

Projeto “Yoté”

Versão	Autor(es)	Data	Ação
1.4	Lucas Cardoso Soares André Amaral Rocco João Pedro Perez Resmer	26/11/2023	Alteração no critério de empate do jogo

Conteúdo:

1.1. Introdução	3
1.1 Objetivo	3
1.2 Definições, abreviaturas	3
1.3 Referências:	3
2.Visão Geral	3
2.1 Arquitetura do programa	3
2.2 Premissas de desenvolvimento	3
3.Requisitos de Software	3
3.1 Requisitos Funcionais:	3
3.2 Requisitos Não Funcionais	6
Apêndice: Regras “Yoté”	7

1.1. Introdução

1.1 Objetivo

Desenvolvimento de um programa distribuído que suporte a disputa de partidas de Yoté na modalidade usuário contra usuário.

1.2 Definições, abreviaturas

Regras do jogo: ver apêndice

1.3 Referências:

Apresentação das regras do jogo (video do canal Vem Ka Jogar):

 COMO JOGAR YOTE - UM DO MELHORES JOGOS DE TABULEIRO DO MUNDO

2. Visão Geral

2.1 Arquitetura do programa

Cliente-servidor distribuído.

2.2 Premissas de desenvolvimento

- O programa deve ser implementado em Python;
- O programa deve usar PyNetGames como suporte para execução distribuída;
- Além do código, deve ser produzida especificação de projeto baseada em UML, segunda versão.

3. Requisitos de Software

3.1 Requisitos Funcionais:

Requisito funcional 1 - Iniciar programa: Ao ser executado, o programa deve apresentar o tabuleiro do jogo em seu estado inicial (todas as peças dos jogadores em suas respectivas pilha de peças e tabuleiro vazio e a mensagem ‘Yoté’). Depois disso, deve solicitar conexão

com o PyNetGames Server (usando os recursos do PyNetGames). 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.

Requisito funcional 2 - Iniciar jogo: Após a conexão com o PyNetGames Server for bem sucedida, o jogo deve enviar uma solicitação para iniciar uma partida. O procedimento de envio de partida consiste em enviar uma solicitação de início ao PyNetGames 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 o tabuleiro em seu estado inicial.

Requisito funcional 3 - Selecionar peça: O programa deve permitir a um jogador habilitado selecionar uma peça sua presente no tabuleiro ou em sua pilha de peças de modo a iniciar um movimento (selecionando a origem do movimento). Se a ação for executada após o jogador já ter uma peça selecionada, o programa deve manter como seleção válida a peça a qual foi clicada por último pelo jogador.

Requisito funcional 4 - Selecionar destino: O programa deve permitir a um jogador habilitado selecionar uma posição do tabuleiro, onde será colocada a peça previamente selecionada (de sua pilha de peças ou de uma posição do tabuleiro). Esta funcionalidade só deve estar habilitada se o programa estiver com peça selecionada (ver Requisito funcional 3), seja ela do tabuleiro ou da pilha. O programa deve avaliar se a posição destino pode receber a peça selecionada, de acordo com as regras do jogo (ver apêndice). Em caso positivo, deve efetuar a jogada - isto é, remover a peça de seu posicionamento inicial (seja este a pilha ou o tabuleiro) e colocá-la na posição destino - e, em de lance irregular e o programa deve novamente aguardar a primeira ação do jogador habilitado (selecionar peça ou selecionar origem). No caso de êxito na colocação de peça em seu destino, o programa deve enviar a jogada ao adversário (utilizando os recursos do PyNetGames) e avaliar o encerramento de partida. A jogada enviada deve conter a posição de origem do tabuleiro (no caso de seleção de origem) ou a posição da peça selecionada (no caso de seleção de peça da sua área) e a posição destino. No caso de encerramento de partida, deve ser notificado a cor das peças do

jogador vencedor ou a situação de empate; no caso de não encerramento, deve ser desabilitado o jogador local e o programa fica no aguardo de jogada do adversário (ver Requisito funcional 7) ou de notificação de abandono (ver Requisito funcional 8).

Requisito funcional 5 - Receber determinação de início: o programa deve poder receber uma notificação de início de partida, originada em PyNetGames 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 ‘Requisito funcional 2 – 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.

Requisito funcional 6 - Receber jogada: o programa deve poder receber uma jogada do adversário, enviada por servidor, quando for a vez do adversário do jogador local. A jogada recebida deve ser um lance regular e conter as informações especificadas para o envio de jogada no ‘Requisito funcional 5 – Selecionar destino’. O programa deve remover a peça da origem definida e colocá-la no destino. Após isso, deve-se avaliar o encerramento de partida. No caso de encerramento de partida, deve ser notificado o nome do jogador vencedor ou da situação de empate; no caso de não encerramento, deve ser habilitado o jogador local, para que possa proceder a seu lance;

Requisito funcional 7 - Receber notificação de abandono: o programa deve poder receber uma notificação de abandono de partida por parte do adversário remoto, enviada pelo PyNetGames. Neste caso, a partida deve ser considerada encerrada e o abandono notificado na interface.

3.2 Requisitos Não Funcionais

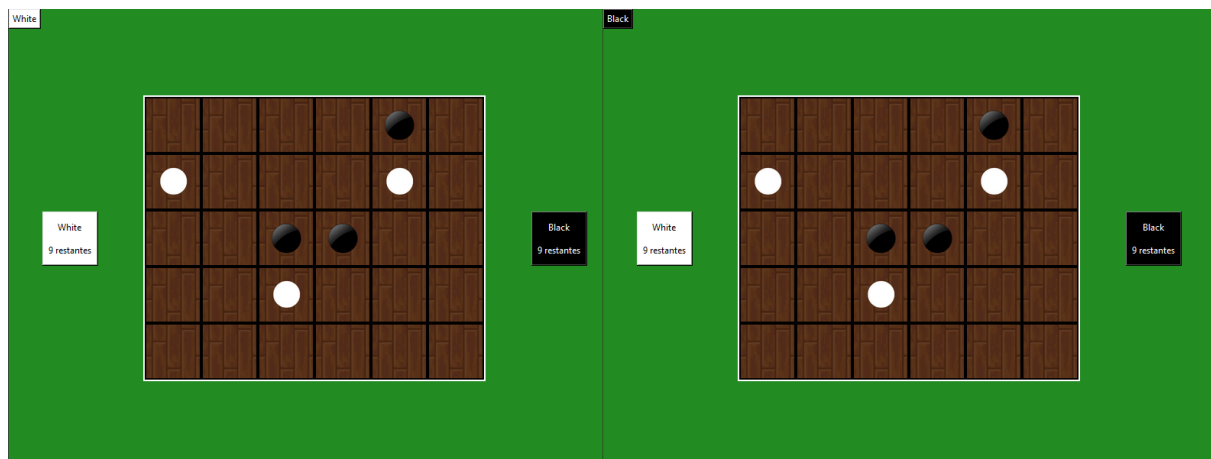
Requisito Não Funcional 1 - Versão do Python: Para o desenvolvimento do programa, deverá ser utilizado Python versão 3.

Requisito Não Funcional 2 - Biblioteca gráfica: A biblioteca gráfica adotada deverá ser Tkinter.

Requisito Não Funcional 3 - Modelagem de Diagramas: Referente a modelagem de diagramas, deve ser feita baseada em UML utilizando a ferramenta Visual Paradigm.

Requisito Não Funcional 4 - Biblioteca de jogo distribuído: O jogo deverá implementar jogabilidade em rede entre dois jogadores remotos utilizando-se da biblioteca PynetGames.

Requisito Não Funcional 5 - A interface do usuário (GUI) deve seguir os esboços anexados:



Apêndice: Regras “Yoté”

Número de jogadores: Dois.

Objetivo: Capturar ou bloquear todas as peças do adversário. Vence aquele que elimina ou bloqueia o seu adversário para que não haja possibilidade de movimentos. Se ambos os jogadores ficarem com 3 peças ou menos no tabuleiro, a partida termina empatada.

Regras:

Yoté é jogado por duas pessoas em um tabuleiro 5x6 (cinco linhas de seis colunas), como mostrado na imagem do tabuleiro. Cada jogador inicia o jogo com 12 peças em sua “mão”. Um dos jogadores jogará com as peças brancas e o outro jogador jogará com as peças pretas.

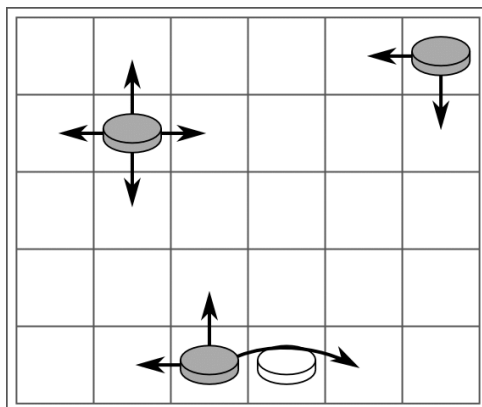


Imagem do tabuleiro. Fonte: <http://www.cyningstan.com/>

1. No início do jogo, o tabuleiro está vazio.
2. O jogador que segura as peças pretas joga primeiro, após o qual o jogador com as brancas faz sua jogada, alternando entre eles.
3. Em seu primeiro turno, cada jogador deve colocar uma peça em qualquer quadrado vago do tabuleiro.
4. Nos turnos subsequentes, um jogador pode optar por colocar outra peça se tiver alguma em sua mão, ou em vez disso, mover uma peça que já esteja no tabuleiro.
5. Uma peça move-se uma casa horizontal ou verticalmente, para uma casa adjacente, que deve estar vazia. Alguns movimentos de exemplo são mostrados na imagem do tabuleiro. Não existem movimentos diagonais neste jogo.
6. Uma peça inimiga é capturada saltando sobre ela e pousando na casa além, que deve estar vazia. Isso é mostrado no diagrama.
7. A peça capturada é removida do tabuleiro e não participa mais do jogo.

8. Após capturar uma peça, o jogador seleciona então uma segunda peça inimiga e a remove do tabuleiro.
9. Assim como nos movimentos, capturas diagonais não são permitidas.
10. Este jogo não possui múltiplos saltos.
11. O jogo termina quando um jogador captura todas as peças inimigas, nesse caso ele é declarado o vencedor.
12. Se um dos jogadores não possuir nenhuma jogada válida para efetuar, o outro jogador é declarado vencedor.
13. Se nenhum jogador conseguir fazer uma ação válida ganhará quem tiver mais peças no tabuleiro.
14. Se ambos os jogadores ficarem com exatamente uma peça simultaneamente, o jogo é declarado empate.