

Especificação de requisitos

Jogo de baralho 99

Versão	Autores	Data	Ação
1.0	Amanda Oliveira e Willian Kruscinski	09/05/2022	Estabelecimento dos requisitos
1.1	Amanda Oliveira e Willian Kruscinski	16/05/2022	Adição das regras do jogo e esboço da interface gráfica
1.2	Amanda Oliveira, Matheus Novais e Willian Kruscinski	18/09/2022	Atualizar RFs e RNFs.
1.3	Amanda Oliveira, Matheus Novais e Willian Kruscinski	01/10/2022	Atualizar RFs.
1.3.1	Amanda Oliveira, Matheus Novais e Willian Kruscinski	12/11/2022	Explanação de regra
1.4	Matheus Novais	18/12/2022	Atualização tabela 1

CONTEÚDO

1. Introdução;
 - 1.1 Definições e abreviações
 - 1.2 Regras do jogo
 - 1.3 Esboço da interface
2. Visão geral;
3. Requisitos de software.

INTRODUÇÃO

Objetivo: desenvolvimento de um programa que suporte a disputa de partidas do jogo de baralho 99 na modalidade usuários contra usuário, estando ambos em diferentes máquinas.

Referências: Canal VemKa Jogar <<https://www.youtube.com/watch?v=vdk1-1MhUJo&t=>>.

DEFINIÇÕES E ABREVIações

Neste documento, as seguintes abreviações serão utilizadas:

- RF: Requisito Funcional
- RNF: Requisito Não Funcional

REGRAS DO JOGO

1. Deve ser jogado com um baralho de 52 cartas.
 - a. Este inicialmente comporá total o monte de compra, mas terá seu estado alterar ao longo do jogo, ficando dividido entre o monte de compra e o monte de descarte.

2. Ao iniciar o jogo, cada jogador recebe 3 cartas e durante todo o jogo cada jogador deve ter 3 cartas na mão.
 - a. Ao descartar uma carta, se compra outra do monte.
3. Cada carta do monte irá agregar um valor ao placar, vide Tabela 1.
4. As cartas que são descartadas tem seus valores somados ao placar da partida.
5. O jogador que fizer a jogada que ultrapasse o valor de 99 do placar perde, tornando seu oponente o ganhador.

Tabela de referência de valores por carta:

Carta	Valor
A	1
2	2
3	3
4	0
5	5
6	6
7	7
8	8
9	0
10	-10
J	10
Q	10
K	fixa o placar em 99

Tabela 1

VISÃO GERAL

Arquitetura do programa: programa orientado a objetos em arquitetura cliente-servidor distribuído.

Premissas de desenvolvimento:

- O programa deve apresentar uma interface gráfica bidimensional;
- O programa deve ser implementado em Python.
- O programa deve ser produzido em meio distribuído sob DOG.

REQUISITOS DE SOFTWARE

Requisitos Funcionais:

RF 1. Iniciar programa: ao ser executado, o programa deve apresentar na interface a mesa do jogo em seu estado inicial (o baralho em sua respectiva área, o restante da mesa vazia e a mensagem '99'). Após isso, deve solicitar conexão com DOG Server (utilizando os recursos de DOG). O resultado da tentativa de conexão deve ser informado ao usuário. Apenas em caso de conexão bem sucedida as demais funcionalidades estarão habilitadas. No caso de conexão mal sucedida, a única alternativa deve ser encerrar o programa;

RF 2. Iniciar jogo: o programa deve apresentar a opção de menu "iniciar jogo" para o início de uma nova partida. O procedimento de início de partida consiste em enviar uma solicitação de início a Dog Server, que retornará o resultado, que será a identificação e a ordem dos jogadores, em caso de êxito, ou a razão da impossibilidade de início de partida, caso contrário. A interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja

quem inicia a partida, a interface deve estar habilitada para seu procedimento de lance. Esta funcionalidade só deve estar habilitada se o programa estiver em seu estado inicial, isto é, sem partida em andamento e com a mesa em seu estado inicial;

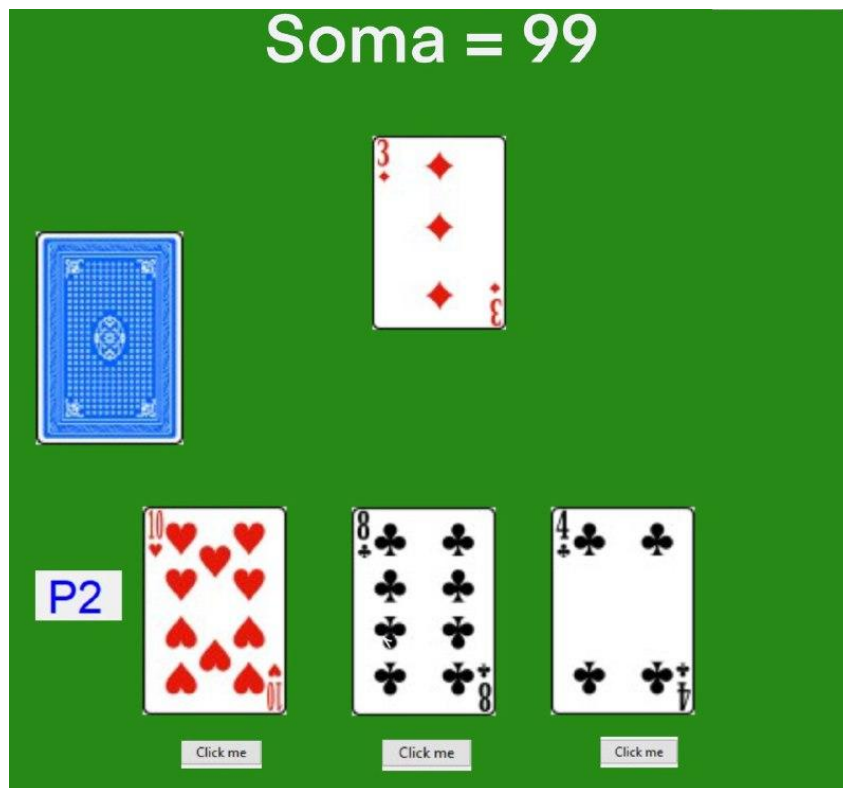
- RF 3. Restaurar estado inicial:** o programa deve apresentar a opção de menu “restaurar estado inicial” para levar o programa ao seu estado inicial, isto é, sem partida em andamento e com o tabuleiro em seu estado inicial. Esta funcionalidade só deve estar habilitada se o programa estiver com uma partida finalizada;
- RF 4. Procedimento de lance:** o programa deve analisar se há cartas disponíveis no monte de compra, se não houver ele deve embaralhar o monte de descarte e então utilizá-lo como monte de compra. Havendo cartas no monte de compra o programa deve permitir a um jogador habilitado o procedimento de lance onde por meio de click na carta esta deve ser colocada no monte de descarte da mesa e o placar deve ser alterado. A cada lance, o programa deve verificar se o jogador que jogou perdeu a partida; No caso de encerramento de partida, deve ser notificado o nome do jogador vencedor; no caso de não encerramento, deve ser desabilitado o jogador local e o programa fica no aguardo de jogada do adversário (ver RF 6) ou de notificação de abandono (ver RF 7);
- RF 5. Receber determinação de início:** o programa deve poder receber uma notificação de início de partida, originada em Dog Server, em função de solicitação de início de partida por parte de outro jogador conectado ao servidor. O procedimento a partir do recebimento da notificação de início é o mesmo descrito no 'RF2 – Iniciar jogo', isto é, a interface do programa deve ser atualizada com as informações recebidas e caso o jogador local seja quem inicia a partida, a interface deve estar habilitada para seu procedimento de lance.

- RF 6. Receber jogada:** o programa deve poder receber uma jogada do adversário, enviada por Dog Server, quando for a vez do adversário do jogador local. A jogada recebida deve conter as informações especificadas para o envio de jogada no 'RF 4'. O programa deve alterar a carta do topo do monte de descarte de acordo com o lance do adversário.
- Após isso, deve-se avaliar o encerramento de partida. No caso de encerramento de partida, deve ser notificado o nome do jogador vencedor; no caso de não encerramento, deve ser habilitado o jogador local, para que possa proceder a sua jogada;
- RF 7. Receber notificação de abandono:** o programa deve poder receber uma notificação de abandono de partida por parte do adversário remoto, enviada por Dog Server. Neste caso, a partida deve ser considerada encerrada e o abandono notificado na interface.

Requisitos Não-Funcionais:

- RNF 1. Especificação de projeto:** além do código Python, deve ser produzida especificação de projeto baseada em UML.
- RNF 2. Especificação de projeto:** o jogo deve ter funcionamento distribuído sob uso do DOG.
- RNF 3. Interface gráfica para usuário:** o programa deverá ter interface gráfica única similar a do esboço abaixo, disponível e atualizada a

cada nova jogada recebida a partir e pelo DOG;



RNF 4. Tecnologia de interface gráfica para usuário: A interface gráfica deve ser baseada em TKinter.