Universidade Federal de Minas Gerais
Departamento de Ciência da Computação

Trabalho Prático - Black Jack

Felipe Louzada Mingote.
Felipe da Cruz Rocha
Pedro Luis Costa Mucci
Vinicius Brenner Moreira da Silva

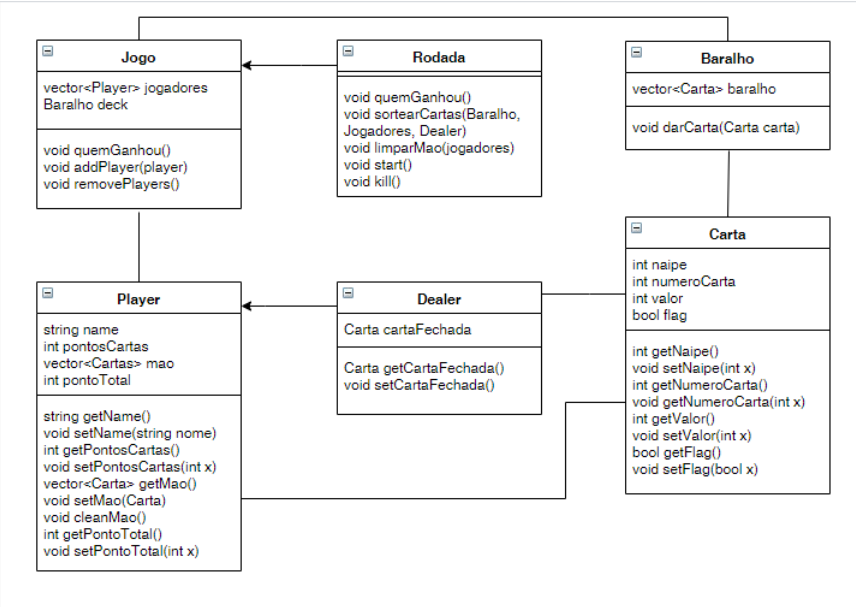
11 de Junho de 2019

1 Introdução

Este trabalho possui como objetivo a aplicação dos conhecimentos adquiridos na disciplina, sobre o paradigma de Orientação a Objeto. O tema escolhido foi Jogo de Cartas, e o jogo escolhido foi o Black Jack, conhecido informalmente como Vinte e Um.

2 Modelagem do Problema

Primeiramente, identificamos as entidades, que foram modeladas como objetos: jogadores, *dealer*, baralho e cartas. Também modelamos as rodadas e o jogo todo em si como objetos. Depois, mapeamos o fuxo do jogo em uma classe controladora, chamada Jogo.
As classes seguem o diagrama a seguir:



3 Implementação

Utilizando a linguagem c++, fizemos a implementação seguindo os conceitos de Programação Orientada a Objetos. Como citado na modelagem de problemas, identificamos os objetos como Objetos programacionais, e criamos suas classes. Logo depois, tendo cada escopo de classe definido e seu fluxo identificado no Diagrama de Classes, implementamos cada método com base nesse fluxo, mantendo toda a logística do jogo ao longo da execução. Por dúvidas nas decisões de implementação, nem todos os métodos necessários foram devidamente implementados, uma vez que, por questões de logística, encapsulamento e dúvidas, não sabemos como ou onde implementar os métodos, precisamos rever algumas decisões que tomamos, falaremos mais a seguir. Dentro dos conceitos vistos em sala de aula sobre Programação Orientada a Objetos, podemos observar o uso de abstração, uma vez que cada classe representa um objeto na realidade. Depois, podemos destacar o uso de herança entre as classes "Player"e "Dealer", uma vez que o Dealer tem os mesmos atributos que sua classe pai, além de um atributo extra que é sua carta fechada que somente ele pode ver. O terceiro pilar da POO é o encapsulamento, uma vez que escondemos todos os atributos do usuário, ou seja, os atributos são privados e só acessados via get/set. Por último, temos o polimorfismo, uma técnica poderosa que transforma um método de uma classe em outra, utilizado no cosntrutor da classe Carta, uma vez que cada cosntrutor tem seu objetivo e casos de usos. Também utilizamos o polimorfismo para o método que determina quem ganhou, uma vez que separamos a rodada do jogo e criamos classes pra cada, a rodada herda de jogo, e ambos possuem quem ganha, mas um é sobre a rodada e o outro o jogo todo, a soma de todas as rodadas para determinar o vencedor geral.

3.1 Decisões de Implementação

Como descrito acima, tomamos as decisões de implementação baseado nos pilares que regem a Programação Orientada a Objetos. Ainda temos alguns problemas a resolver e decidir para implementação, uma vez que temos dúvidas sobre como deixar a estrututa toda fluida e, ainda sim, ter uma lógica de desenvolvimento bem encapsulado e não colocarmos métodos em classes na qual não existe sentido semântico.

4 Conclusão

O jogo BlackJack foi modularizado e todo contruido em c++, utilizando a metodologia de Programação Orientada a Objetos. Dentro dos conceitos abordados, os pilares da POO foram utilizados no trabalho, como especificados no enunciado do trabalho. Ainda temos algumas decisões de desenvolvimento para tomarmos, uma vez que todo o código não está completo, além de que surgiram algumas dúvidas de como melhor implementar alguns métodos, e onde eles vão vifar encapsulados.

Referências

[1]