Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTAMENTO DE / DEPARTMENT OF
INFORMÁTICA E SISTEMAS

Licenciatura de Engenharia Informática

Simulador de Viagens no Deserto

Autores / Authors

Diogo Marques Silva **a2023139070**

André Pessoa Tavares **a2023155012**

Instituto Superior
de Engenharia
Politécnico de Coimbra

Coimbra, dezembro de 2024

ÍNDICE

1. INTRODUÇÃO

Este relatório apresenta o desenvolvimento e a implementação de um simulador baseado no enunciado do trabalho prático de Programação Orientada a Objetos (POO). O objetivo principal foi criar um ambiente de simulação dinâmica, no qual caravanas, cidades, bárbaros e itens interagem entre si num mapa predefinido. Durante o desenvolvimento, adotaram-se princípios da programação orientada a objetos, como encapsulamento, herança e polimorfismo, garantindo uma solução modular e extensível.

O simulador inclui funcionalidades como a movimentação automática e manual de caravanas, a interação com itens no mapa, combates entre caravanas e bárbaros, e um sistema dinâmico para a criação e eliminação de entidades. Destaca-se também a adição de uma caravana especial, o Pai Natal, que distribui presentes de forma programada como a caravana Secreta proposta.

Este documento detalha o planeamento e a estruturação do projeto executado.

2. ESTRUTURAÇÃO DO CÓDIGO

O simulador foi desenvolvido com uma abordagem centrada na Programação Orientada a Objetos (POO), garantindo uma organização modular, escalável e de fácil manutenção. A estrutura do código reflete uma separação clara de responsabilidades entre os vários componentes, permitindo que as funcionalidades estejam bem distribuídas e que a interação entre as entidades do simulador seja coesa e lógica.

O design baseia-se numa hierarquia de classes bem definida, onde cada entidade do simulador – como cidades, caravanas, bárbaros e itens – desempenha um papel específico, com métodos e atributos próprios. Esta abordagem assegura uma implementação coesa e flexível, permitindo interações claras entre os elementos do simulador.

Os principais aspetos da estruturação incluem:

- **Herança**: as classes partilham características comuns através de hierarquias bem definidas. Por exemplo, todas as caravanas derivam de uma class base, permitindo a criação de novos tipos de caravana com relativa facilidade e consistência.
- **Encapsulamento**: cada class controla os seus próprios dados e funcionalidades, protegendo a lógica interna de alterações indevidas e garantindo uma implementação robusta.

- **Polimorfismo**: métodos genéricos são capazes de lidar com diferentes tipos de objetos derivados. Por exemplo, as caravanas de comércio, militares e Pai Natal têm comportamentos distintos, mas são tratadas de forma uniforme em certas operações.
- **Separação de responsabilidades**: cada class tem um propósito bem definido. A class Simulador é responsável por gerir a lógica e o fluxo do programa, enquanto outras classes, como Buffer, são responsáveis por componentes específicos, como o estado do mapa.

3. CLASSES BASE E HERANÇA

3.1 Entidade

A class Entidade representa uma entidade genérica no simulador. Esta class serve como base para outras classes derivadas, fornecendo funcionalidades comuns como a gestão de posição e representação visual no mapa.

Funcionalidades Principais:

- Construtor: Inicializa uma entidade com um nome, posição (linha e coluna) e símbolo.
- setPosicao: Define a nova posição da entidade no mapa.
- atualizarEstado: Atualiza o estado da entidade (método virtual puro).
- imprimirStatus: Imprime os detalhes da entidade (método virtual puro).

3.2 Caravana (derivada de Entidade)

A class Caravana representa uma caravana no simulador. Esta class herda da class Entidade e adiciona funcionalidades específicas para a gestão de caravanas, como movimento, consumo de recursos, carga e descarga de mercadorias, e atualização do estado.

Funcionalidades Principais:

- Construtor: Inicializa uma caravana com um identificador, nome, posição (linha e coluna), símbolo, capacidade de carga, capacidade de água, consumo de água por instante e número de tripulantes.
- mover: Move a caravana no mapa.
- consumirRecursos: Consome água e verifica o estado da tripulação.

- reabastecerAgua: Reabastece a água ao máximo.
- imprimirStatus: Exibe informações da caravana.
- num_combate: Retorna a probabilidade para ganhar o combate consoante o número de tripulantes da caravana.
- alteraTripulantes: Altera o número de tripulantes.

3.2.1 Comercio

A class Comercio representa uma caravana de comércio no simulador. Esta class herda da class Caravana e adiciona funcionalidades específicas para a gestão de caravanas de comércio, como movimento, venda de mercadorias, e lidar com tempestades de areia.

Funcionalidades Principais:

- Construtor: Inicializa uma caravana de comércio com um identificador, nome, posição (linha e coluna), símbolo, capacidade de carga, capacidade de água, consumo de água por instante e número de tripulantes.
- mover: Move a caravana de comércio no mapa, alterando a sua posição (linha e coluna).
- lidarComTempestade: Lida com eventos de tempestade de areia, com uma chance de destruição e perda de carga.
- consumirRecursos: Consome água a cada instante, ajustando o consumo com base no número de tripulantes e verificando se a caravana fica sem água. Gasta 2 litros de água por instante, ou 1 se tiver menos de metade dos seus tripulantes, 0 se não tiver nenhum tripulante. Se ficar sem tripulantes move-se de forma aleatória e passados 5 instantes a caravana desaparece.
- imprimirStatus: Exibe informações detalhadas sobre a caravana, incluindo localização, carga, água, tripulantes e modo de operação.

3.2.2 Militar

A class Militar representa uma caravana militar no simulador. Esta class herda da class Caravana e adiciona funcionalidades específicas para a gestão de caravanas militares, como movimento, combate e lidar com tempestades de areia.

- Conhece: Caravana, Utilizador.

Funcionalidades Principais:

- Construtor: Inicializa uma caravana militar com um ID, nome, posição (linha e coluna), símbolo, capacidade de água, consumo de água por instante e número de tripulantes.
- lidarComTempestade: Lida com eventos de tempestade, perdendo 10% dos tripulantes e com 33% de chance de ser destruída.
- getAndarNaMesmaDirecao: Retorna se a caravana militar deve andar na mesma direção.
- consumirRecursos: Gasta 3 litros de água por instante, 1 litro apenas se tiver menos que metade dos tripulantes (inclusive se não tiver tripulante nenhum). Se ficar sem tripulantes, a caravana desloca-se sempre na mesma direção do último movimento que fez quando tinha tripulantes. Mas, passados 7 instantes, acaba por desaparecer.

3.2.3 PaiNatal

A class PaiNatal representa uma caravana especial no simulador que espalha presentes. Esta class herda da class Caravana e adiciona funcionalidades específicas para a gestão da caravana do Pai Natal, como espalhar presentes. Esta caravana espalha Itens de x em x turnos pelo mapa. Cada presente custa x mercadoria. Esta Caravana só dura x turnos e por isso não consome recursos. Anda sempre em modo automático. Esta class não é dona de nenhuma outra e tem apenas conhecimento da class Caravana.

Funcionalidades Principais:

- Construtor: Inicializa uma caravana do Pai Natal com um ID, nome, posição (linha e coluna), símbolo, capacidade de carga, capacidade de água, consumo de água por instante, número de tripulantes e intervalo de drop.
- espalharPresentes: Espalha presentes.

3.2.4 Barbara

A class Barbara representa uma caravana bárbara no simulador. Esta class herda da class Caravana e adiciona funcionalidades específicas para as caravanas bárbaras, como movimento automático, combate, resistência a tempestades de areia. Esta class não é dona de nenhuma outra e tem apenas conhecimento da class Caravana. Anda sempre em modo automático.

Funcionalidades Principais:

- Construtor: Inicializa uma caravana bárbara com um número máximo de instantes, nome, posição (linha e coluna) e número de tripulantes.
- combater: Realiza um combate com uma caravana do jogador, destruindo-a se tiver mais tripulantes e roubando as suas mercadorias.
- lidarComTempestade: Garante a resistência da caravana bárbara a tempestades de areia, com uma chance de destruição.

3.3 Item

A class Item representa um item genérico no simulador. Esta class herda da class Entidade e adiciona funcionalidades específicas para a gestão de itens, como a aplicação de efeitos em caravanas e utilizadores, e a gestão do tempo de existência no mapa.

- Conhece: Caravana, Utilizador. (Todas as classes que herdam de Item têm o mesmo conhecimento)

Funcionalidades Principais:

- Construtor: Inicializa um item com um tipo, nome, posição (linha e coluna), símbolo e tempo de existência.
- aplicarEfeito: Aplica o efeito do item na caravana e no utilizador (método virtual puro).
- reduzirTempoExistencia: Reduz o tempo de existência do item.
- estaAtivo: Verifica se o item ainda está ativo.

3.3.1 CaixaDePandora

A class CaixaDePandora representa um item específico no simulador que reduz 20% da tripulação de uma caravana. Herda de Item.

Funcionalidades Principais:

- Construtor: Inicializa uma Caixa de Pandora com um nome, posição (linha e coluna), símbolo e tempo de existência.
- aplicarEfeito: Aplica o efeito da Caixa de Pandora na caravana, reduzindo 20% da tripulação.
- imprimirStatus: Imprime os detalhes da Caixa de Pandora.

3.3.2 ArcaDoTesouro

A class ArcaDoTesouro representa um item específico no simulador que aumenta 10% das moedas do utilizador. Herda de Item.

Funcionalidades Principais:

- Construtor: Inicializa uma Arca do Tesouro com um nome, posição (linha e coluna), símbolo e tempo de existência.
- aplicarEfeito: Aplica o efeito da Arca do Tesouro no utilizador, aumentando 10% das moedas.
- imprimirStatus: Imprime os detalhes da Arca do Tesouro.

3.3.3 Jaula

A class Jaula representa um item específico no simulador que adiciona 10 prisioneiros à tripulação de uma caravana. Herda de Item.

Funcionalidades Principais:

- Construtor: Inicializa uma Jaula com um nome, posição (linha e coluna), símbolo e tempo de existência.
- aplicarEfeito: Aplica o efeito da Jaula na caravana, adicionando 10 prisioneiros à tripulação.
- imprimirStatus: Imprime os detalhes da Jaula.

3.3.4 Mina

A class Mina representa um item específico no simulador que destrói uma caravana. Herda de Item.

Funcionalidades Principais:

- Construtor: Inicializa uma Mina com um nome, posição (linha e coluna), símbolo e tempo de existência.
- aplicarEfeito: Aplica o efeito da Mina na caravana, destruindo-a.
- imprimirStatus: Imprime os detalhes da Mina.

3.3.5 Surpresa

A class Surpresa representa um item específico no simulador que restabelece a água da caravana e aumenta as moedas do utilizador (Lagoa Dourada). Herda de Item.

Funcionalidades Principais:

- Construtor: Inicializa uma Surpresa com um nome, posição (linha e coluna), símbolo e tempo de existência.
- aplicarEfeito: Aplica o efeito da Surpresa na caravana e no utilizador, restabelecendo a água da caravana e aumentando as moedas do utilizador na mesma quantidade de água recebida. Existe 40% de probabilidade da caravana encontrar uma pedra preciosa que aumenta a quantidade de moedas do utilizador para o dobro.
- imprimirStatus: Imprime os detalhes da Surpresa.

3.4 Cidade (derivada de Entidade)

A class Cidade representa uma cidade no simulador. Esta class herda da class Entidade e adiciona funcionalidades específicas para a gestão de cidades, como a compra e venda de mercadorias, contratação de tripulantes, e gestão de caravanas disponíveis para venda.

- Conhece: Caravana, Utilizador, Comercio, Militar, PaiNatal
- É dona de: Caravana (através de shared_ptr) para gerir as caravanas disponíveis para venda.

Funcionalidades Principais:

- Construtor: Inicializa uma cidade com um nome, letra representativa, preços de venda e compra de mercadorias, preço de caravanas, e posição (linha e coluna). São criadas 3 caravanas dos 3 tipos que tem conhecimento para venda.
- entrarCaravana: Adiciona uma caravana à lista de caravanas presentes na cidade.
- saidaCaravana: Remove uma caravana da lista de caravanas presentes na cidade.
- existeCaravana: Verifica se uma caravana está presente na cidade.
- inspecionar: Exibe informações sobre as caravanas presentes na cidade.
- comprarMercadoria: Permite a uma caravana comprar mercadorias na cidade.
- venderMercadoria: Permite a uma caravana vender mercadorias na cidade.
- contratarTripulantes: Permite a uma caravana contratar tripulantes na cidade.
- comprarCaravana: Permite a um utilizador comprar uma nova caravana na cidade.

3.5 Buffer

A class Buffer gere uma grelha 2D de caracteres, representando o mapa ou área de exibição no simulador. Fornece funcionalidades para manipular e exibir a grelha, incluindo escrever caracteres e strings, mover um cursor e limpar o buffer. É dona de uma matriz dinâmica e não herda de nenhuma outra class.

Funcionalidades Principais:

- Construtor: Inicializa o buffer com dimensões especificadas e aloca memória para a grelha de caracteres.
- Construtor por Cópia: Cria uma cópia de um buffer existente.
- Destrutor: Liberta a memória alocada para a grelha de caracteres.
- escrever: Escreve um caractere, string ou inteiro na posição atual do cursor.
- imprimir: Imprime o conteúdo do buffer na consola.
- getCaracter: Obtém um caractere de uma posição especificada no buffer.
- setCaracter: Define um caractere numa posição especificada no buffer.

- Operadores Sobrecarga: Sobrecarga do operador << para escrever caracteres, strings e inteiros no buffer.
- Operador de Atribuição: Atribui o conteúdo de um buffer a outro.

3.6 Parâmetros

A class Parametros representa os parâmetros configuráveis do simulador. Esta class permite carregar os parâmetros a partir de um ficheiro e fornece métodos de acesso para obter os valores dos parâmetros.

- Conhece: Buffer, Exceções

Funcionalidades Principais:

- Construtor: Inicializa os parâmetros com valores padrão.
- carregarDeFicheiro: Carrega os parâmetros a partir de um ficheiro.
- mostrarConteudo: Mostra o conteúdo dos parâmetros carregados.

Métodos de Acesso

- getLinhas: Retorna o número de linhas do mapa.
- getColunas: Retorna o número de colunas do mapa.
- getMoedas: Retorna o número de moedas.
- getInstantesEntreNovosItens: Retorna os instantes entre novos itens.
- getDuracaoItem: Retorna a duração do item.
- getMaxItens: Retorna o número máximo de itens.
- getPrecoVenda: Retorna o preço de venda da mercadoria.
- getPrecoCompra: Retorna o preço de compra da mercadoria.
- getPrecoCaravana: Retorna o preço da caravana.
- getInstantesEntreNovosBarbaros: Retorna os instantes entre novos bárbaros.
- getDuracaoBarbaros: Retorna a duração dos bárbaros.

3.7 Utilizador

A class Utilizador representa o jogador no simulador. Esta class gere as moedas, turnos, caravanas controladas e combates vencidos pelo utilizador.

Funcionalidades Principais:

- Construtor: Inicializa o utilizador com um número inicial de moedas.
- adicionarMoedas: Adiciona uma quantidade de moedas ao total do utilizador.
- removerMoedas: Remove uma quantidade de moedas do total do utilizador.
- incrementarTurno: Incrementa o número de turnos jogados.
- registrarCombateVencido: Regista um combate vencido pelo utilizador.
- exibirResumoSimulacao: Exibe um resumo da simulação.

3.8 Exceções

Existem duas classes para dois tipos de Exceções diferentes:

- A class **SimuladorExcecao** é a class base para todas as exceções no simulador. Herda de runtime_error e permite a criação de mensagens de erro personalizadas. A class representa exceções relacionadas a erros de simulação no simulador.
- A class **ConfiguracaoExcecao** representa exceções relacionadas a erros de configuração no simulador. Herda de SimuladorExcecao.

3.9 Simulador

A class Simulador gere a execução do simulador, incluindo a configuração, execução de comandos, movimentação de caravanas, gestão de itens e bárbaros, e atualização do estado do simulador.

- Conhece: Parametros, Buffer, Comercio, Militar, Utilizador, Cidade, Item, Barbara, Caravana, PaiNatal.
- É dona e gere as seguintes classes:
    - Parametros
    - Buffer
    - Utilizador
    - Caravana (e suas classes derivadas Comercio, Militar, PaiNatal)
    - Cidade
    - Item
    - Barbara

Funcionalidades Principais:

- Construtor: Inicializa o simulador com valores padrão.
- comandoConfig: Carrega a configuração a partir de um ficheiro.
- comandoSair: Termina a execução do simulador.
- comandoExec: Executa comandos a partir de um ficheiro.
- comandoMoedas: Ajusta o número de moedas do utilizador.
- comandoTerminar: Termina a simulação e reinicia as variáveis.
- comandoCompraC: Compra uma caravana duma cidade.
- comandoPrecos: Lista os preços de compra, venda e caravana.
- comandoCidade: Lista informações de uma cidade.
- comandoCaravana: Lista informações de uma caravana.
- comandoCompra: Compra mercadorias para uma caravana.
- comandoVende: Vende mercadorias de uma caravana.
- comandoMove: Move uma caravana para uma nova posição.
- comandoAuto: Ativa o modo automático de uma caravana.
- comandoStop: Desativa o modo automático de uma caravana.

- comandoBarbaro: Cria um bárbaro numa posição específica.
- comandoAreia: Aplica uma tempestade de areia numa área.
- comandoTripul: Contrata tripulantes para uma caravana.
- comandoSaves: Guarda o estado atual do buffer.
- comandoLoads: Carrega um estado guardado do buffer.
- comandoLists: Lista os estados guardados.
- comandoDels: Elimina um estado guardado.
- processarComando: Processa um comando.
- estaEmExecucao: Verifica se o simulador está em execução.
- atualizarEstado: Atualiza o estado do simulador.
- inicializarObjetos: Inicializa os objetos do simulador e cria as entidades que estão no buffer.
- mostrarEstado: Mostra o estado do simulador.
- executarTurno: Executa um turno.
- moverParaPosicaoAdjacenteAleatoria: Move uma caravana para uma posição adjacente aleatória.
- gerirCaravanasBarbaras: Gere as caravanas bárbaras.
- gerirItens: Gere os itens.
- gerirCaravanas: Gere as caravanas.
- comportamentoCaravanaComercio: Define o comportamento de uma caravana de comércio.
- comportamentoCaravanaMilitar: Define o comportamento de uma caravana militar.
- comportamentoCaravanaPaiNatal: Define o comportamento de uma caravana do Pai Natal.
- combate: Realiza um combate entre uma caravana e um bárbaro.
- encontrarCaravanaPorSimbolo: Encontra uma caravana pelo símbolo.
- eliminarCaravana: Elimina uma caravana.
- atualizarEntidades: Atualiza as entidades do simulador após um novo buffer ser carregado.

4. JUSTIFICAÇÃO DAS COLEÇÕES E SMART POINTERS

4.1 Coleções Utilizadas

4.1.1 std::map:

- Uso: O **std::map** é utilizado para armazenar cópias do buffer identificadas por nome (copiasBuffer) e para contar movimentos das caravanas (moveCounter).
- Escolha: O **std::map** é escolhido porque permite acesso rápido aos elementos através de chaves únicas, garantindo que cada nome de cópia ou ID de caravana seja único e facilmente acessível.

4.1.2 std::vector

- Uso: O **std::vector** é utilizado para armazenar listas de caravanas (caravanas), cidades (cidades), itens (itens) e bárbaros (barbaros).
- Escolha: O **std::vector** é escolhido pela sua eficiência em termos de acesso sequencial e pela capacidade de redimensionamento dinâmico, o que é útil para armazenar coleções de objetos que podem crescer ou diminuir durante a execução do simulador.

4.1.3 std::set

- Uso: O **std::set** é utilizado para armazenar os IDs das caravanas encontradas no buffer durante a atualização das entidades (idsEncontrados).
- Escolha: O **std::set** é escolhido porque garante que os elementos são únicos e permite operações eficientes de inserção, remoção e verificação de existência, o que é essencial para garantir que não há duplicação de IDs durante a atualização.

4.2 Smart Pointers

4.2.1 std::shared_ptr

- Uso: O std::shared_ptr é utilizado para gerir a memória de objetos como caravanas, cidades, itens e bárbaros.
- Escolha: O std::shared_ptr é escolhido porque permite a partilha de propriedade dos objetos entre diferentes partes do código. Vários shared_ptr podem apontar para o mesmo objeto, e o objeto é automaticamente destruído quando o último shared_ptr que o referencia é destruído. Isto é particularmente útil no simulador, onde múltiplas entidades podem referenciar o mesmo objeto, garantindo uma gestão de memória segura e evitando fugas de memória.

5. CONCLUSÃO

No geral, o projeto demonstra uma abordagem bem estruturada e eficaz para a construção de um sistema de simulação funcional, aproveitando as capacidades do C++ de forma prática e eficiente. O uso de ponteiros inteligentes (smart pointers) para gestão de memória garante segurança e minimiza problemas como fugas de memória, enquanto o recurso a coleções como vetores e mapas assegura um desempenho sólido no armazenamento e manipulação de dados.

O sistema de processamento de comandos destaca-se pela sua flexibilidade, permitindo ao utilizador interagir e controlar a simulação de forma direta e funcional. A clara separação de responsabilidades entre as várias classes contribui para a organização do código, tornando-o mais compreensível e fácil de manter ou expandir.

Este projeto apresenta uma aplicação prática e eficaz de princípios de programação orientada a objetos, demonstrando um equilíbrio adequado entre complexidade técnica e simplicidade funcional.

Instituto Superior
de Engenharia
Politécnico de Coimbra