

O PROBLEMA DA MONTANHA RUSSA

Projeto Multithread

MC504 - Sistemas Operacionais

César Devens Grazioti - RA: 195641

João Miguel De Oliveira Guimarães - RA: 174358

Otávio Anovazzi - RA: 186331

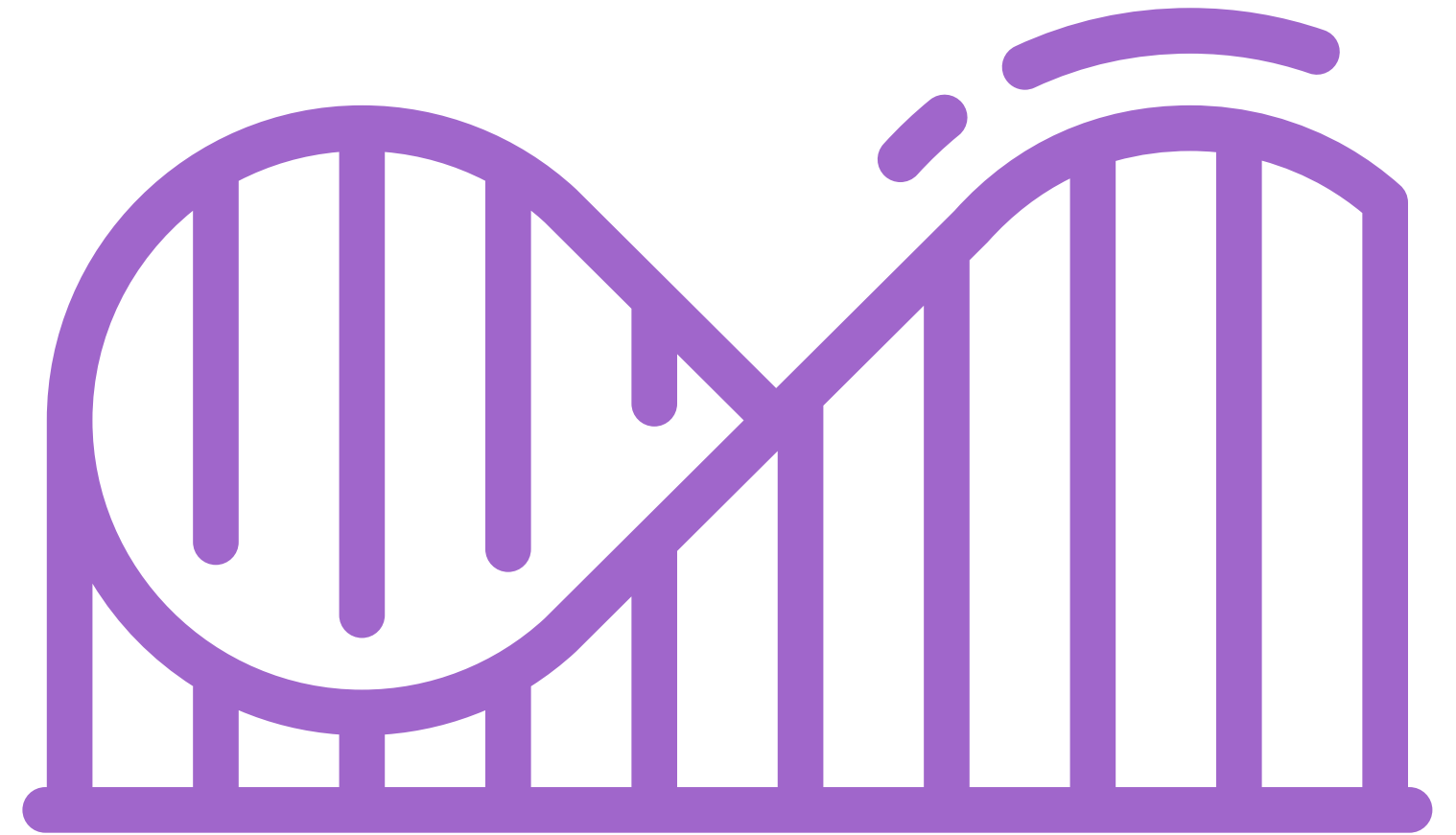
Renan Luis Moraes De Sousa - RA: 243792



Universidade Estadual de Campinas

Descrição do Problema

Suponha que haja N passageiros e um trem de montanha-russa. Os passageiros esperam repetidamente para passear na montanha-russa. O trem tem capacidade para C lugares, $C < N$, mas o parque não deixa o trem sair da estação até que esteja cheio. Além disso, a montanha russa dá no máximo K voltas em um dia.



Detalhes Adicionais



Passageiros devem solicitar embarque e desembarque.



O trem pode sinalizar três diferentes estados: embarque, andando ou desembarque.



Os passageiros não podem embarcar enquanto o trem sinalizar que o embarque está disponível.



O trem não pode partir até que os C passageiros tenham embarcado.



Os passageiros não podem desembarcar até que o trem inicie o período de desembarque.

Decisões do Grupo

Algumas decisões tomadas para a resolução do problema.



Animação

Optamos pelo estilo ASCII mas sem usar bibliotecas gráficas externas (ex: NCurses) para evitar dependências

Renderização

Implementamos nosso próprio sistema de renderização utilizando um sistema de buffer, com funcionalidades de interesse

Criatividade

A versão original do problema era pouco especificada, então adicionamos um contexto: funcionamento do parque



Implementação

Para a solução do problema, utilizamos N threads para os passageiros e uma thread para o carro. Definimos também um número N_RUNS de voltas que o carro executa.

Semáforos

Mutex locks

Variáveis de condição

◆ **mutex**

semáforo utilizado como mutex lock que protege a variável `boarders` para que somente 1 passageiro embarque no carinho por vez.

◆ **mutex2**

semáforo utilizado como mutex lock que protege a variável `unboarders` para que somente 1 passageiro desembarque no carinho por vez.

◆ **mutexPrint**

semáforo utilizado como mutex lock que protege a trava a execução durante a animação do programa.

◆ **boarders**

variável utilizada para armazenar o número de passageiros que embarcou no trem

◆ **unboarders**

variável utilizada para armazenar o número de passageiros que desembarcou no trem

◆ **boardQueue**

semáforo utilizado para solicitar a entrada na montanha russa. Se comporta como fila.

◆ **unboardQueue**

semáforo utilizado para solicitar a saída da montanha russa. Se comporta como fila.

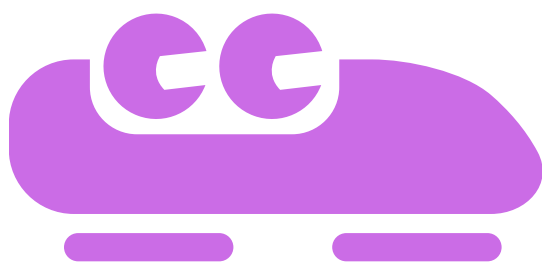
◆ **allAboard**

semáforo que indica quando o trem está completo com `C` passageiros.

◆ **allAshore**

semáforo que indica quando todos os `C` passageiros desembarcaram do trem

Thread do carro



Loop até que o número de voltas que o trem seja igual a `N_RUNS`;



Posta que o embarque iniciou e espera até que todos os `C` passageiros estejam a bordo;



Roda uma animação da montanha russa;



Sinaliza que o desembarque iniciou;



Espera que todos os passageiros estejam em terra e o processo se reinicia.

Threads dos passageiros



Passageiros esperam que o carro poste o embarque e que sua vez seja atingida (mutex);



Ao embarcar, é removido da fila e mostrada a animação;



Último passageiro a embarcar sinaliza que o carro está completo com C passageiros;

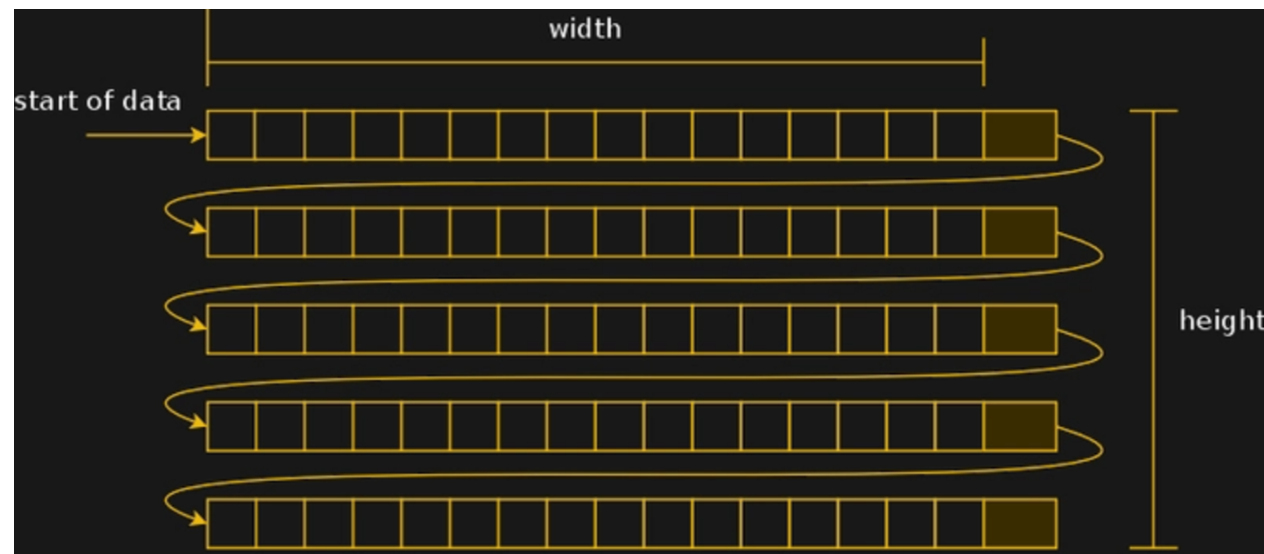
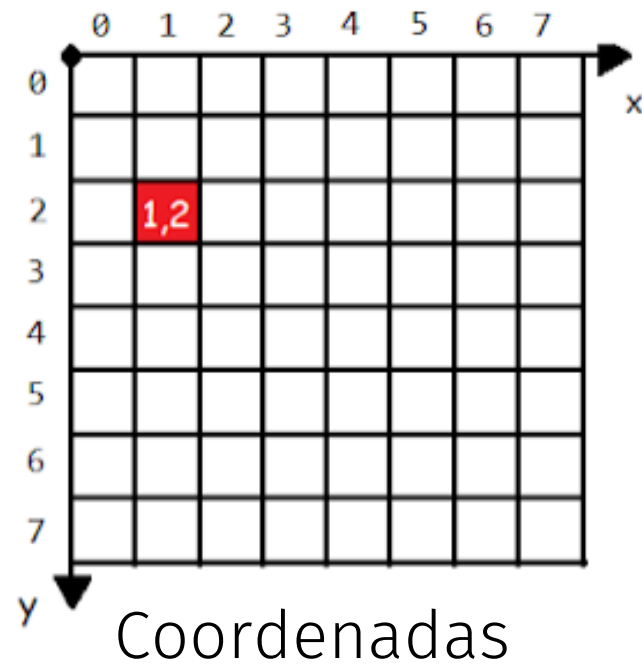


Espera a liberação de desembarque e sua vez do desembarque ocorra;



Último passageiro a desembarcar sinaliza que o carro está totalmente vazio e pronto para um novo passeio!

Biblioteca



Representação da tela em memória

Possui um buffer (1d) de chars que armazena o estado da tela (2d).

Efetua o "swap" do buffer : Renderiza as informações em memória.

Por ser custoso, "swap" ocorre apenas quando requerido.

Fornece funcionalidades para geração de objetos a partir de textos ASCII - Garantindo flexibilidade.

Sistema de coordenadas permite a renderização em qualquer coordenada da tela.

Animação



Montanha-russa com pessoas esperando para entrar e passeando quando o carrinho está cheio.



Cada carrinho possui um passageiro e uma posição.



Cada thread pessoa é responsável por atualizar seu estado no buffer.



Cada thread pessoa é responsável por atualizar seu estado no buffer.

Parametros da animação



Capacidade do carrinho



Número de pessoas



Resolução da tela



Número de voltas da montanha


Exemplo de Customização

 Capacidade da montanha russa: 2



 Número de pessoas: 5



 Resolução da tela: 120x40



 Número de voltas da montanha : 3




Exemplo de Customização

 Capacidade da montanha russa: 12



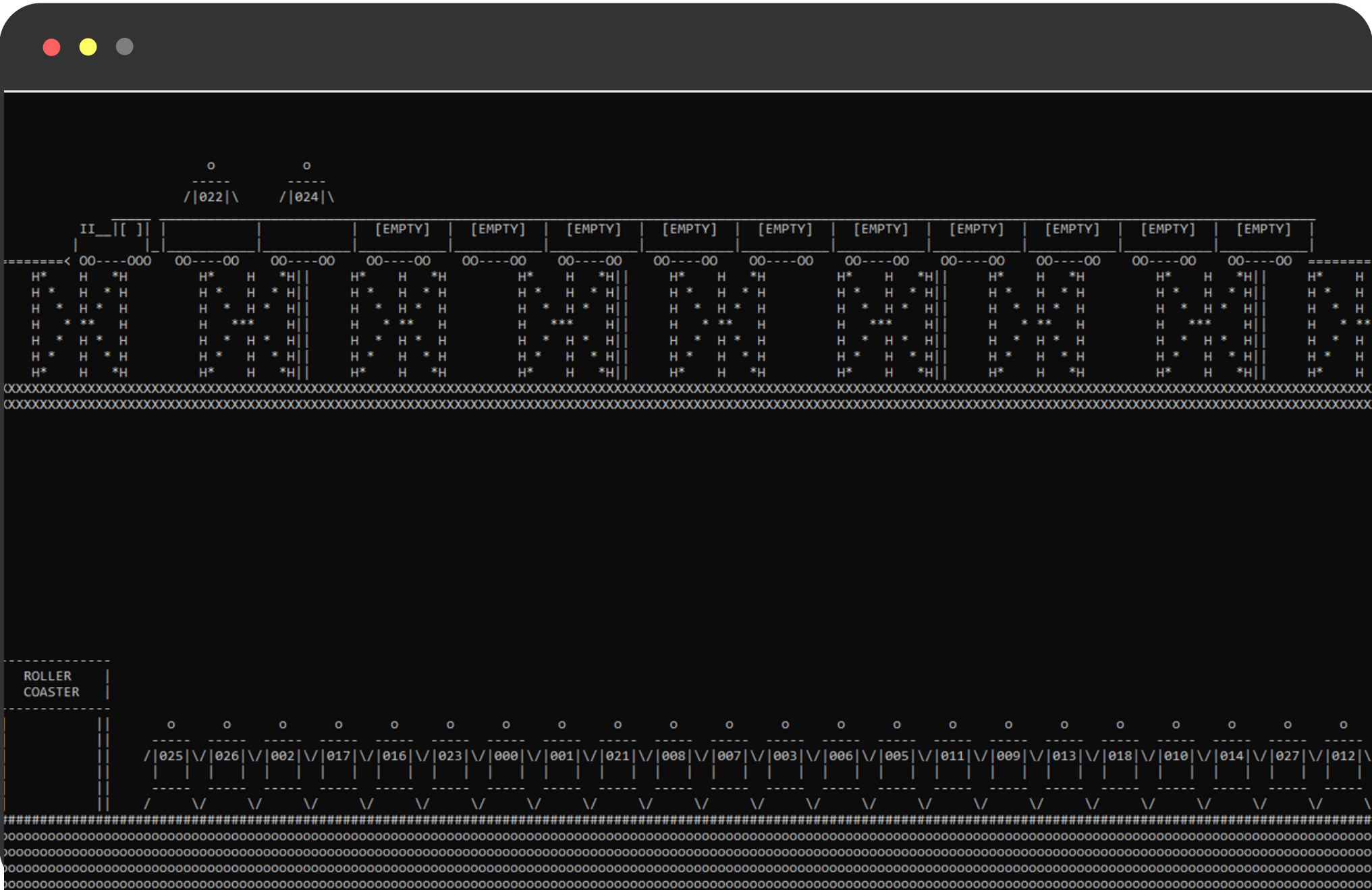
 Número de pessoas: 28



 Resolução da tela: 280x50



 Número de voltas da montanha : 2



Membros do grupo



João Miguel



César



Renan



Otavio

Obrigado pela atenção
