Desenvolvimento de um jogo de cartas em C++ - Presidente

Adolfo Steiner da Silva
Centro Tecnológico de Joinville
Universidade Federal de Santa Catarina
Joinville, Brasil
adolfostsi@hotmail.com

I. Introdução

O presente trabalho tem como objetivo apresentar o desenvolvimento do jogo de baralho Presidente feito ao longo do semestre na disciplina de Programação III, utilizando um framework desenvolvido em conjunto entre todos os alunos da disciplina, mostrando os problemas encontrados ao longo do desenvolvimento assim como as correções aplicadas. O jogo foi desenvolvido em C++ (linguagem estudada na disciplina ao longo do semestre). O desenvolvimento do jogo partiu como idéia de projeto com objetivo de aplicar os conceitos de programação orientada a objeto aprendidos durante o semestre de uma maneira descontraída e didática. As principais dificuldades do projeto se encontraram no desenvolvimento do framework e na implementação da interface gráfica.

II. Desenvolvimento

A. Regras do jogo

O jogo Presidente é um jogo de baralho popular que tem como base um baralho comum. O objetivo do jogo é descartar todas as cartas de sua mão primeiro para ficar com a nomeação de presidente. O jogo é para 4 jogadores, onde cada um recebe um total de 13 cartas no início de cada rodada. Na primeira rodada quando não existem nomeações, o jogador depois do dealer inicia o turno de descarte de cartas. Nas rodadas seguintes é sempre o presidente que inicia o descarte. Antes de cada início de rodada é necessário haver uma troca de cartas, onde o bobo cede suas duas cartas mais altas ao presidente, que cede suas duas cartas mais baixas ao bobo. O mesmo acontece com o vicebobo e o vicepresidente, porém eles trocam apenas 1 carta. Depois de feitas as trocas é iniciada a rodada.

O jogador que inicia o turno de descarte pode jogar qualque carta, podendo ser uma carta única, um par, uma trinca ou uma quadra. O jogador seguinte deve jogar o mesmo número de cartas que o anterior, porém com um valor maior seguindo a ordem definida (2 como carta mais alta, depois o Ás, Rei, Dama, Valete, 10, 9, 8, 7, 6, 5, 4 e por fim o 3). Quando nenhum dos outros jogadores possuir uma carta mais alta que a carta jogada , inicia-se um novo turno, o qual é iniciado pelo jogador que jogou a última carta no turno anterior. Quando acabam as cartas de um jogador, é atribuída a este a nomeação mais alta disponível segundo a ordem (presidente, vicepresidente, vicebobo e bobo).

Depois que três jogadores descartarem todas as cartas a rodada é finalizada, restando ao jogador que restou com cartas em mão a última posição. Sãa atribuídas, então, as pontuações de acordo com a nomeação de cada jogador: o presidente recebe três pontos, o vicepresidente dois, o vicebobo 1 e o bobo 0 pontos. Ao fim do número de rodadas escolhida pelo usuário para jogar o jogador com maior pontuação é o vencedor.

B. A implementação

O jogo foi implementada basicamente em duas IDEs: Eclipse e Qt Creator. O Eclipse foi utilizado para escrever a execução do jogo além das partes do framework designadas enquanto que o Qt Creator foi usado para a implementação da interface gráfica. A base do projeto começou sendo decidida durante as aulas, discutindo primeiro a relação entre as classes presentes no framework. Depois disso, iniciou-se a implementação do mesmo e o principal problema encontrado foi a dificuldade em tornar o namespace criado o mais genérico possível para todos os diferentes tipos de jogos que o utilizariam. Como solução a esta dificuldade optou-se pelo uso de templates para tipos de cartas. Depois, extendeu-se este conceito também para tipos de jogadores. No caso do meu jogo (Presidente), não foi necessário a utilização dos templates, então acabei utilizando as classes Carta e JogadorBasico.

Para implementar meu jogo fiz uma herança da classe JogoBasico para criar a classe JogoPresidente. Esses métodos foram utilizados para controlar a execução do jogo. Após issso, criei a classe Interface que usei para fazer a relação do meu jogo com o terminal. Essa classe foi substituída pela classe criada no Qt: TelaPresidente. Dentro desta classe, existem métodos para mostra a mão dos jogadores e mostrar os montes que estão na mesa contendo as cartas jogadas (Figura 1). Assim como métodos para mostrar a tela do menu (Figura 2), para mostrar o vencedor da partida e as pontuações em cada rodada. Além de fazer a interação para adiquirir o nome do jogador e número de rodadas que o mesmo deseja jogar (Figura 3).

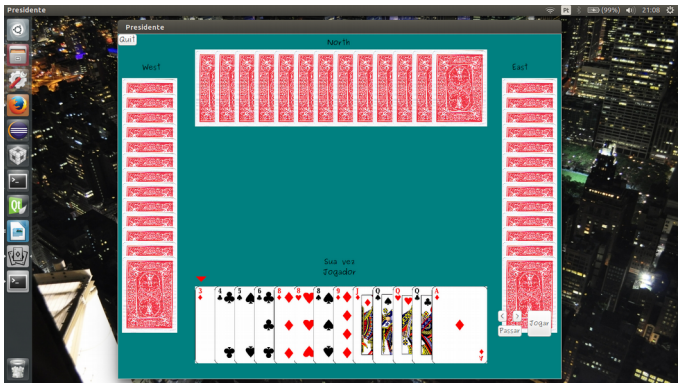

Fig. 1. ela do jogo.

Fig. 2. Tela de menu.

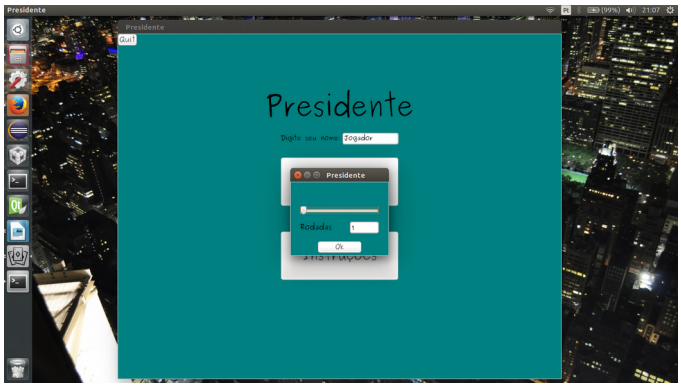

Fig. 3. Seleção do número de rodadas.

As execuções de jogadas são controladas pelo jogo, que difere entre uma jogada automática feita pela CPU e uma feita pelo jogador. O jogador tem acesso somente às suas cartas e não são permitidas jogadas erradas. Se o jogador tentar jogar um número de cartas diferente do necessário ou tentar jogar uma carta com valor menor ao exigido sua vez é passada. As cartas que devem ser trocadas entre as posições já são trocadas no início da rodada automaticamente para que não existam erros.

Quando o jogador descarta todas as suas cartas o jogo passa a correr numa velocidade maior para diminuir o tempo de espera do jogador até o início de uma nova rodada. Ao final da rodada é apresesentado ao usuário sua posição e a pontuação total de todos os jogadores na mesa (Figura 4) para que ele saiba se receberá ou entregará cartas no início da próxima rodada além de saber se está perto de vencer o jogo ou não.

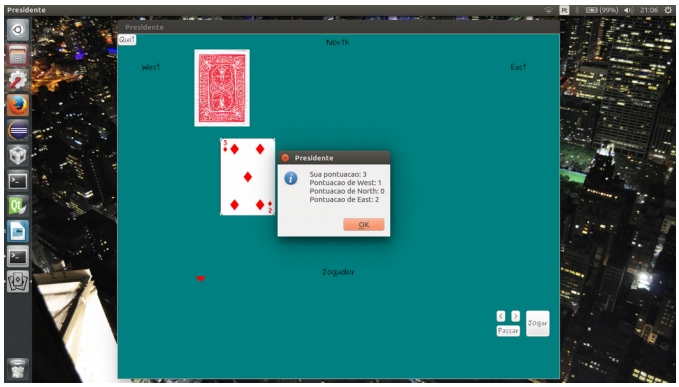

Fig. 4. Tela de pontuação.

Seguinte à isso iniciei a implementação da parte gráfica no Qt Creator. Criei quatro classes para isso, uma para representar as cartas na tela e as outras para representarem as telas que aparecem ao longo da execução do programa. A classe das cartas recebe como parâmetros de construtor o valor, o naipe como um inteiro, uma variável booleana que representa se está na mão do jogador ou não e a posição do ponto superior esquerdo da carta na tela. As classes de tela são a TelaPresidente mencionada anteriormente, Rounds onde seleciono o número de rodadas e WinnerWindow onde mostro o jogador vencedor. Na Figura 1 apresento a tela de jogo que é dada por um objeto da classe QgraphicsScene dentro da classe TelaPresidente. Na Figura 2 mostro a tela inicial do jogo desenvolvida na classe TelaPresidente utilizando outro objeto da classe QgraphicsScene. Na Figura 3 mostro a seleção de rodadas feita com a classe Rounds. Por fim na Figura 4 apresente a tela apresentada após cada rodada ao jogador mostrando sua pontuação.

III. Discussão

À partir da classe JogoPresidente criada, tive que implementar uma série de métodos que controlariam o jogo. Além disso também foram acrescentados um vetor de inteiros que serve para gravar a ordem dos jogadores que descartam suas cartas primeiro e um ponteiro para a classe TelaPresidente para que pudessem ser chamadas os métodos desta classe de dentro da execução do jogo. O método principal da classe é o play, onde toda a lógica de desenvolvimento do jogo acontece. Esta é a única função pública da classe, já que é a única que deve ser usada pelo usuário. Os outros métodos criados foram necessários para o controle das condições de jogo, fazendo o mesmo seguir as regras antes citadas.

A. Os métodos acrescentados

- Acha carta jogavel: método desenvolvido para conseguir uma jogada automática da CPU. É passado como parâmetros um vetor de Carta sendo a mão do jogador, um inteiro que diz o numero de cartas que precisam ser jogadas e um inteiro que diz qual o valor da carta a ser superada.

- Joga cartas: método feito para realizar a ação de descarte de cartas da mão do jogador para os montes. Os parâmetros são um inteiro com o valor da carta à ser

retirada da mão do jogador e o número de cartas a serem descartadas.

- How many in the hand: método para encontrar quantas cartas com um determinado valor se encontram na mão de um jogador. Os parâmetros são um vetor de Carta sendo a mão do jogador e um inteiro com o valor a ser procurado.
- Add position: este método serve para gravar a ordem dos jogadores que descartaram as cartas primeiro. É passado como parâmetro um inteiro que representa a posição do jogador que teve todas as suas cartas descartadas.
- Retorna bobo: método que retorna a posição na mesa do jogador que terminou a última rodada como bobo.
- Retorna pres: método que retorna a posição na mesa do jogador que terminou a última rodada como presidente.
- Retorn vicebobo: método que retorna a posição na mesa do jogador que terminou a última rodada como vicebobo.
- Retorna vicepres: método que retorna a posição na mesa do jogador que terminou a última rodada como vicepresidente.
- Find highest: método que encontra as maiores cartas da mão do jogador. Os parâmetros são o número de cartas a serem encontrados e a posição do jogador que terá sua mão investigada.
- Find lowest: método que encontra as menores cartas da mão do jogador. Os parâmetros são o número de cartas a serem encontrados e a posição do jogar que terá sua mão invetigada.
- Pass card: método que realiza a troca de cartas entre dois jogadores. Os parâmetros são um vetor de inteiros com o valor das cartas a serem passadas, a posição do jogador que cederá as cartas e a posição do jogador que receberá as cartas.
- Ordena mao jogador: método que ordena a mão do usuário do jogo para apresentar as cartas de maneira mais prática.
- Play: método que executa a lógica do jogo.

B. *Métodos Sobrescritos*

- Fim jogada: este método precisou ser reescrito para que realizassem as trocas de carta antes do início de cada rodada e também para que fossem atribuídas as pontuações à cada jogador.
- Verifica fim de jogo: o método foi reescrito para atender as especificações da regra, quando atingido o número máximo de rodadas o jogo é encerrado.
- Verifica vitoria: foi alterado para checar que jogador tem maior pontuação para declará-lo vencedor.
- Verifica jogador pontuacao maxima: foi sobrescrito para que chamasse o método que declara vencedor ao final da excução da função.
- Declara vencedor: foi sobrescrito para que chamasse o método da tela que apresenta o jogador vencedor.
- Verifica jogador unico: esse método foi sobrescrito para que a rodada se encerrasse quando só restasse um jogador apto e assim as cartas desse jogador eram retiradas de sua mão e sua posição era adicionada ao vetor de posições.

C. *Justificativas*

Segundo a implementação escolhida, quando o jogador seleciona uma carta para jogar ele sempre descartará o maior número de cartas possível com aquele valor, por isso foi implementado o método “how many in the hand”. Esses métodos precisaram estar na classe jogo devido a arquitetura do framwork ser composta totalmente por agregação de classes fazendo com que com o jogo criado não pudesse ocorrer o acesso a qualquer outra classe senão por meio dele.

Com relação à interface, optei por fazer todas as ações utilizando o mouse e não o teclado e através de botões que se encontram na tela. O cursor é movido com o auxílio de dois botões durante o jogo fazendo-o ir da esquerda para direita, além de um botão para passar a vez e outro para jogar a carta selecionada.

IV. Conclusão

Apesar de que, em minha convicção, o trabalho seria cumprido com muito mais simplicidade sem o uso de um framework comunitário, a experiência de desenvolvimento em equipe se mostrou em certos aspectos de grande proveito, pois promoveu a discussão de idéias para melhores práticas de programação. Mesmo que ao final não se tenha conseguido uma base altamente genérica e abrangente quanto possível, pode-se concluir a implementação utilizando do código desenvolvido em grupo facilmente.

Outro ponto destacável do trabalho seria o uso do Qt Creator como plataforma para criação da interface gráfica. A IDE foi de grande utilidade, facilitando muito o desenvolvimento por apresentar muitas fontes de ajuda e muitos bons exemplos que poderiam ser encontrados sem muito esforço através de uma busca rápida.