Programação Concorrente
Mecanismos de sincronização

Alexandre Guimarães

Ângelo Gustavo

1 Como a solução foi projetada

Seguindo as diretrizes do projeto, temos um banheiro unissex no qual se homens tiverem no banheiro outros homens podem entrar. Caso mulheres queiram entrar no banheiro elas teriam que esperar os homens saírem para que elas possam entrar, com mulheres dentro do banheiro outras mulheres podem entrar e portanto a mesma regra de entrada vai se aplicar para os homens. A partir disso, foi desenvolvido um programa concorrente que trata os casos de mulheres querem entrar ao mesmo tempo que homens e vice-versa. Essa solução foi projetada com o auxilio do conceito de semáforos. Utilizamos o AtomicInteger para gerenciar a ocupação do banheiro e utilizamos semáforos para a entrada no banheiro e para o gênero.

O projeto pode ser Encontrado no seguinte repositório:
`https://github.com/AngeloGustavo/Trabalho_Uni02_PC`

2 Lógica de sincronização utilizada

A classe Banheiro cria 5 objetos da classe ThreadPessoa com gêneros aleatórios, e cada um dos 5 cria uma thread na qual adiciona 10 pessoas à 'fila' do banheiro. Do começo ao fim passarão 50 objetos Pessoa pelo banheiro. Primeiramente, há um semáforo com 1 vaga para controle do gênero que está usando o banheiro no momento. Além disso, há um semáforo com 5 vagas para controle do número de pessoas que podem usar o banheiro ao mesmo tempo.

3 Como é garantida a corretude da solução com relação a concorrência

Conseguimos garantir a corretude do código utilizando de semáforos, utilizamos também o AtomicInteger que nos deu a possibilidade de não precisar usar mutex. Acabamos utilizando de dois semaforos, um para a entrada de pessoas no banheiro e outro para o genero

4 Dificuldades encontradas durante o desenvolvimento

O momento de passar de uma fila com pessoas independentes do gênero para uma fila que depende de gênero teve alguns contratempos. Além disso, o uso do inteiro normal (o qual depende de um Mutex, isto é, um Semáforo de 1 espaço) para a contagem da ocupação do banheiro trouxe problemas, até que começamos a utilizar o AtomicInteger, deixando a necessidade do uso de um Mutex de lado e, assim, garantindo corretude.

5 Instruções para compilação e execução do programa

```
1 javac Banheiro.java
2
3 java Banheiro
```

Listing 1: Instruções de compilação