Instruções para Executar o Programa

Alunos:

- **Pedro Magalhães - 1611074**
- **Marcelo Costalonga - 1421229**

```
Formas de executar:
  Execucao Unica:
  $ make
  $ ./quadratura <NumThreads> <a> <b> <Tol> <V> <Fx>

  10 Execucoes Seguidas pegando o resultado da Mediana:
  $ bash quadratura.sh <NumThreads> <a> <b> <Tol> <V> <Fx>
```

Onde:

- <V> corresponde à versão da implementação (valor inteiro 1 ou 2):
  - 1: Sem bag of tasks
  - 2: Utiliza bag of tasks

- <Fx> Parâmetro **opcional** (valor inteiro 1, 2 ou 3) corresponde a função utilizada:
  - 1: Fx1 = $\int 10000 \exp(3x) / \sin(2x)\, dx$
  - 2: Fx2 = $\int 100000 \exp(x)\, dx$
  - 3: Fx3 = $\int 100x^2\, dx$

É possível encontrar exemplos de execuções com respostas na secção de Experimentos, mais em diante.

Apesar de demorar mais, na maioria de nossos testes utilizamos o script para executar 10 vezes o programa, ordernar os resultados e pegar a mediana para tentar reduzir possíveis outliers em nossos resultados.

Experimentos

Resultados para função:

- Fx1:
  - Intervalo 0 à 1
  - Área total estimada ≈ 56544.8049414393

$$\int_0^1 10\,000\,(e^{3x}\sin(2x))\,dx$$

Para Implementação 1 (v1) com NumThreads = 1 e Tol = 0.001

```
(base) [Marcelo@Marcelo T2]$ bash quadratura.sh 1 0 1 0.001 1 1
make: Nothing to be done for 'all'.
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.559897
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.569350
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.545829
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.541727
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.561253
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.540577
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.539922
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.540192
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.557096
Nthreads=1, a=0, b=1, Tol=0.001000, Implementacao=1, Funcao=fx1
Implementacao 1: Area somada = 56544.846335 | tempo = 0.545811

        Area somada = 56544.846335
        Mediana: tempo = 0.545811s
```

Para Implementação 1 (v1) com NumThreads = 2 e Tol = 0.001
($ bash quadratura.sh 2 0 1 0.001 1 1)

```
Area somada = 56544.846335
Mediana: tempo = 0.277340s
```

Para Implementação 1 (v1) com NumThreads = 4 e Tol = 0.001
($ bash quadratura.sh 4 0 1 0.001 1 1)

```
Area somada = 56544.846335
Mediana: tempo = 0.163730s
```

Assim, podemos ver que os resultados da Implementação 1 (v1) estão de acordo com o esperado, aumentado o numero de threads o tempo de execução do programa diminui.

Entretanto, para a Implementação 2 (v2), não conseguimos observar o mesmo comportamento, acreditamos que isso foi devido à nossa implementação, fazendo muitas operações envolvendo regiões críticas, o que provavelmente atrasou o tempo de execução de v2.

Para Implementação 2 (v2) com NumThreads = 1 e Tol = 0.001
($ bash quadratura.sh 1 0 1 0.001 2 1)

```
Area somada = 56544.846335
Mediana: tempo = 0.545359s
```

Para Implementação 2 (v2) com NumThreads = 2 e Tol = 0.001
($ bash quadratura.sh 2 0 1 0.001 2 1)

```
Area somada = 56544.846335
Mediana: tempo = 0.540478s
```

Para Implementação 2 (v2) com NumThreads = 4 e Tol = 0.001
($ bash quadratura.sh 4 0 1 0.001 2 1)

```
Area somada = 56544.846335
Mediana: tempo = 0.551120s
```

Esse comportamento de v2 fica mais evidente utilizando uma tolerância mais baixa, para aumentar a precisão, ou utilizando Fx2 (que cria uma carga de processamento maior)

- Fx2:
  - Intervalo 0 à 1
  - Área total estimada ≈ 171828.182845905

$$\int_0^1 100\,000\, e^x \, dx$$

Para Implementação 2 (v2) com NumThreads = 1 e Tol = 0.001
($ bash quadratura.sh 1 0 1 0.001 2 2)

```
Area somada = 171828.236714
Mediana: tempo = 1.829441s
```

(Reduzindo a tolerância para 0.00001)
Para Implementação 2 (v2) com NumThreads = 1 e Tol = 0.00001
($ bash quadratura.sh 1 0 1 0.00001 2 2)

```
Area somada = 171828.183199
Mediana: tempo = 10.674284s
```

(Aumentando o número de Threads para 4)
Para Implementação 2 (v2) com NumThreads = 4 e Tol = 0.00001
($ bash quadratura.sh 4 0 1 0.00001 2 2)

```
Area somada = 171828.184769
Mediana: tempo = 12.910605s
```

OBS: Notamos também que a precisão da Implementação 2 (v2) está limitada pelo tamanho da nossa fila (bag of tasks), porque, como esperado, quanto menor a tolerância, mais tarefas irão ser criadas e caso a as tarefas não consigam acabar na mesma velocidade que novas tarefas estão chegando, nós acabamos perdendo tarefas e com isso o resultado fica pior. O que não acontece com a implementação 1 (v1), já que nesse caso, as tarefas não vão sendo armazenadas em uma fila.

Exemplo: Variando a precisão da função Fx3 para v1 e v2

```
                        Implementacao 2 (v2) usando Fx3, variando precisao

(base) [Marcelo@Marcelo T2]$ ./quadratura 4 1 3 0.000000001 2 3
Nthreads=4, a=1, b=3, Tol=0.000000, Implementacao=2, Funcao=fx3
Implementacao 2: Area somada = 866.666667 | tempo = 0.021700
(base) [Marcelo@Marcelo T2]$
(base) [Marcelo@Marcelo T2]$ ./quadratura 4 1 3 0.00000000000001 2 3
Nthreads=4, a=1, b=3, Tol=0.000000, Implementacao=2, Funcao=fx3
Implementacao 2: Area somada = 465.319975 | tempo = 0.364932
```

```
                        Implementacao 1 (v1) usando Fx3, variando precisao

(base) [Marcelo@Marcelo T2]$ ./quadratura 4 1 3 0.000000001 1 3
Nthreads=4, a=1, b=3, Tol=0.000000, Implementacao=1, Funcao=fx3
Implementacao 1: Area somada = 866.666667 | tempo = 0.004896
(base) [Marcelo@Marcelo T2]$
(base) [Marcelo@Marcelo T2]$ ./quadratura 4 1 3 0.00000000000001 1 3
Nthreads=4, a=1, b=3, Tol=0.000000, Implementacao=1, Funcao=fx3
Implementacao 1: Area somada = 866.666667 | tempo = 0.115411
```