



Desenvolvimento Python para Redes e Sistemas Operacionais

Assessment

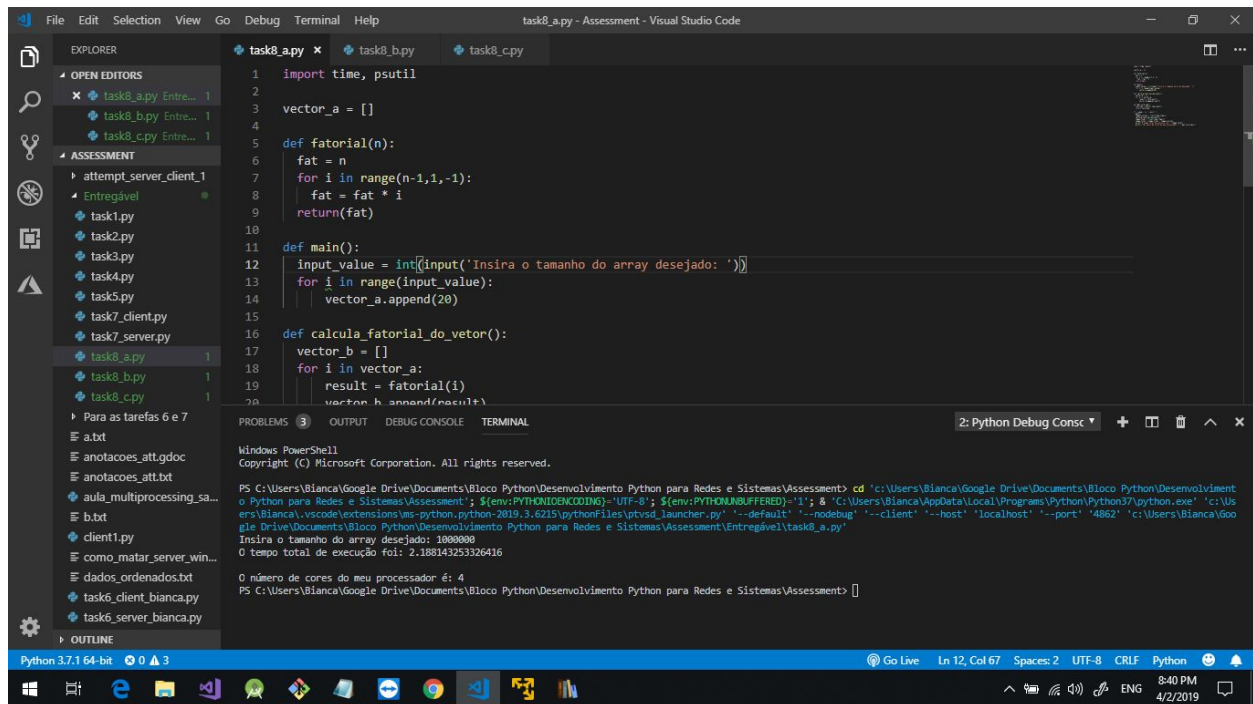
Bianca Gotaski de Melo

16159777750

Comparação entre as execuções da tarefa 8 (a, b e c):

Obs.: Todas as imagens estão em melhor resolução no arquivo em anexo.

Questão 8 a:



The screenshot shows the Visual Studio Code interface with a Python file named `task8_a.py` open. The code defines a `fatorial` function, a `main` function that takes user input for the array size and appends values to a vector, and a `calcula_fatorial_do_vetor` function that iterates through the vector and calculates the factorial of each element. The terminal output shows the execution of the script, displaying the input size (1000000), the total execution time (2.188143253326416 seconds), and the number of processor cores (4).

```
1 import time, psutil
2
3 vector_a = []
4
5 def fatorial(n):
6     fat = n
7     for i in range(n-1,1,-1):
8         fat = fat * i
9     return(fat)
10
11 def main():
12     input_value = int(input('Insira o tamanho do array desejado: '))
13     for i in range(input_value):
14         vector_a.append(20)
15
16 def calcula_fatorial_do_vetor():
17     vector_b = []
18     for i in vector_a:
19         result = fatorial(i)
20         vector_b.append(result)
```

Terminal Output:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment> cd 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment'; $(env:PYTHONIDENCODING='UTF-8'; $(env:PYTHONUNBUFFERED='1'; & 'C:\Users\Bianca\AppData\Local\Programs\Python\Python37\python.exe' 'c:\Users\Bianca\.vscode\extensions\ms-python.python-2019.3.6215\pythonFiles\ptvsd_launcher.py' '--default' '--nodebug' '--client' '--host' 'localhost' '--port' '4862' 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment\Entregavel\task8_a.py')
Insira o tamanho do array desejado: 1000000
O tempo total de execução foi: 2.188143253326416

O número de cores do meu processador é: 4
PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment>
```

Questão 8 b:

The screenshot shows the Visual Studio Code interface with the file `task8_b.py` open. The code defines a function `get_core_cpu()` and a `main()` function. The `main()` function calculates the number of CPU cores and uses `threading.Thread` to calculate factorials for each element in a vector. The terminal output shows the execution of the script, indicating that the number of cores is 4 and the total execution time is 2.3541781902313232 seconds.

```
def get_core_cpu():
    core_cpu = psutil.cpu_count()
    return core_cpu

if __name__ == "__main__":
    main()

    qtd_threads = 4
    tamanho = len(vector_a)
    threads = []
    resultado = []
    inicio_tempo = float(time.time())
    for i in range(qtd_threads):
        inicio = i*int(tamanho//qtd_threads)
        fim = (i+1)*int(tamanho//qtd_threads)
        #passar a virgula junto com calcula_fatorial_do_vetor: calcula_fatorial_do_vetor, hecer que é uma tupla
        t = threading.Thread(target=calcula_fatorial_do_vetor, args=(vector_a[inicio:fim], ))
        threads.append(t)
        t.start()
        resultado.append(t)
    for t in resultado:
```

```
PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment> cd 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment'; $(env:PYTHONIDENCODING='UTF-8'; $(env:PYTHONUNBUFFERED='1'; & 'c:\Users\Bianca\AppData\Local\Programs\Python\Python37\python.exe' 'c:\Users\Bianca\vscode\extensions\ms-python.python-2019.3.6215\pythonFiles\ptvsd_launcher.py' '--default' '--nodebug' '--client' '--host' 'localhost' '--port' '4886' 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment\Entregavel\task8_b.py')
Insira o tamanho do array desejado: 1000000
O tempo total de execução foi: 2.3541781902313232

O número de cores do meu processador é: 4
PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment>
```

Questão 8 c:

The screenshot shows the Visual Studio Code interface with the file `task8_c.py` open. The code defines a function `get_core_cpu()` and a `main()` function. The `main()` function calculates the number of CPU cores and uses `multiprocessing.Process` to calculate factorials for each element in a vector. The terminal output shows the execution of the script, indicating that the number of cores is 4 and the total execution time is 0.32796478271484375 seconds.

```
tempo_inicial = float(time.time())
for i in range(n_processos):
    inicio = i*int(tamanho//n_processos)
    fim = (i+1)*int(tamanho//n_processos)
    fila_entrada.put(vector_a[inicio:fim])
    p = multiprocessing.Process(target = calcula_fatorial_do_vetor, args = (fila_entrada, fila_saida))
    p.start()
    processos.append(p)
for t in processos:
    t.join()

lista_final = []
for p in processos:
    lista_final += fila_saida.get()
# print(lista_final)
tempo_final = float(time.time())
tempo_total = tempo_final - tempo_inicial
print('O tempo total de execução foi:', tempo_total)
print('\nO número de cores do meu processador é:', get_core_cpu())
```

```
PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment> cd 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment'; $(env:PYTHONIDENCODING='UTF-8'; $(env:PYTHONUNBUFFERED='1'; & 'c:\Users\Bianca\AppData\Local\Programs\Python\Python37\python.exe' 'c:\Users\Bianca\vscode\extensions\ms-python.python-2019.3.6215\pythonFiles\ptvsd_launcher.py' '--default' '--nodebug' '--client' '--host' 'localhost' '--port' '5030' 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment\Entregavel\task8_c.py')
Insira o tamanho do array desejado: 10
O tempo total de execução foi: 0.32796478271484375

O número de cores do meu processador é: 4
PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment>
```

Usando um array de tamanho 1.000.000:

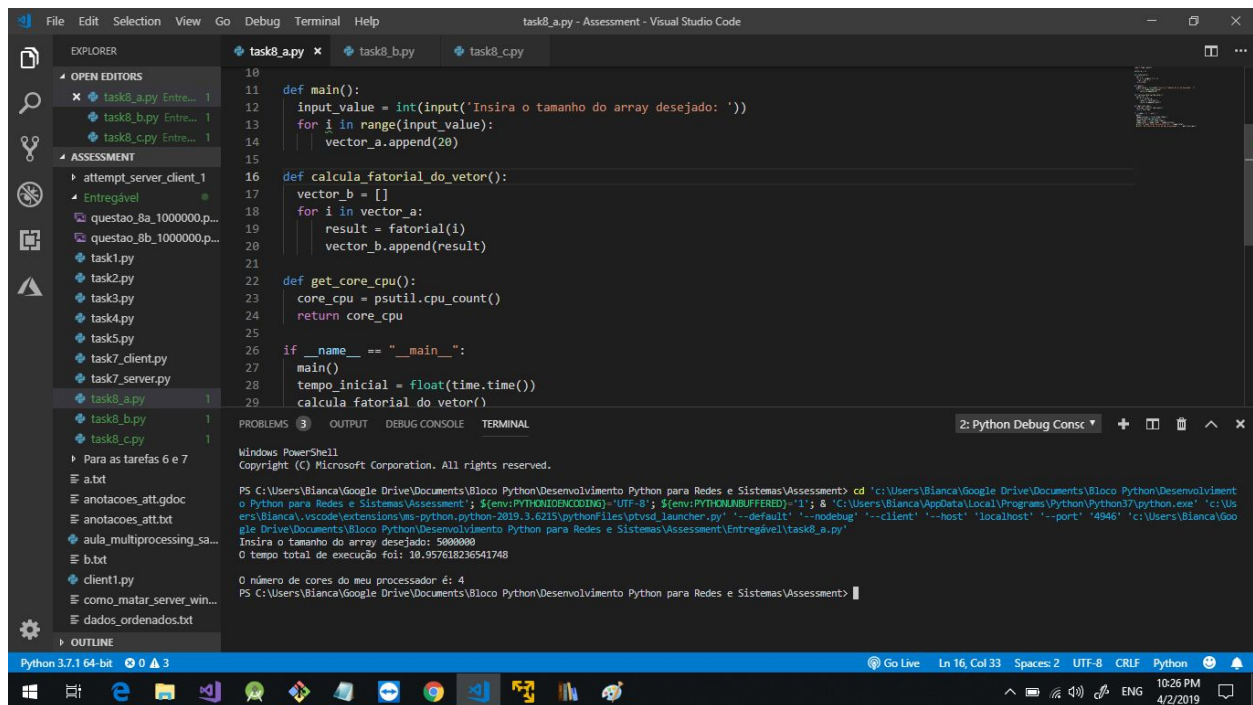
O tempo de execução na 8a foi de 2.18 segundos, enquanto na questão 8b foi de 2.35 segundos.

Logo, a diferença entre 8a e a 8b foi muito pouca.

Já a 8c eu não consegui processar no meu computador, devido ao tamanho do array.

Isso aconteceu porque meu computador não tem capacidade de processamento tão alto para executar em uma quantidade tão pequena de processos, todo o tamanho do array passado. Então, para demonstrar de que a questão funcionou normalmente no meu computador, usei um array de tamanho 10.

Questão 8a para um array de tamanho 5.000.000:



The screenshot shows the Visual Studio Code interface with a Python file named `task8_a.py` open. The code defines a `main` function that takes an input value, creates a vector of that size, and appends the value 20 to it. It also defines a `calcula_fatorial_do_vetor` function that calculates the factorial of each element in the vector. The `main` function calls `calcula_fatorial_do_vetor` and prints the result. The terminal output shows the execution of the script, which takes 10.957618236541748 seconds to complete. The output also shows the number of CPU cores (4) and the total execution time (10.957618236541748 seconds).

```
task8_a.py - Assessment - Visual Studio Code
10
11 def main():
12     input_value = int(input('Insira o tamanho do array desejado: '))
13     for i in range(input_value):
14         vector_a.append(20)
15
16 def calcula_fatorial_do_vetor():
17     vector_b = []
18     for i in vector_a:
19         result = fatorial(i)
20         vector_b.append(result)
21
22 def get_core_cpu():
23     core_cpu = psutil.cpu_count()
24     return core_cpu
25
26 if __name__ == "__main__":
27     main()
28     tempo_inicial = float(time.time())
29     calcula_fatorial_do_vetor()
30
31 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
32 2: Python Debug Consc
33
34 Windows PowerShell
35 Copyright (C) Microsoft Corporation. All rights reserved.
36
37 PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment> cd 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment'; $env:PYTHONPATH='c:\Users\Bianca\AppData\Local\Programs\Python\Python37\python.exe'; & 'c:\Users\Bianca\AppData\Local\Programs\Python\Python37\python.exe' 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment\Entregavel\task8_a.py'
38 Insira o tamanho do array desejado: 5000000
39 O tempo total de execução foi: 10.957618236541748
40
41 O número de cores do meu processador é: 4
42 PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment>
```

Questão 8b para um array de tamanho 5.000.000:

```
task8_b.py
24 return core_cpu
25
26 if __name__ == "__main__":
27     main()
28     qtd_threads = 4
29     tamanho = len(vector_a)
30     threads = []
31     resultado = []
32     inicio_tempo = float(time.time())
33     for i in range(qtd_threads):
34         inicio = i*int(tamanho//qtd_threads)
35         fim = (i+1)*int(tamanho//qtd_threads)
36         ##passar a virgula junto com o primeiro argumento do parametro para o python reconhecer que é uma tupla
37         t = threading.Thread(target=calcula_fatorial_do_vetor, args=(vector_a[inicio:fim], ))
38         threads.append(t)
39         t.start()
40         resultado.append(t)
41     for t in resultado:
42         t.join()
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment> cd 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment'; \$(env:PYTHONIDENCODING="UTF-8"; \$(env:PYTHONUNBUFFERED="1"; & 'c:\Users\Bianca\AppData\Local\Programs\Python\Python37\python.exe' 'c:\Users\Bianca\vscode\extensions\ms-python.python-2019.3.6215\pythonFiles\ptvsd_launcher.py' '--default' '--nodebug' '--client' '--host' 'localhost' '--port' '4978' 'c:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment\Entregavel\task8_b.py'

Inserir o tamanho do array desejado: 5000000
O tempo total de execução foi: 10.847373247146696

O número de cores do meu processador é: 4
PS C:\Users\Bianca\Google Drive\Documents\Bloco Python\Desenvolvimento Python para Redes e Sistemas\Assessment>

Questão 8c para um array de tamanho 5.000.000:

Sem sucesso....

Então, o tempo total de execução da questão 8a para 5.000.000 foi de 10.95 segundos, e o tempo total de execução da questão 8b para 5.000.000 foi de 10.84 segundos. Apesar de a diferença ser pequena, isso demonstra que a thread costuma funcionar de maneira muito mais eficiente e rápida, pelo fato de distribuir a tarefa para executar uma única função.

Enquanto de forma sequencial, a função é executada (como o nome já sugere), sequencialmente. Então, uma única thread realiza todo o trabalho.

Questão 8a para um array de tamanho 10.000.000:

The screenshot shows the Visual Studio Code interface with the file `task8_a.py` open. The code defines a `main()` function that takes an input value and appends 20 to a vector. It also defines a `calcula_fatorial_do_vetor()` function that calculates the factorial of each element in the vector and appends the result to another vector. A `get_core_cpu()` function returns the number of CPU cores. The `if __name__ == '__main__':` block calls `main()` and prints the initial time and the result of `calcula_fatorial_do_vetor()`.

The terminal output shows the command prompt running the script with the input value 10000000. The output indicates that the total execution time was 23.069214582443237 seconds and the number of CPU cores is 4.

Questão 8b para um array de tamanho 10.000.000:

The screenshot shows the Visual Studio Code interface with the file `task8_b.py` open. The code defines a `get_core_cpu()` function that returns the number of CPU cores. The `if __name__ == '__main__':` block calls `get_core_cpu()` and sets the number of threads to 4. It then calculates the factorial of each element in the vector using a parallel approach with threads. The `main()` function calculates the factorial of each element in the vector and appends the result to another vector. The `if __name__ == '__main__':` block calls `main()` and prints the initial time and the result of `calcula_fatorial_do_vetor()`.

The terminal output shows the command prompt running the script with the input value 10000000. The output indicates that the total execution time was 2.3541781982313232 seconds and the number of CPU cores is 4.

Questão 8c para um array de tamanho 10.000.000:

Sem sucesso...

Então, o tempo total de execução da questão 8a para 10.000.000 foi de 23.06 segundos, e o tempo total de execução da questão 8b para 10.000.000 foi de 21.54 segundos.

E, novamente, a questão 8c, eu não consegui chegar nem tão longe....

E mais uma vez, podemos notar a diferença de tempo entre executar uma função em threads e sequenciais.

O de multiprocessamento é muito mais lento porque o sistema operacional abre um processo a cada chamada pelo script para executar a função. E dependendo do tamanho desse array, para apenas 4 processos, não é o suficiente.

Pois ele pode atribuir para cada processo, mas no 4o, ele trava, por não conseguir executar tantas coisas ao mesmo tempo.

Obs.: Em anexo também tenho imagens comprovando de que não consegui rodar a questão 8c em todas as opções de tamanho de array, de acordo com a hora em que iniciei e a hora em que forcei a parada do processo.