



1º TP

Cálculo do Número π

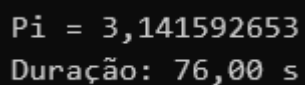
Segunda-feira, 16 de outubro de 2023.

O objetivo deste trabalho é usar os conceitos de sistemas operacionais na implementação de um programa que realiza o cálculo do número π ¹ com nove casas decimais²: 3,141592653. A matemática que deve ser usada para calcular esse valor é a [Fórmula de Leibniz para \$\pi\$](#) mostrada abaixo.

$$\sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} = \frac{\pi}{4}$$

O programa deve calcular o número π usando 2 bilhões de termos da série acima.

A imagem abaixo mostra que foi necessário 76 segundos para calcular o número π usando um computador com processador de 4 núcleos e 8 *threads*.



```
Pi = 3,141592653
Duração: 76,00 s
```

Para reduzir o tempo de geração do número π o programa deve usar processos e *threads*. Cada *thread* será responsável por calcular uma quantidade de termos da série de Leibniz e, sem usar mecanismos de sincronização, compartilhar o resultado com o processo que criou a *thread*. Isso deve ser feito usando o procedimento abaixo.

1. O programa deve criar dois processos, logo serão três processos: pi (processo pai), pi1 (processo filho 1) e pi2 (processo filho 2). O processo pai deve ser usado apenas para criar os processos filhos e exibir a mensagem Processo pai finalizou sua execução. Após isso ele deve ser finalizado.
2. Cada processo filho deve criar 16 *threads*, logo teremos a geração do número π ocorrendo simultaneamente e de modo independente em dois processos (pi1 e pi2).
3. Cada *thread* em cada processo filho deve calcular a soma parcial de 125 milhões de termos da série de Leibniz e armazenar em uma variável local do processo filho. Não se deve usar variáveis globais.

¹ 18 Ways NASA Uses Pi: <https://www.jpl.nasa.gov/edu/learn/list/oh-the-places-we-go-18-ways-nasa-uses-pi/>

² A NASA usa 15 casas decimais no número π para cálculos de maior precisão, como os de navegação interplanetária.
Fonte: <https://www.jpl.nasa.gov/edu/news/2016/3/16/how-many-decimals-of-pi-do-we-really-need/>

4. Após a conclusão de cada *thread* o processo filho deve calcular a soma total a partir das somas parciais geradas pelas *threads*, como são 16 *threads* tem-se os 2 bilhões (16 x 125.000.000) de termos supracitados para se calcular o número π .

O processo filho deve obter do sistema operacional o tempo antes de iniciar a criação das 16 *threads* e o tempo de término após calcular a soma total a partir das somas parciais geradas pelas *threads*. Assim, a duração a ser calculada pelo processo filho considera todo o comportamento *multithread* criado pelo programa para o cálculo do número π .

A mensagem de identificação de cada processo filho, o número de *threads*, o tempo de início e término, a duração (em segundos) e o valor do número π (com nove casas decimais) deve ser exibido no relatório abaixo por cada processo filho.

O programa deve usar o leiaute abaixo para exibir o resultado.

Cálculo do Número π

Criando os processos filhos pi1 e pi2...
Processo pai (PID 6923) finalizou sua execução.

- Processo Filho: pi1 (PID 6924)

Nº de threads: 16

Início: 10:45:12

Fim: 10:45:21

Duração: 9,59 s

Pi = 3,141592653

- Processo Filho: pi2 (PID 6925)

Nº de threads: 16

Início: 10:45:14

Fim: 10:45:24

Duração: 10,00 s

Pi = 3,141592653

Cada processo filho deve criar no diretório atual um arquivo (pi1.txt ou pi2.txt) para armazenar os tempos em segundos que cada *thread* gastou para calcular apenas a soma parcial dos 125 milhões de termos da série de Leibniz.

Esse relatório deve usar o leiaute exibido na imagem abaixo.

Observe que o tempo de cada *thread* deve ser gravado em uma linha do texto usando o formato TID NNN: tempo, onde NNN é o número de identificação da *thread* obtida com a chamada de sistema `gettid` e tempo é o valor em segundos. O valor total no fim do arquivo corresponde ao somatório dos tempos de cada *thread*.

```
pi1.txt x
1  Arquivo: pi1.txt
2  Descrição: Tempo em segundos das 16 threads do processo filho pi1.
3
4  TID 150: 0,63
5  TID 151: 0,60
6  TID 152: 0,59
7  TID 153: 0,57
8  TID 154: 0,61
9  TID 155: 0,48
10 TID 156: 0,55
11 TID 157: 0,58
12 TID 158: 0,50
13 TID 159: 0,60
14 TID 160: 0,70
15 TID 161: 0,42
16 TID 162: 0,75
17 TID 163: 0,73
18 TID 164: 0,65
19 TID 165: 0,63
20
21 Total: 9,59 s
```

- Critérios de avaliação

O trabalho será avaliado considerando:

1. Estrutura da solução:

- A validação dos dados fornecidos pelo usuário.
- A lógica empregada na solução do problema.
- O funcionamento do programa.
- Escrever funções específicas, ou seja, com atribuição clara e objetiva.
- Não escrever código redundante.
- Código-fonte sem erros e sem advertências do compilador.
- Código-fonte legível, indentado, organizado e comentado.
- Identificadores significativos para aprimorar a inteligibilidade do código-fonte.

2. O programa deve ser desenvolvido no sistema operacional Linux usando apenas a Linguagem C padrão ISO³ e a GNU C Library⁴, que inclui a API POSIX. A programação das *threads* deve ser feita usando a biblioteca *Pthreads* (POSIX *threads*). Programas desenvolvidos em outras linguagens, mesmo que parcialmente, receberão nota zero.

3. Para que o programa seja avaliado, o código deve executar com sucesso. Programas que apresentarem erros de compilação, ligação ou *segmentation fault* receberão nota zero.

³ Documentação da Linguagem C padrão ISO: <https://en.cppreference.com/w/c>

⁴ Documentação da GNU C Library: <https://www.gnu.org/software/libc/>

4. Trabalho com plágio, ou seja, programa com código-fonte copiado de outra pessoa ou qualquer fonte disponível na Internet (cópia integral ou parcial) receberão nota zero.
5. O desenvolvimento do trabalho é individual.
6. O programa deve ser composto de um único arquivo de cabeçalho (pi.h) e um único código-fonte (pi.c).
7. O código-fonte (.c) e o arquivo de cabeçalho (.h) devem ser codificados como UTF-8.
8. É proibido modificar os nomes de arquivos, identificadores, os protótipos de função, as definições de constantes e variáveis fornecidos neste texto ou em anexo.
9. Deve-se implementar o programa usando as definições fornecidas no arquivo anexo pi.h.
10. É permitido acrescentar novas declarações e/ou definições de tipos de dados, variáveis, constantes e funções, desde que estejam de acordo com os critérios acima.

- Instruções para entrega do trabalho

1. Compacte os dois arquivos para criar um arquivo 7z com o seu nome e sobrenome, por exemplo: NelsonMandela. Use o *software* livre 7-Zip, que está disponível em <https://www.7-zip.org/download.html>.
2. Coloque o arquivo 7z na atividade criada no Google *Classroom* para esse Trabalho de Programação.

- Data de entrega

Sexta-feira, 27 de outubro de 2023.

- Valor do trabalho

10,0 pontos

Prof. Márlon Oliveira da Silva
marlon.silva@ifsudestemg.edu.br