Gabriel Diniz Gisoldo **RA: 222140007-1**

Para simular o funcionamento, eu transformei os valores absolutos dos minutos para segundo seguindo a regra:

60 minutos --> 3600 segundos --> 36 segundos

Dessa forma, considerando um dia de 8 horas, surge um cliente a cada 18 segundos e o barbeiro trabalha entre 12 e 18 segundos e fica aberto para clientes por 288 segundos.

O algoritmo foi escrito em *Python3* usando a biblioteca *threading* para executar em múltiplas *threads* e para controle da fila de clientes, foi usada a biblioteca *queue* que pode ser acessada por vários *threads* simultaneamente e já possui um *lock* interno.

O barbeiro fica de olho na fila e, no momento que um cliente entra na fila, se o barbeiro estiver dormindo, ele acorda e já atende o cliente.

Não foi usado um *mutex* explicito, pois, como foi feito uso da estrutura de dados *Queue* do Python, não é necessário. A *Queue* já faz uso de dois *locks* internamente, um para inserção e outro para remoção.

Nesse código o barbeiro é executado em uma *thread* enquanto cada cliente é uma *thread* a parte. A *thread* de um cliente, simplesmente, tenta inserir ele na fila do barbeiro, gerando um *lock* de inserção na *Queue* que tem um *timeout* de 2 segundos (dentro da simulação 0,02 segundos), se não abrir um lugar na fila, o *timeout* é invocado, *lock* de inserção é liberado e o cliente é considerado como recusado. Caso tenha lugar na fila, o cliente é inserido na fila do barbeiro, o *lock* de inserção e liberado e a *thread* do cliente espera a chamada de *join*.

A *thread* do barbeiro por sua vez, tenta remover um cliente da fila, enquanto não existe cliente na fila, um *timer* é usado para contar o tempo no qual ele dormiu. Esse ato de tentar remover um cliente da fila, invoca o *lock* de remoção da *Queue* e, como nesse processo não tem um *timeout*, o *lock* vai permanecer até o barbeiro atender um cliente. Quando um cliente é inserido na fila, o barbeiro realiza a remoção do cliente da fila, liberando o *lock* de remoção da *Queue* e inicia um *sleep* entre 12 e 18 segundos, para simular o tempo gasto trabalhado. Depois desse tempo ele chama o *join* da *thread* do cliente para finalizar essa *thread* e aguardar o próximo cliente.

O algoritmo usado para solucionar o problema do barbeiro dorminhoco pode ser encontrado no link:

https://github.com/huine/sistemas-distribuidos/blob/master/aula6/barbeiro.py

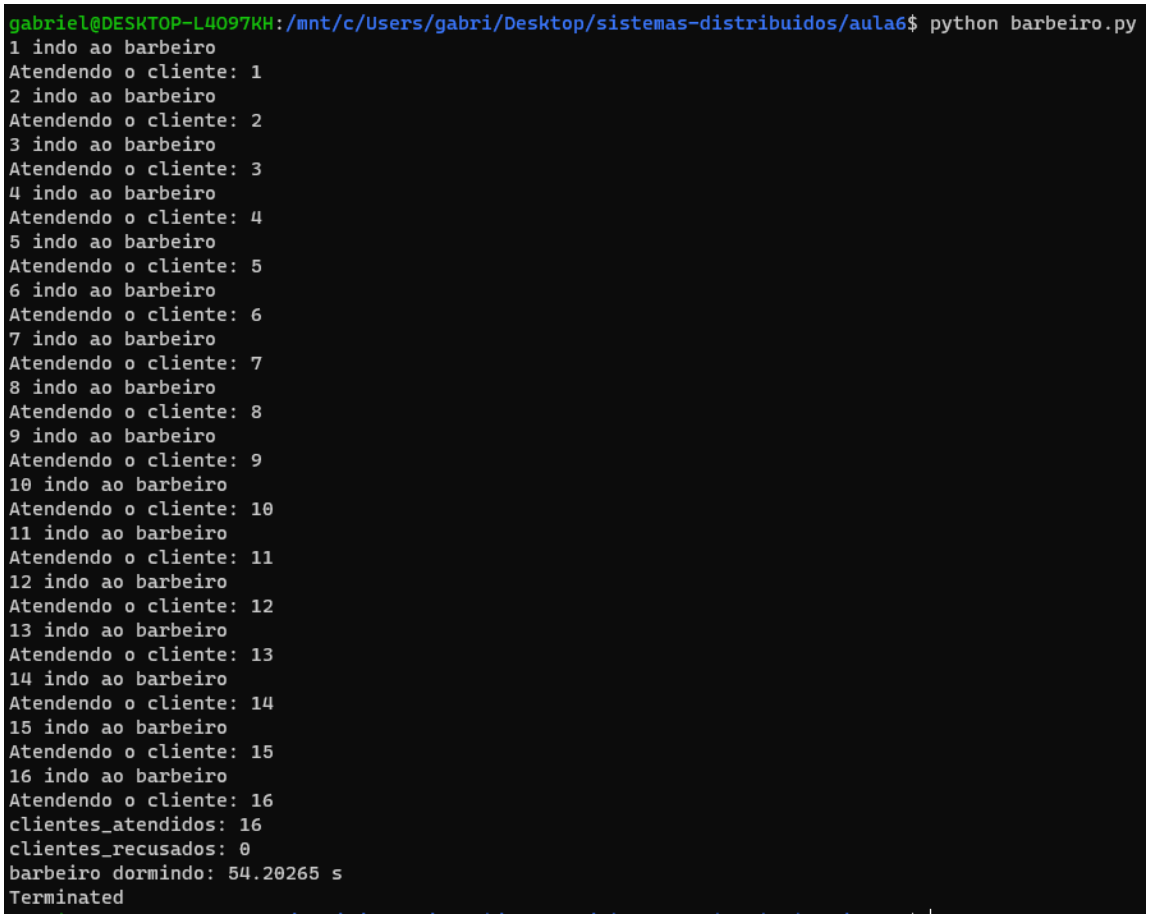

Figura 1 Output do programa

Clientes atendidos: 16

Clientes recusados: 0

Tempo dormindo: 32,52159 minutos (Dentro da simulação) ou 54,20265 segundos (Tempo real)