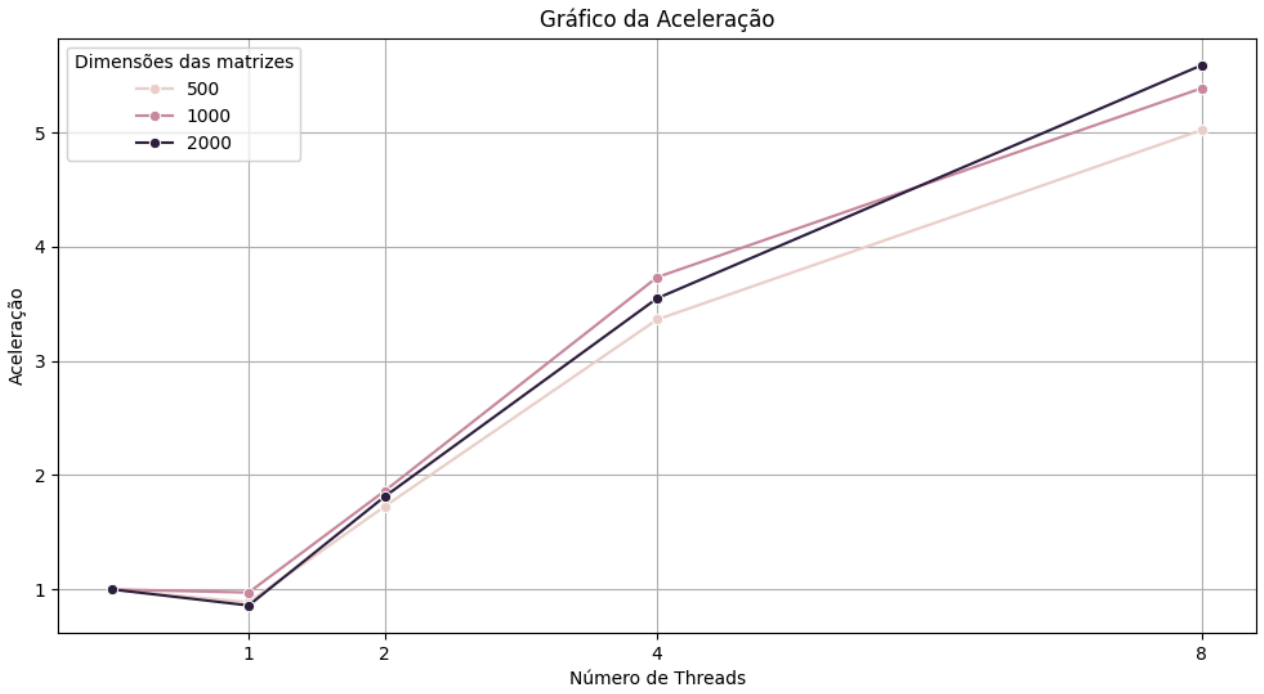
Aceleração

```
[12] # Função para calcular a aceleração
     def calcula_aceleracao(row):
         num_dimensoes = row.name[1]
         return df_medias.loc[0, num_dimensoes]["Tempo de Processamento"] / row["Tempo de Processamento"]
```

```
[13] # Aplicar a função de aceleração para todas as combinações de Threads e Dimensões (exceto o sequencial)
     df_medias['Aceleracao'] = df_medias.apply(calcula_aceleracao, axis=1)
```

```
[14] # Criar o gráfico de linhas
     plt.figure(figsize=(12, 6))
     sns.lineplot(data=df_medias, x='Quantidade de threads', y='Aceleracao', hue='Dimensões das matrizes', marker='o')

     # Configurações do gráfico
     plt.title('Gráfico da Aceleração')
     plt.xlabel('Número de Threads')
     plt.ylabel('Aceleração')
     plt.grid(True)
     plt.xticks([1, 2, 4, 8])
     plt.show()
```



Eficiência

```
[15] # Função para calcular a eficiência
     def calcula_eficiencia(row):
         num_threads = row.name[0]
         if num_threads == 0:
             return 1
         return row["Aceleracao"]/num_threads
```

```
[16] # Aplicar a função de eficiência
     df_medias['Eficiencia'] = df_medias.apply(calcula_eficiencia, axis=1)
```

```
[17] # Criar o gráfico de linhas
     plt.figure(figsize=(12, 6))
     sns.lineplot(data=df_medias, x='Quantidade de threads', y='Eficiencia', hue='Dimensões das matrizes', marker='o')

     # Configurações do gráfico
     plt.title('Gráfico da Eficiência')
     plt.xlabel('Número de Threads')
     plt.ylabel('Eficiência')
     plt.grid(True)
     plt.xticks([1, 2, 4, 8])
     plt.show()
```