UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL

Raíssa de Jesus Pereira dos Santos

Neylane Pereira Lopes

Trabalho Prático - Mecanismos de Sincronização

NATAL, RN
2022

Raíssa de Jesus Pereira dos Santos

Neylane Pereira Lopes

Trabalho Prático - Mecanismos de Sincronização

Relatório técnico apresentado a disciplina Programação Corrente para obtenção de nota parcial da segunda unidade do semestre letivo 2022.2 do curso Bacharelado em Tecnologia da Informação pela Universidade Federal do Rio Grande do Norte.

Orientador: Prof.ª Dr. Everton Ranielly

NATAL, RN
2022

Sumário

1 INTRODUÇÃO 4

2 METODOLOGIA 5
2.1 Solução projetada 5
2.2 Lógica de sincronização utilizada 5
2.3 Corretude da solução em relação a Concorrência 5
2.4 Materiais utilizados 5
2.4.1 Ambiente de Execução 5
2.4.2 Ferramentas e Linguagem de Programação 6
2.5 Compilação e execução do programa 6
2.6 Dificuldades encontradas durante o desenvolvimento 6

3 RESULTADOS 7
3.1 Resultados de Execução 7

4 CONCLUSÃO 8

1 INTRODUÇÃO

Este presente relatório tem como objetivo descrever a solução concorrente proposta em Java para o Problema do Banheiro Unissex.

O problema do Banheiro Unissex consiste de um banheiro que pode ser utilizado tanto por homens quanto por mulheres, mas não por ambos ao mesmo tempo. Se um homem estiver no banheiro, outros homens podem entrar, porém eventuais mulheres que desejem utilizar o banheiro devem esperar ele ficar vazio. Se uma mulher estiver no banheiro, outras mulheres podem entrar, porém eventuais homens que desejem utilizar o banheiro devem esperar ele ficar vazio. Cada pessoa (homem ou mulher) pode passar um determinado tempo utilizando o banheiro, que possui uma capacidade limite de pessoas que podem utilizá-lo ao mesmo tempo.

Nas seções seguintes, vai ser explicado todo o processo de implementação da solução projetada, vai ser mostrado a corretude da solução com relação a concorrência, a lógica de sincronização utilizada, as dificuldades encontradas durante o desenvolvimento e as instruções para compilação e execução do programa e, por último, os resultados obtidos serão comentados e discutidos.

2 METODOLOGIA

Essa seção tem como objetivo explicar minuciosamente a solução adotada para solucionar a problematica proposta, além de descrever as informações técnicas relacionadas a infraestrutura, ambiente e execução do programa.

2.1 Solução projetada

A solução projetada é composta pelas as classes Bathroom, Person, ThreadPerson e Main. A classe Bathroom controla a capacidade de pessoas no banheiro, o gênero que pode utilizar o banheiro ou ir para a fila, e quem saiu do banheiro. A classe Person contém características das pessoas que utilizam o banheiro, dessa forma possui o gênero, o tempo que a pessoa irá utilizar o banheiro e o identificador. A classe ThreadPerson é responsável por simular a solicitação e utilização do banheiro por uma pessoa, objeto da classe Person. Na classe Main é definida a capacidade limite do banheiro e o tempo que cada pessoa passa no banheiro, ademais ela é responsavél pela inicialização dos objetos Person e ThreadPerson e também por iniciar e finalizar a execução das threads.

2.2 Lógica de sincronização utilizada

Para a sincronização dos recursos utilizados pelas *threads*, utilizamos o *syncronized* tanto no método de entrar no banheiro, como o de sair do banheiro. Visto que nesses métodos, está presente a manipulação da lista responsável por armazenar o identificador de quais *threads* estão utilizando o banheiro no momento de verificação.

2.3 Corretude da solução em relação a Concorrência

A garantia da corretude em relação a concorrência, é dada através do mecanismo de sincronização, como mencionando anteriormente, a solução proposta adota o uso *syncronized methods*. Apesar de reduzir a concorrência, permite manter a consistência dos resultados, dado que apenas uma única *thread* poderá realizar uma operação, uma vez iniciada.

2.4 Materiais utilizados

2.4.1 Ambiente de Execução

Para a execução e compilação foi utilizado um ambiente de execução que usufrui de um sistema operacional com kernel linux (Ubuntu 20.04.5 LTS). Ademais, a máquina usada possui processador Intel Core i5-1035G1 CPU @ 1.00GHz $\times$ 8 . Quanto a memória RAM, possui 4GB.

2.4.2 Ferramentas e Linguagem de Programação

A linguagem de programação usada para implementar a solução foi a linguagem Java 11.0.16. A versão do compilador javac é a 11.0.16.

2.5 Compilação e execução do programa

Para compilar e executar o projeto, selecione o botão "Run"presente na sua IDE.

2.6 Dificuldades encontradas durante o desenvolvimento

Inicialmente tivemos dificuldade em perceber qual seria a lógica a ser aplicada para solucionar o problema proposto. Então, tentamos utilizar a ideia de produtor e consumidor, mas percebemos que não era a melhor solução a ser implementada, dessa forma não obtemos sucesso com o que haviamos pensado e mudamos a implementação da solução para a forma que já foi explicitada anteriormente.

3 RESULTADOS

Essa seção tem como objetivo exibir os resultados coletados, mostrando a entrada e saída de uma pessoa do banheiro, bem como quantas pessoas (homens ou mulheres) estão utilizando o banheiro no momento.

3.1 Resultados de Execução

Para o resultado da execução que pode ser visualizada na Figura 1 a capacidade limite do banheiro foi definida como sendo 5 e a quantidade de pessoas que iram utilizar o banheiro sendo 20.

Figura 1 – Recorte da saída do programa.

```
Pessoa 89(F) quer entrar no banheiro.
Pessoa 90(F) quer entrar no banheiro.
Pessoa 91(F) quer entrar no banheiro.
Pessoa 7(F) tentando sair do banheiro.
Pessoa 7(F) saiu do banheiro. Qtd no banheiro: 4
Pessoa 4(F) tentando sair do banheiro.
Pessoa 4(F) saiu do banheiro. Qtd no banheiro: 3
Pessoa 92(F) quer entrar no banheiro.
Pessoa 92(F) começou a usar o banheiro. Qtd de pessoas no banheiro: 4
Pessoa 96(M) quer entrar no banheiro.
Pessoa 99(F) quer entrar no banheiro.
Pessoa 99(F) começou a usar o banheiro. Qtd de pessoas no banheiro: 5
Pessoa 6(F) tentando sair do banheiro.
Pessoa 6(F) saiu do banheiro. Qtd no banheiro: 4
Pessoa 98(M) quer entrar no banheiro.
Pessoa 97(M) quer entrar no banheiro.
Pessoa 95(M) quer entrar no banheiro.
Pessoa 93(M) quer entrar no banheiro.
Pessoa 94(M) quer entrar no banheiro.
Pessoa 13(F) começou a usar o banheiro. Qtd de pessoas no banheiro: 5
Pessoa 13(F) tentando sair do banheiro.
Pessoa 13(F) saiu do banheiro. Qtd no banheiro: 4
Pessoa 12(F) começou a usar o banheiro. Qtd de pessoas no banheiro: 5
Pessoa 0(F) tentando sair do banheiro.
Pessoa 0(F) saiu do banheiro. Qtd no banheiro: 4
Pessoa 10(F) começou a usar o banheiro. Qtd de pessoas no banheiro: 5
Pessoa 99(F) tentando sair do banheiro.
Pessoa 99(F) saiu do banheiro. Qtd no banheiro: 4
Pessoa 10(F) tentando sair do banheiro.
Pessoa 10(F) saiu do banheiro. Qtd no banheiro: 3
Pessoa 12(F) tentando sair do banheiro.
Pessoa 12(F) saiu do banheiro. Qtd no banheiro: 2
Pessoa 5(F) tentando sair do banheiro.
Pessoa 5(F) saiu do banheiro. Qtd no banheiro: 1
Pessoa 92(F) tentando sair do banheiro.
Pessoa 92(F) saiu do banheiro. Qtd no banheiro: 0
Pessoa 17(M) começou a usar o banheiro. Qtd de pessoas no banheiro: 1
Pessoa 14(M) começou a usar o banheiro. Qtd de pessoas no banheiro: 2
Pessoa 17(M) tentando sair do banheiro.
Pessoa 17(M) saiu do banheiro. Qtd no banheiro: 1
Pessoa 24(M) começou a usar o banheiro. Qtd de pessoas no banheiro: 2
```

Fonte: Autoria própria

4 CONCLUSÃO

De maneira geral, acredita-se que foi implemetada uma solução de qualidade para o problema proposto, que funciona corretamente e eficientemente, que foi testada inúmeras vezes, documentada e com tratamento de eventuais exceções. Além disso, a solução abrange a corretude com relação a concorrência e foram aplicados os conceitos e mecanismos de sicronização, evitando, portanto, a ocorrência de condições de deadlock ou starvation e realizando exclusão mútua de forma apropriada.