

Universidade Católica de Angola

Faculdade de Engenharia

Departamento de Informática

JOGO SOLITÁRIO

IMPLEMENTAÇÃO COM ESTRUTURAS DE DADOS

Docente

Solander Agostinho

Luanda aos

Relatório do projeto de Fundamentos de Programação III

Sala: 208

Período: Tarde

Ano: 2º

Curso: Engenharia Informática

Disciplina: FP III

Integrantes do Grupo

Amarildo Miguel -

Denízia Cazequene - 21400

Eufrânio Diogo - 21099

Isaura Manico - 20507

Índice

Relatório do projeto de Fundamentos de Programação III	2
Integrantes do Grupo	3
Introdução	5
Objectivos	6
Objetivo geral	6
Objetivo específico	6
Explicação estruturas Utilizadas	6
O que é uma lista?	6
O que é uma pilha?	6
Regras e Funcionamento do Jogo Solitário	7
Estrutura do Projeto	8
Interface Gráfica	9
Movimentações	10
SolucaoBaralho	10
Tabuleiro	10
Conclusão	10

Introdução

Estrutura de dados, em um contexto geral, estudar diferentes técnicas e formas de armazenamento e organização de dados, levando em consideração suas características, requisitos de armazenamento e tipos de problema que se pretende resolver com os mesmos. No projeto presente, implementamos um jogo na linguagem Java, com o componente swing, basicamente java2D, para a criação da cena, e internamente, usamos as estruturas de dados Lista e Pilhas, para a construção e organização das cartas, o tal dito objeto.

Objectivos

Objetivo geral

Criação do jogo Solitário, usando estruturas de dados aprendidas na cadeira de.

Objetivo específico

Implementação do jogo com uma UI feita em java, para obter maior interação com o usuário.

Explicação estruturas Utilizadas

O que é uma lista?

A lista é uma estrutura de dados muito utilizada. Cujo armazena os dados em um formato de uma lista da vida real.

Tendo como as suas principais operações a inserção de novos elementos no início, fim e em qualquer outra posição, a sua remoção e também a possibilidade de saber quantos elementos existem e ver quais são.

Sendo que o elemento base para a implementação de uma lista é um ADT chamado Nó, cujo possui a função de armazenar dada informação e fazer a interconexão com outros nós da mesma lista.

O que é uma pilha?

É uma estrutura de dados que faz uso do algoritmo de movimentação **LIFO** basicamente significando que o último elemento a ser inserido será o primeiro a ser removido. Sendo assim possuímos basicamente 4 operações que são: inserção(que só pode ser no topo da pilha), remoção, ver o topo da pilha e retornar a o seu tamanho.

Regras e Funcionamento do Jogo Solitário

O Jogo solitário é um jogo do tipo single player(jogador único) cujo são dispostas inicialmente para o usuário 7 colunas com 28 cartas dispostas nelas e no total 7 abertas e as restantes fechadas um baralho auxiliar para a possível retirada de cartas para complementar nas combinações de cartas que o jogo exige, e sendo que a combinação de cartas que deve ser uma carta de inferior valor ficar por baixo de uma de superior valor e de uma cor diferente, sendo que ao total são 52 cartas, 26 vermelhas e 26 pretas, e o seu objectivo final é conseguir que todas as combinações de cartas sejam colocadas nos slots superiores em uma ordem crescente.

As jogadas de movimentação de cartas não se limitam a uma única carta. Você também pode mover grupos de cartas sequencialmente organizadas entre as colunas. Se não houver mais jogadas possíveis, o jogador deverá tirar as cartas do baralho auxiliar. Se o baralho acabar, o jogador deverá voltar ao seu estado inicial.

Estrutura do Projeto

O projeto contém dois pacotes que são: **Imagens** e **Main**. Sendo que o pacote **Imagens** contém as imagens das cartas e o pacote **Main** sendo responsável de armazenar as classes necessárias para implementação da lógica do jogo.

O pacote **Main** contém 11 classes sendo que cada uma tem as suas responsabilidades.

Classes:

- **Pilha**: Pilha para
- **ListaLigada**: Representação de uma lista que tem a função de armazenar uma lista de cartas.
- **BaralhoPrincipal**: Cartas restantes após a construção do jogo e que podem ser utilizadas para fazer movimentações.
- **BaralhoSecundario**: Representação lógica da pilha auxiliar quando as cartas são retiradas do baralho principal.
- **Carta**: Representação lógica de carta é capaz da geração de um conjunto de cartas de um determinado tipo.
- **PainelDoJogo**: Classe que tem a função de conter todos os elementos do jogo sendo ela filha de JPanel.
- **OuvidorDeEventos**: MouseEventListener ouvidor de eventos do mouse, classe que permite saber quais movimentos o usuário pretende realizar.
- **Tabuleiro**: Classe que tem a função de representar cada uma das 7 colunas que ficam as cartas.
- **SolucaoBaralho**: Classe que representa os slots superiores para a solução do jogo.
- **Main**: Classe principal.

Interface Gráfica



O nosso jogo solitário está estruturado de uma forma que cada um dos elementos gráficos aqui apresentados têm a sua própria representação lógica correspondente, sendo que o jogo foi dividido em 4 areas principais cujo cada possui as suas funcionalidades

- **Solução Baralho:** é um conjunto de pilhas cujo devem compor a solução do jogo sendo que devem seguir a sequência: A -> 2 -> 3 -> 4 -> Sendo que todas devem pertencer ao mesmo simbolo.
- **Baralho Secundario:** é uma pilha que serve como pilha de destino quando uma carta é removida do baralho principal.
- **Baralho Principal:** é uma pilha que tem o restante das cartas após elas serem distribuídas pelos tabuleiros e cartas que podem ajudar na solução do jogo.
- **Tabuleiro[]:** é um vector que contém 7 pilhas representando as 7 colunas do jogo que inicialmente elas começam com apenas uma carta no topo das pilhas.

Movimentações

Para efetuarmos as devidas movimentações do jogo, usamos o recurso **MouseEvents** disponibilizado pelo Java com os seus métodos **mousePressed** e **mouseReleased**, cujo o **mousePressed** é disparado quando o mouse é clicado e continuamente pressionado e o **mouseReleased** quando o mouse deixa de ser clicado considerando que quando o usuário deixa de clicar dessa forma disparando o evento **mouseReleased** ele acabou de nos informar qual o destino da carta(s) ele quer dar.

SolucaoBaralho

- `movimentarCartaDoBaralhoSecundarioParaSolucaoBaralho`
- `movimentaCartaDeSolucaoBaralhoParaTabuleiroDestino`

Tabuleiro

- `movimentaCartaDeBaralhoSecundarioParaTabuleiroDestino`
- `movimentarCartaDoTabuleiroParaSolucaoBaralho`
- `movimentaCartaDeTabuleiroOrigemParaTabuleiroDestino`

Com estes métodos foi possível implementarmos as movimentação requeridas no jogo actuando apenas nas classes principais do jogo sendo elas a **SolucaoBaralho** e a **Tabuleiro**.

Conclusão

Acabamos concluindo que com os conhecimentos de estrutura de dados, Java 2D e **MouseEvents** é possível a criação do jogo Solitário.