Trabalho Prático 2
Jogo de Baralho - Burro

Édipo Fernandes Vieira de Oliveira - 2011054324
Diego Henrique de Castro Aniceto - 2011054286
Departamento de Ciência da Computação – Universidade Federal de Minas Gerais

18 de maio de 2015

Resumo:
Este relatório descreve a implementação do Jogo de baralho Burro. Para que fosse possivel essa implemetação foi utilizado a linguagem de programação Java além de teorias de Orientação a Objeto, dentre elas Modularização, Encapsulamento, Herança, Polimorfismo, entre outras.
O resultado obtido foi satisfatório, tanto em relação a solução do problema, quanto aos conceitos envolvidos.

1. Introdução

Este trabalho tem como objetivo, observar na prática os principais conceitos da Programação Orientada ao Objetoo. Para que estes fossem exercitados foi proposto a implementação de um jogo de baralho. O jogo escolhido é chamado **Burro**, ele funciona da seguinte forma:

São necessários no mínimo duas pessoas para jogar, porém não contém um numero máximo de jogadores. O jogo se inicia com as pessoas recebendo quatro cartas cada, uma destas deverá jogar uma carta na mesa, as demais deverão jogar uma carta do mesmo naipe daquela que está na mesa, caso não tenham o naipe solicitado em mãos a pessoa deverá comprar cartas no monte de cartas ate que encontre o naipe necessário, quem jogar a maior carta será vencedor da jogada, e deverá iniciar a próxima rodada. O vencedor do será aquele que jogar todas as suas cartas da mão na mesa. O perdedor será aquele que ficar com cartas na mão por ultimo. o numero de cartas que estiver na mão dele serão os 'anos de burrice' da pessoa.

O jogo foi modelado utilizando como base as entidades principais do jogo, *Jogador* e *Baralho*, e a partir destas foram geradas as classes utilizadas para o desenvolvimento do jogo e que também serão descritas nas próximas sessões.

Para que a implementação deste trabalho fosse possível, foram utilizados conceitos de Orientação a Objeto, como herança, polimorfismo, restrição de acesso, entre outros, pois dessa forma o algoritmo cobre os requisitos básicos solicitados e também facilita a compreensão do trabalho.

O trabalho está composto das seguintes seções:

- A seção 2 discute detalhes de implementação e da modelagem do problema.

- A seção 3 traz os testes realizados para verificar a solução do trabalho, bem como a saída gerada
- A seção 4 apresenta uma breve conclusão sobre o trabalho.
- E por fim a seção 5 traz as referências bibliográficas.

2. Implementação

2.1. Modelagem do Jogo

O jogo foi modelado em torno de duas classes principais, *Jogador* e *Baralho*.

A classe *Jogador* é abstrata, logo contem métodos abstratos e concretos, ela também é a classe base de outras duas, *JogadorHumano* e *JogadorBot* que extendem da super classe. A classe *JogadorHumano* como o proprio nome diz, modela um jogador humano e de acordo com o conceito de herança, ela "É-UM"*Jogador*, assim ela só pode ser um jogador e a classe *JogadorBot* modela um jogador controlado pelo computador e segue o mesmo conceito da classe anterior.

A classe *Baralho* é uma classe concreta que é composta por *Cartas*, estas por sua vez são compostas de *Naipe* e *Valor*, essas duas ultimas, são enumerações, uma forma encontrada para facilitar o desenvolvimento e deixa-lo mais simplificado.

Essa modelagem pode ser visualizada de forma simples no Diagrama de Classes a seguir:

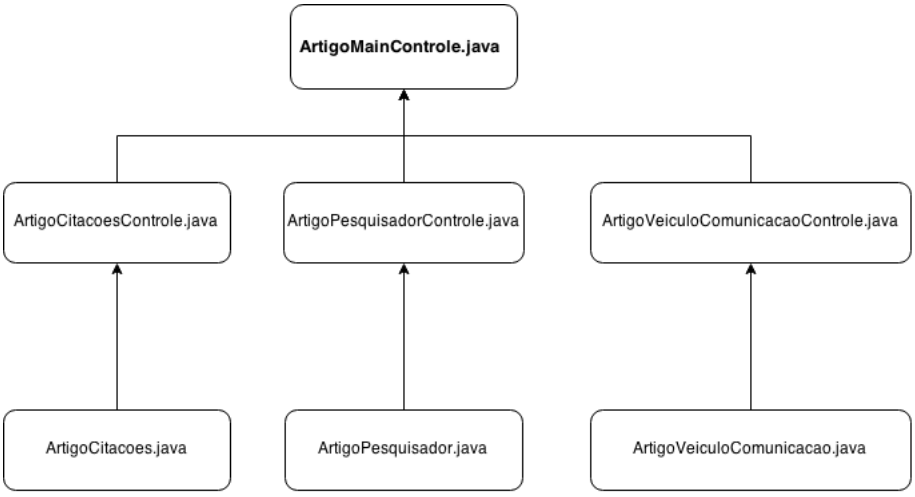

Figura 1: Estrutura de Dependência de Artigo

Como pode ser visto no diagrama acima, a classe **ArtigoMainControle.java** faz o encapsulamento das demais classes, fornecendo o conjunto de todas as operações necessárias envolvendo um Artigo, não sendo necessario assim nenhuma altaração nas classes bases.

A modularização das entidades Pesquisador e Veiculo de Comunicação foram feitas de forma similar a classe Artigo, porém de forma mais simples pois essas classes não tem dependências de demais classes, o que facilita o reuso e a manutenção destas.

2.2. Conceitos OO

Para a modularização e o encapsulamento das classes fossem possíveis, foi necessária a implementação dos conceitos de *Orientação a Objeto*.

O conceito utilizado para realizar o relacionamento entre as classes foi o de **Composição** onde temos uma classe que funciona como o *todo* e classes que funcionam como uma parte deste todo, por exemplo, a classe **PesquisadorControle** (responsável pela implementação dos metódos referentes a entidade Pesquisador) é o todo e a classe **Pesquisador** é parte da classe **PesquisadorControle**. Foi utilizado também o conceito de Objeto, que foi utilizado para que o acesso as classes dependentes fosse possível.

2.3. Calculos e Execução

O principal objetivo do trabalho era fornecer uma lista com a popularidade de cada pesquisador, o fator de impacto de cada veiculo de comunicação e a pontuação de cada artigo. Para resolver esses problemas, foi criada uma classe especifica que processa todos esses dados e gera os dados solicitados.

Esta classe é composta por objetos de todas as classes **Controle** que fornecem as operações de suas respectivas entidades, tornando possível o processamento dos dados e o fornecimento das respostas.

Para que a solução fosse executada, foi desenvolvida uma classe principal, **MainClass**, que contem um construtor e um objeto da classe **Resultado** que realiza os cálculos necessários pra solucionar o problema proposto.

3. Testes

A solução proposta resolve o problema proposto de forma eficaz e eficiente, gerando a saida esperada, com os dados de Popularidade do pesquisador, fator de inpacto dos veiculos de comunicação e a pontuação dos artigos de forma ordenada. Segue abaixo a saida gerada pelo programa

popularidade_pesquisador.txt

1;1356.6667
2;264.7500
3;357.8333
4;355.8333
5;639.2500
6;522.2500
7;778.8333
8;732.0000
9;377.5000
10;357.1667
11;405.0000
12;420.6667
13;595.4167
14;305.0000

15;366.0000

fatorImpacto_veiculo.txt

1;4.0161
2;4.2308
3;4.0862
4;3.9444

pontuacao_artigo.txt

1;15.7778
2;12.0484
3;11.8333
4;4.0862
5;8.1724
6;12.0484
7;24.5172
8;8.4615
9;23.6667
10;8.0323
11;4.2308
12;12.2586
13;7.8889
14;21.1538 15;16.3448 16;12.2586 17;24.0968 18;8.1724 19;7.8889 20;16.9231 21;20.4310 22;12.0484 23;12.0484 24;19.7222 25;7.8889 26;16.3448 27;8.1724 28;25.3846 29;8.0323 30;12.2586 31;12.0484 32;12.2586 33;3.9444 34;16.3448 35;0.0000 36;12.0484 37;8.0323 38;15.7778 39;4.2308 40;0.0000 41;21.1538 42;24.0968 43;11.8333

4. Conclusão

5. Referências bibliográficas