

Pedro Inácio Rodrigues Pontes

## **Prática 5: Jantar dos Filsósofos(Threads)**

Belo Horizonte, Brasil

2025

# 1 Introdução

O objetivo da presente prática foi a implementação do Jantar dos Filósofos, tendo em vista o aprendizado do gerenciamento de recursos compartilhados em um ambiente concorrente.

Tal situação consiste em, de acordo com o enunciado da prática, "Imagine cinco filósofos sentados em torno de uma mesa circular, com um prato de comida em frente a cada um. Entre cada par de pratos, há um garfo. Os filósofos passam o tempo pensando e, ocasionalmente, tentam comer. Para comer, um filósofo precisa ter acesso a dois garfos: o garfo à sua esquerda e o garfo à sua direita. Se um filósofo não conseguir obter os dois garfos, ele não pode comer e deve continuar pensando."

## 2 Desenvolvimento

Para estruturar a implementação dos resultados, foram criadas as classes Program, Philosopher e Console Lock.

Program é a main, gerenciando o programa e a atuação de suas classes. Philosopher implementa os filósofos e suas lógicas, com destaque à de gerenciamento de recursos. Console-Lock é uma classe simples para as Threads escreverem no console, já implementando o Lock para evitar conflitos entre elas (race conditions).

A seguir temos um detalhamento das partes mais significativas de Philosopher e Program.

### 2.1 Philosopher

Pode se definir o comportamento inicial básico da classe vendo seu construtor:

```
public Philosopher(string name, string thought,
    SemaphoreSlim[] forks)
{
    Name = name;
    Thought = thought;
    Id = Interlocked.Increment(ref _nextId);
    Forks = forks;
}
```

A lógica do filósofo é representada em Live():

```
public void Live()
{
    Think();
    Eat();
}
```

```
}
```

Em Think ele simplesmente emite o pensamento definido em seu construtor, antecedido por sua identificação pelo nome.

Em Eat primeiramente é chamado o método GetFork, que consiste no método para o filósofo obter os garfos, que são gerenciados a partir do SemaphoreSlim, assim é chamado um Wait para os garfos que corresponde à esquerda e direita dele (calculadas a partir do seu id). É feita também uma lógica para implementar uma lógica fixa de obtenção dos garfos (menor para o maior), evitando o um ciclo de espera circular que caracteriza o deadlock. Este fenômeno seria causado quando todos os filósofos tentarem pegar os garfos ao mesmo tempo, e o filósofo 5 tentaria pegar o garfo 5 antes do 1, o que ocorre seguindo a lógica normal de obtenção, e com isso, o filósofo 4 não conseguiria pegar o seu segundo garfo, que já estaria sendo usado pelo 5, e o 5 não conseguiria pegar seu segundo também, que já estaria sendo usado pelo 1. Com a implementação desse ordenamento não ocorre tal ciclo, pois o filósofo 5 primeiro tenta pegar o garfo 1 e depois o 5.

Após o GetFork o filósofo emite uma mensagem demonstrando que está comendo e, por fim, libera esses garfos a partir de ReleaseFork, que chama o método Release do SemaphoreSlim nos garfos utilizados.

Segue o método principal de gerenciamento de recursos GetFork:

```
public void GetFork()
{
    var leftFork = getLeftFork();
    var rightFork = getRightFork();
    if (leftFork > rightFork)
    {
        var temp = leftFork;
        leftFork = rightFork;
        rightFork = temp;
    }
    Forks[leftFork].Wait();
    Forks[rightFork].Wait();
}
```

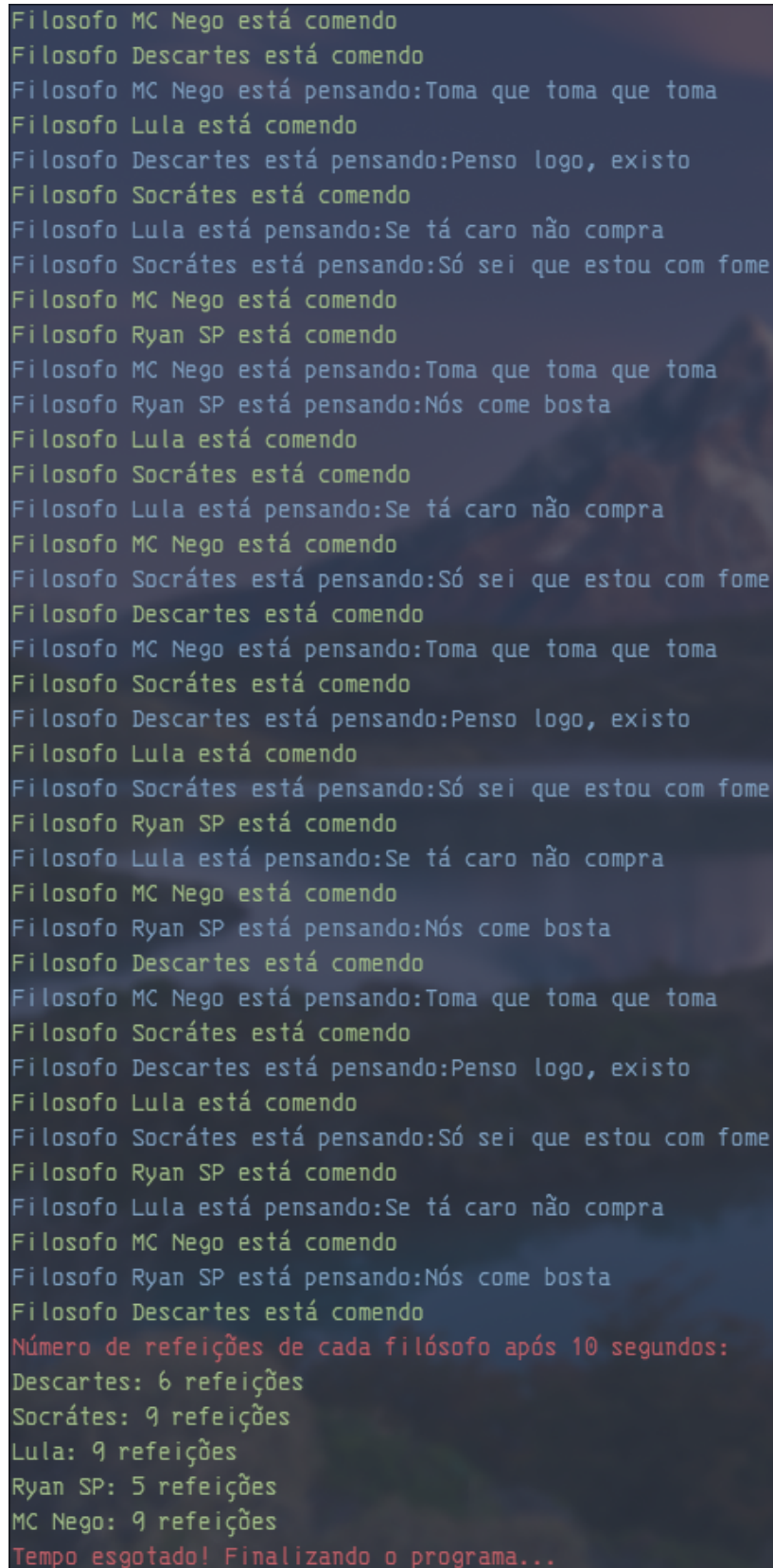
A classe também possui um método Start, que gera uma Task que roda infinitamente o ciclo Live(), que está dentro de um While(true)

## 2.2 Program

Cria e inicializa o Array de Filósofos e o dos garfos SemaphoreSlim. Segue seu cerne:

```
for(int i = 0; i < 5; i ++)  
{  
    forks[i] = new SemaphoreSlim(1, 1);  
}  
for (int i = 0; i < philosophers.Length; i++)  
{  
    philosophers[i] = new Philosopher(names[i], thoughts[i],  
        forks);  
    philosophers[i].Start();  
}
```

### 3 Resultados



```
Filosofo MC Nego está comendo
Filosofo Descartes está comendo
Filosofo MC Nego está pensando:Toma que toma que toma
Filosofo Lula está comendo
Filosofo Descartes está pensando:Penso logo, existo
Filosofo Sócrates está comendo
Filosofo Lula está pensando:Se tá caro não compra
Filosofo Sócrates está pensando:Só sei que estou com fome
Filosofo MC Nego está comendo
Filosofo Ryan SP está comendo
Filosofo MC Nego está pensando:Toma que toma que toma
Filosofo Ryan SP está pensando:Nós come bosta
Filosofo Lula está comendo
Filosofo Sócrates está comendo
Filosofo Lula está pensando:Se tá caro não compra
Filosofo MC Nego está comendo
Filosofo Sócrates está pensando:Só sei que estou com fome
Filosofo Descartes está comendo
Filosofo MC Nego está pensando:Toma que toma que toma
Filosofo Sócrates está comendo
Filosofo Descartes está pensando:Penso logo, existo
Filosofo Lula está comendo
Filosofo Sócrates está pensando:Só sei que estou com fome
Filosofo Ryan SP está comendo
Filosofo Lula está pensando:Se tá caro não compra
Filosofo MC Nego está comendo
Filosofo Ryan SP está pensando:Nós come bosta
Filosofo Descartes está comendo
Filosofo MC Nego está pensando:Toma que toma que toma
Filosofo Sócrates está comendo
Filosofo Descartes está pensando:Penso logo, existo
Filosofo Lula está comendo
Filosofo Sócrates está pensando:Só sei que estou com fome
Filosofo Ryan SP está comendo
Filosofo Lula está pensando:Se tá caro não compra
Filosofo MC Nego está comendo
Filosofo Ryan SP está pensando:Nós come bosta
Filosofo Descartes está comendo
Número de refeições de cada filósofo após 10 segundos:
Descartes: 6 refeições
Sócrates: 9 refeições
Lula: 9 refeições
Ryan SP: 5 refeições
MC Nego: 9 refeições
Tempo esgotado! Finalizando o programa...
```

Figura 1 – Filósofos emitindo seus pensamentos e comendo, com um log final de quantas vezes cada um comeu em 10 segundos de execução

## 4 Conclusão

Todos os resultados foram alcançados, sendo evitados os deadlocks e a starvation. O método da ordem fixa de obtenção dos garfos foi o melhor para o contexto, pois sua alternativa seria um método tentativa com falha, o qual pode vir a gerar starvation se não implementado adequadamente, pois um filósofo pode conseguir os dois garfos constantemente, evitando que outros consigam os dois garfos e causando deles sempre abandonarem a tentativa em que estavam de comer por falha. Isso seria evitado implementando uma fila ou política de acesso aos garfos, mas deixaria o programa mais complexo que com o método da ordem fixa, além de pior em relação a desempenho pelos filósofos poderem abandonar suas tentativas se os dois garfos não estiverem disponíveis no momento exato em que o método para comer for iniciado.