

Componente:

- Rafael Araújo de Andrade - 20230036431

Exercício 1:

Nessa questão, eu criei um processo filho e um processo neto para o pai, e quando eu chamava qualquer um dos dois, somava o mesmo valor ao valor inicial, passando como endereço do valor, o local de memória compartilhado entre os 3. Nas seguintes capturas de tela, mostra como estava saindo o valor final:

```
Sou o processo 1248 e o valor da variável é 100
Olá, processo pai, sou o seu neto e o meu ID é 1249 e adicionei 100 ao valor que o pai me deu. O valor agora é: 200
Olá, processo pai, sou o seu neto e o meu ID é 1250 e estou adicionando 200 ao valor que o filho me deu. O valor agora é: 400
Olá novamente, sou o processo pai e o valor é: 400

Run Ask AI 221ms • 3 minutes ago ✓

Sou o processo 1282 e o valor da variável é 50
Olá, processo pai, sou o seu neto e o meu ID é 1283 e adicionei 50 ao valor que o pai me deu. O valor agora é: 100
Olá, processo pai, sou o seu neto e o meu ID é 1284 e estou adicionando 100 ao valor que o filho me deu. O valor agora é: 200
Olá novamente, sou o processo pai e o valor é: 200

Run Ask AI 189ms • 3 minutes ago ✓

Sou o processo 1307 e o valor da variável é 80
Olá, processo pai, sou o seu neto e o meu ID é 1308 e adicionei 80 ao valor que o pai me deu. O valor agora é: 160
Olá, processo pai, sou o seu neto e o meu ID é 1309 e estou adicionando 160 ao valor que o filho me deu. O valor agora é: 320
Olá novamente, sou o processo pai e o valor é: 320
```

O código da questão pode ser encontrado em:

<https://github.com/leafaraujo/SO/blob/bf7c97182a9a4228ec0b39f9ba80dc5484749201/LAB2/questionOne.c>

Exercício 2:

No começo, foi um pouco difícil de concatenar todas as saídas esperadas pela questão, mas com um pouco de pesquisa, e com o código resposta colocado pela professora no primeiro laboratório virtual, consegui concatenar todos os valores da sequência de fibonacci, segue a captura de tela com as saídas obtidas:

O código da questão pode ser encontrado em:

<https://github.com/leafaraujo/SO/blob/c6e9e889db8bbc3bc35c8cfcffc2657b1e50f267/LAB2/questionTwo.c>

Run	Ask AI	239ms • Just now	✓
Sou o processo pai, vou criar um filho para calcular a sequência de Fibonacci com 20 termos. Sou o processo pai, essa é a sequência de Fibonacci gerada pelo meu filho: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765			
Run	Ask AI	228ms • Just now	✓
Sou o processo pai, vou criar um filho para calcular a sequência de Fibonacci com 10 termos. Sou o processo pai, essa é a sequência de Fibonacci gerada pelo meu filho: 0 1 1 2 3 5 8 13 21 34 55			
Run	Ask AI	180ms • Just now	✓
Sou o processo pai, vou criar um filho para calcular a sequência de Fibonacci com 5 termos. Sou o processo pai, essa é a sequência de Fibonacci gerada pelo meu filho: 0 1 1 2 3 5			
Run	Ask AI	185ms • Just now	✓
Sou o processo pai, vou criar um filho para calcular a sequência de Fibonacci com 25 termos. Sou o processo pai, essa é a sequência de Fibonacci gerada pelo meu filho: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946 17711 28657 46368 75025			
Run	Ask AI	197ms • Just now	✓
Sou o processo pai, vou criar um filho para calcular a sequência de Fibonacci com 9 termos. Sou o processo pai, essa é a sequência de Fibonacci gerada pelo meu filho: 0 1 1 2 3 5 8 13 21 34			

Exercício 3:

Nesse exercício, a minha maior dificuldade foi em controlar a quantidade de filhos que estavam sendo criados durante o laço for com o valor de n, mas consegui matar o processo filho durante o percorrimento de seu laço, e com isso realizar a soma de forma certa, e com o valor sendo o esperado no final do programa, quando impresso pelo processo 1 na memória compartilhada.

O código da questão pode ser encontrado em:

<https://github.com/leafaraujo/SO/blob/acf1f90f300d06dddaf7900b2eae8bed5c132589/LAB2/questionThree.c>

```
Sou o processo pai 485, vou criar um filho para fazer somas sucessivas.
Sou o processo filho 486, e vou somar por aqui o valor 1!
Sou o processo filho 487, e vou somar por aqui o valor 2!
Sou o processo filho 488, e vou somar por aqui o valor 3!
Sou o processo filho 489, e vou somar por aqui o valor 4!
Sou o processo filho 490, e vou somar por aqui o valor 5!
Sou o processo filho 491, e vou somar por aqui o valor 6!
Sou o processo filho 492, e vou somar por aqui o valor 7!
Sou o processo filho 493, e vou somar por aqui o valor 8!
Sou o processo filho 494, e vou somar por aqui o valor 9!
Sou o processo filho 495, e vou somar por aqui o valor 10!
O valor final da soma é: 55
```

Run

Ask AI

182ms • 3 minutes ago



```
Sou o processo pai 529, vou criar um filho para fazer somas sucessivas.
Sou o processo filho 530, e vou somar por aqui o valor 1!
Sou o processo filho 531, e vou somar por aqui o valor 2!
Sou o processo filho 532, e vou somar por aqui o valor 3!
Sou o processo filho 533, e vou somar por aqui o valor 4!
Sou o processo filho 534, e vou somar por aqui o valor 5!
O valor final da soma é: 15
```

Run

Ask AI

192ms • 3 minutes ago



```
Sou o processo pai 558, vou criar um filho para fazer somas sucessivas.
Sou o processo filho 559, e vou somar por aqui o valor 1!
Sou o processo filho 560, e vou somar por aqui o valor 2!
O valor final da soma é: 3
```

Exercício 4:

Esse é o output do código retirado diretamente do laboratório virtual:

Run

Ask AI

200ms • 1 minute ago



```
Olá, eu sou a thread 0
Olá, eu sou a thread 1
Olá, eu sou a thread 2
Olá, eu sou a thread 3
Olá, eu sou a thread 4
```

Ao retirar, a linha de código 33 que tem o comando “pthread_join(threads[i],NULL);”, o código apresenta resultados inesperados que podem ser vistos a seguir:

```
Run Ask AI 208ms • 3 minutes ago ✓
Olá, eu sou a thread 0
Olá, eu sou a thread 2
Olá, eu sou a thread 3

Run Ask AI 23ms • 3 minutes ago ✓
Olá, eu sou a thread 0
Olá, eu sou a thread 1
Olá, eu sou a thread 2

Run Ask AI 34ms • 3 minutes ago ✓

Run Ask AI 36ms • 2 minutes ago ✓
Olá, eu sou a thread 0
Olá, eu sou a thread 1
Olá, eu sou a thread 2
```

Isso acontece porque a thread principal não está esperando pelo fim da filha, pode-se ver pela terceira execução que inclusive, o programa não conseguiu imprimir nenhuma das threads (o que pode ser erro do compilador, não sei ao certo). Consegui resolver esse problema colocando o programa para dormir logo após a declaração da thread utilizando a função sleep do C, e assim ele esperava que a próxima thread (filha) fosse concluída para encerrar todo o programa.

O código pode ser encontrado no seguinte repositório:

<https://github.com/leafaraujo/SO/blob/main/LAB2/questionFour.c>