**Universidade Federal de Minas Gerais**

**Disciplina**: Progamação Modular

**Tarefa**: Jogo BlackJack

**Alunos**: Rodrigo Augusto Gontijo

Caio Godoy

**Introdução:**

Este trabalho consiste na implementação de um software que irá simular um jogo de cartas chamado BlackJack ,simulando, por exemplo: as retiradas de cartas do baralho, o embaralhamento, a exibição dos resultados, etc.

O programa tem um funcionamento simples para o usuário, ele irá entrar com os seguintes dados:

* Quantidade de jogadores
* Nome dos jogadores
* Retiradas de cartas

Feita essa entrada, o sistema irá processar os dados informados, e emitir uma saída, sendo ela:

* A pontuação dos jogadores
* Quem foi(ram) o(s) vencedor(es) ( caso haja ).

Nosso jogo possue o fluxo dividido em rodadas, sendo que cada rodada representa a jogada de um só jogador. Nessa rodada, o jogador tem a opção de comprar ou não mais cartas, caso o somatório de seus pontos passe 21 pontos, sua rodada termina, caso não, ele tem a opção de comprar mais cartas ou não. Outra opção que decidimos, é quando o jogador faz exatamente 21 pontos, ele não pode comprar mais cartas.

**Implementação:**

Para a implementação desse software,foi criado dois TAD´s , sendo um para o usuário e outro para as cartas:

O TAD de Usuário possue:

* Nome.
* Conjunto de cartas que ele tem em mãos.

O TAD de Carta possue:

* Um enum que nos diz qual tipo de naipe é a carta.
* Nome
* Uma variável que nos fala se essa carta possue valor múltiplo.
* Uma variável que nos diz o valor da carta , ou os valores caso ela seja de valor múltiplo.

E também pelas funções que iram dar entrada nesses dados e por funções que iram exibir esses dados( como por exemplo a função que exibe o "valor da mão" de um determinado usuário).

Foi criada uma interface para a classe de Card a fim de estabelecer um contrato entre essas duas classes e viabilizar a modularidade do programa.

O funcionamento do programa ocorre da seguinte maneira:

* Dentro da main, as instâncias são feitas, sendo que a instância do Deck é um Singleton ( optamos por implementar esse Singleton pois todos os usuários acessam o mesmo baralho) e as outras são coleções de objetos, como por exemplo a intância de coleções de usuários. Depois variáveis auxiliares são criadas também.
* Feita as inicializações, o progama lê do teclado a quantidade de jogadores que irão jogar o jogo, sendo de 2 a 4 jogadores.
* Recebido o número de jogadores, utilizamos o Design Patter Factory para facilitar a instanciação dos objetos do tipo Usuário, visto que necessitamos instanciar cada jogador separadamente.
* Depois dos passos acima, estamos prontos para começar o jogo. O baralho é embaralhado e as rodadas são realizadas.
* Depois das rodadas de todos jogadores acabarem, o resultado final do jogo é exibido na tela, sendo possível os jogadores empatarem ( ou nínguem ganhou ou o maior número de pontos for compartilhado entre dois jogadores) ou um só jogador ganhar.
* Por último, é perguntado ao usuário se ele deseja jogar novamente, caso não o programa encerra, caso sim o ciclo todo recomeça do início.

UML do sistema:



**Testes:**

Ffeitos alguns testes idênticos ( obviamente as saídas foram diferentes), todos funcionaram perfeitamente na implementação feita. Abaixo estão as SS de alguns dos testes realizados.





Teste com 3 jogadores





Teste com 4 jogadores





Teste com 2 jogadores

**Conclusão:**

O trabalho foi de suma importância para fixar os ensinamentos de padrão de projeto passados em sala. Ambos os integrantes possuem contato diário com desenvolvimento de software e é de vital importância a fixação desses conceitos.

Algumas dificuldades encontradas:

* Decisão entre qual padrão de projeto utilizar.
* Elaborar a melhor implementação possível para o trabalho, possibilitando, por exemplo: flexibilidade de código, modularização do projeto, fácil alteração futura, etc.

**Bibliografia:**

<http://www.devmedia.com.br/leitura-e-escrita-de-arquivos-de-texto-em-java/25529>

<http://pt.stackoverflow.com/questions/88345/passagem-de-par%c3%a2metro-usando-o-string-args-declarado-na-main>

<http://www.devmedia.com.br/convencoes-de-codigo-java/23871>