SpeedUp

Considere o algoritmo fornecido para determinar se um número é primo ou não. Preencha a tabela abaixo

Número avaliado	É primo?	Tempo médio de Execução do Algoritmo (milissegundos)	Desvio padrão
7	1	0.086000	0.025228
37	1	0.084000	0.025154
8431	1	0.155000	0.023968
13033	1	0.189000	0.021223
524287	1	0.108000	0.133746
664283	0	0.153000	0.036663
3531271	1	13.266000	0.942500
2147483647	1	8524.990000	1010.589294

Desenvolva pelo menos outros dois algoritmos para definir se um número é primo ou não e compare o tempo de execução destes algoritmos com o algoritmo fornecido. (Calcule o SpeedUp de cada um deles em relação ao algoritmo de referência)

Número avaliado	Tempo médio de Execução do Algoritmo (milissegundos)		SpeedUp (Em relação ao algoritmo fornecido)	
	Algoritmo 2	Algoritmo 3	Algoritmo 2	Algoritmo 3
7	0.125000	0.138000	0,105500	0,112000
27	0.190000	0.104000	0,137000	0,094000
8431	0.096000	0.219000	0,125500	0,187000
13033	0.117000	0.253000	0,153000	0,221000
524287	0.097000	4.656000	0,102500	2,382000
664283	0.111000	7.189000	0,132000	3,671000
2147483647	9045.819000	?????	8785,404500	?????

Note que o tempo de execução deles varia a cada execução (por diversos fatores relacionados ao ambiente em que estão sendo executados!), por isso, para determinar o tempo médio, considere pelo menos 30 execuções de cada um deles.

No relatório a ser entregue, além de preencher as tabelas acima:

1. **detalhe os algoritmos propostos**

   O programa tem como objetivo verificar o tempo de processamento do computador para calcular se o número é primo ou não retornando 0 para não primo e 1 para primo, e retornando o valor do tempo em milissegundos e a media entre 30 resultados.

2. **detalhe a forma utilizada para capturar os tempos de execução**

   time.h é um arquivo cabeçalho que fornece protótipos para funções para manipulação de datas e horários de modo padrão.
   A função abaixo pega o tempo do valor inicial do clock() e dentro do print subtrai com o clock, assim gerando o tempo de execução do programa (Abaixo está um exemplo dos comandos).

```
clock_t tempo;
tempo = clock();
printf("%f\n", (clock() - tempo)*1000 / (double)CLOCKS_PER_SEC);
```

3. **disponibilize o código utilizado no github (ou no gitlab) e informe o link no seu relatório.**

   MatheusXDs/CC7261---SISTEMAS-DISTRIBUIDOS (github.com)

4. **Responda como a performance de cada um deles é afetada pela ordem de grandeza do número avaliado.**

   Todos os códigos utilizados eles têm muitos problemas com repetição por causa do for isso não só impacta no tempo de execução do problema é necessário máquinas super potentes para fazer o uma conta simples ser calculada, o tempo do código sobe exponencialmente.