Tabuleiro de Números

Leonardo Gama Teixeira

[1]Universidade Estadual de Feira de Santana(UEFS)
Curso de Engenharia da Computação

leogamateixeira@gmail.com

Abstract. *The IEEE UEFS Games League launched a game development challenge for incoming Computer Engineering students. This report presents the development of the proposed game, Numbers Board.*

This system was built in the Python language, using language tricks such as building lists, matrices and functions. In addition to the JSON library, COPY and an external TABULATE library for displaying the board. The program is capable of making matches, checking victories and saving unfinished files for continuation at another time.

Resumo. *O grupo da Liga de Jogos do IEEE UEFS lançaram um desafio de desenvolvimento de um jogo para os alunos ingressantes do curso de Engenharia de Computação. Esse relatório apresenta o desenvolvimento do jogo proposto, o Tabuleiro de Números.*

Esse sistema foi construído na linguagem Python, utilizando-se de artifícios da linguagem como construção de listas, matrizes e funções. Além da biblioteca JSON, COPY e uma biblioteca externa TABULATE para exibição do tabuleiro. O programa é capaz de fazer partidas, verificar vitórias e salvar arquivas inacabadas para a continuação em outro momento.

1. Introdução

O grupo da Liga de Jogos do IEEE UEFS está buscando novos integrantes para ingressar no grupo, considerando isso, isso lançaram um desafio de desenvolvimento de um jogo para os alunos ingressantes do curso de Engenharia de Computação da (UEFS). Esse relatório apresenta o desenvolvimento do jogo.

Como forma de solucionar o problema e desenvolver o jogo, como aluno de MI algorítimos desenvolvi um software em Python capaz de iniciar uma partida, a partir de uma dificuldade selecionada, iniciar uma partida, criando o tabuleiro e sorteando objetivos para cada jogador, estabelecer a vitória do jogador que completar seu objetivo, salvar nome do ganhador, ou caso não termine a partida, a opção de salvar a partida em um arquivo JSON. Estruturas de repetição, condicionais, listas, matrizes e funções foram alguns dos recursos da linguagem usados para confecção desse programa.

2. Metodologia e Fundamentação Teórica

Durante as sessões tutoriais, foi primeiramente discutido modularização, como utilizar cores no terminal, e usar sons no jogo, também foi estudado como seria construído o tabuleiro através das matrizes. Foi discutido em algumas sessões como seriam as condições de vitórias, pois caso os dois jogadores completassem o seu objetivo ao mesmo tempo, teria vencedor? Caso houvesse, quem seria?, na última sessão ficou estabelecido que ficaria de escolhe pessoal, foi pessoalmente selecionado que, caso os dois completem o objetivo na mesma jogada, a pessoa que fez a jogada ganhará a partida.

No início das sessões foi pensado como seriam feitos os salvamentos das partidas inacabadas e do ranking dos jogos, foi escolhido CSV para ambos, porém ao final eu trouxe uma solução mais simples, o uso de arquivos json, além de ser mais fácil de manipular os arquivos do que CSV o json permitia a criação de dicionários, o que facilita na visualização das informações no código.

Algumas ferramentas para o desenvolvimento do jogo sendo uma solução pessoal do problema como a criação de matrizes através do método de list comprehension, a verificação da jogada através da verificação se o número digitado existia nas possíveis jogadas, e após ser jogado, era retirado da lista para não ser jogado novamente.

2.1. Definição de Requisitos

O grupo da Liga de Jogos do IEEE UEFS requisitou as seguintes features no jogo:

- Cada jogador, em sua vez e de forma alternada, pode selecionar um número dentre os números disponíveis e posicionar este número em uma das casas.
- Ganha o jogador que fizer a sequência de N números em linha (diagonal, vertical ou horizontal, com leitura da esquerda para a direita e de cima para baixo) que atende ao seu objetivo.
- O jogo termina em empate se todas as casas do tabuleiro forem marcadas sem que nenhum jogador tenha completado uma sequência de objetivos. definição do objetivo de cada jogador é sorteada no início do jogo e somente o jogador sabe seu objetivo.
- O jogo será jogado em turnos usando o mesmo computador, mas deve haver diferenciação visual de qual jogador colocou cada número no tabuleiro.
- Uma opção que pode ou não ser ativada antes de iniciar uma partida, é incluir uma jogada especial para cada jogador. Uma jogada especial permite que um jogador remova uma linha ou uma coluna de números do tabuleiro.
- Deve ser permitida também a possibilidade de salvar um jogo antes de terminar a partida.
- Manter um ranking de triunfos de jogadores também é um requisito que pode adicionar elemento competitivo ao jogo.

2.2. Descrição de alto nível

O programa não tem interface gráfica, toda interação ocorre dentro do terminal do python, o jogo se inicia com 4 opções, jogar, tutorial, últimos vencedores e sair do jogo, a partir dessas 4 opções, no menu jogar, existem duas opções de começar um novo jogo e continuar um jogo, a opção começar um novo jogo, cria o tabuleiro, os números que podem ser jogados a partir da dificuldade escolhida na função de dificuldade e depois

vem a opção se terá jogada especial na partida, depois sorteia o objetivo dos jogadores, mostra na tela e começa o jogo. A opção de continuar um jogo, usa de um arquivo externo com dados salvos de um jogo anterior não finalizado, e começa o jogo com as informações salvas.

Na partida, é possível fazer jogadas, tanto normal, quanto a especial que consiste em limpar uma linha ou uma coluna do tabuleiro, tem as opções ”sair e salvar o jogo”e ”sair sem salvar o jogo”. Existe uma diferenciação visual entre os players, sendo azul o player1 e vermelho player2, ao final de cada jogada o programa checa se a jogada feita completou o objetivo de algum dos jogadores, se não o jogo continua, até alguém completar seu objetivo, ou não houver mais casas disponíveis para jogar o que resulta em um empate. A opção que salva o jogo, cria um arquivo com os dados da partida e salva, que podem ser recuperados pela opção ”continuar o jogo”, e ”sair sem salvar”simplesmente sai do jogo.

A opção de tutorial, mostra um pequeno texto com as informações básicas sobre o jogo. A opção da tabela de vencedores, teoricamente mostraria uma tabela com os últimos vencedores, porém ela estará sempre vazia por não salvar o ranking dos vencedores será apenas uma impressão sem informações reais. e a última opção encerra o programa.

2.3. Ordem de codificação

A ordem em que o código foi escrito foi mediada através das sessões tutoriais. Inicialmente foi estabelecido para serem criados o sorteamento de objetivos e o tabuleiro. Após ter objetivos e o tabuleiro, foi estudado como seriam executadas as jogadas, e após a finalização das jogadas, foi estudado como seriam feitas as jogadas especiais. A adição de salvar o jogo e sair, e a opção de mostrar ranking de jogadores e suas vitórias em um placar.

O programa foi desenvolvido em Python 3.12.3, utilizando de dois sistemas operacionais, um desktop com Windows 11 e um notebook com Ubuntu 22.04.4 LTS e o IDE utilizado foi o Visual Studio Code.

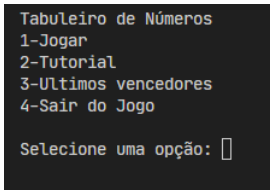
3. Resultados e Discussões

Para uma maior compreensão do funcionamento do produto, será mostrado como utilizá-lo, os testes feitos em seu desenvolvimento, erros que foram encontrados e podem ser encontrados.

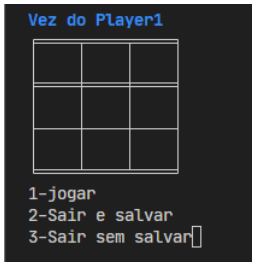
3.1. Manual de Uso

Para o programa funcionar, é necessário a instalação de uma biblioteca externa chamada tabulate na máquina, pelo próprio terminal python é possível instalar a biblioteca, através do comando ”pip install tabulate”, porém é recomendado que seja instalado um ambiente controlado.

- O programa abre mostrando o menu inicial:
- **1- Jogar:**
  Leva o usuário a tela de iniciar o jogo, aqui o usuário escolhe entre iniciar um jogo, ou continuar um jogo já existente;



- Caso for selecionado a opção de começar um novo jogo, será primeiramente perguntado a dificuldade, e em seguida se terá modo especial, após a escolha será mostrado os objetivos dos jogadores e o jogo se inicia. Caso seja selecionado a opção de continuar o jogo, será feita a captura dos dados do último jogo, e em seguida o jogo se inicia;
- **tela do jogo:**



- Dentro da tela de jogo, existem as opções de fazer a jogada, sair e salvar o jogo e sair sem salvar o jogo. Ao jogar, se a jogada especial tiver ativada pergunta se ela será utilizada, e em seguida é mostrado quais números podem ser jogados e pergunta onde serão jogados, após isso finaliza a jogada e muda para o player 2;
- **Sair e salvar**, pergunta qual o nome do arquivo será salvo o jogo, e em seguida retorna ao menu;
- **Sair sem salvar**Sair sem salvar, apenas retorna ao menu sem salvar o jogo;
- **2- Tutorial:**
  Leva o usuário a uma tela com um texto explicando como o jogo funciona em um texto simples;
- **3- Últimos vencedores:**
  Leva o usuário a uma tela com o ranking dos últimos vencedores;
- **4- Sair do jogo:**
  Finaliza o programa.

3.2. Testes e Erros

Diversos testes foram realizados durante a construção do programa, foram realizados testes para a construção do menu, vendo se todos os menus estavam levando as telas corretas, após todas as telas estarem montadas foi adicionado a função de jogar o número, E foram encontrados os seguintes erros:

- **Erro nas sequências vitoriosas:**
  Ao programar as funções que davam as sequências que levavam o player a vitória, foi encontrado um erro, em que a lista que gerava a sequência inversa dos objetivos ”par”e ”impar”, vinham faltando sempre os últimos números. exemplo em

um jogo 3x3 onde o objetivo era impar: [5,7,9], a próxima sequência deveria ser [9,7,5], porém essa sequência era pulada e a próxima era [7,5,3], ao final a matriz ficava [...[5,7,9],[7,5,3]...], isso ocorria, pois a função montava às sequências e parava quando não era mais possível montar(caso tivesse apenas 2 números para montar uma sequência de 3, a função parava), só que com isso, também eram excluídos da sequência reversa, por ela ser dependente da sequência principal, o erro foi resolvido criando as duas listas em paralelo, tanto a crescente quando a decrescente;

- **Erro nas cores:**
  Ao tentar adicionar cores as jogadas, elas não estavam sendo impressas no terminal, após checar a função de criar tabuleiro, foi notado que o tabuleiro que tinha as strings tinha o mesmo endereço de memória que o tabuleiro numérico, o que fazia com que ao alterar um os dois eram alterados, esse problema foi resolvido com a biblioteca nativa do python copy, usando deepcopy para gerar um endereço único para o tabuleiro de strings;
- **Erro na abertura dos arquivos:**
  Caso não tenha um arquivo json, já existente com o nome que o usuário seleciona para continuar um jogo salvo, o programa devolve um erro no terminal, pois não foi adicionado try-except na abertura do arquivo json, o que faz o erro ocorrer. Acontece um erro também ao tentar salvar o nome do ganhador, e a quantidade de vitórias, pois o jogador vitorioso ao adicionar o nome, o nome não é adicionado e é criado um arquivo vazio csv. Caso tente imprimir a tabela de últimos vencedores, a tabela é impressa, porém, é mostrado um ≪class 'OSError'¿"e a tabela também estará vazia, pois não tem dados nela, e nem podem ser adicionados, para corrigir esse erro apenas o link entre o nome digitado e o arquivo feito de forma correta faria com que a tabela de jogadores vitoriosos seja mostrada, apenas usando sorted() e set(), para organizar os dados pelas vitórias e para deixar os dados únicos sem os repetir.

4. Conclusão

Em conclusão, serão mostradas análises sobre os objetivos cumpridos, e não cumpridos e possíveis melhorias do sistema final: Ao analisar os objetivos propostos pelo problema é notado que o programa não conseguiu alguns objetivos, sendo esses:

- Salvar e recuperar o jogo, no quesito onde a jogada especial não é salva se tinha ou não na partida, fazendo com que mesmo que o usuário escolha ter a jogada especial, se salvar o jogo e tentar jogar novamente será entregue como se não houvesse jogada especial;
- ranking de jogadores e salvar e recuperar ranking dos jogadores, pois essa parte não foi executada de forma correta no programa, deixando ele aberto a erros;
- Dois erros causados pela falta do try-except na execução do programa.

Referências

https://docs.python.org/3/library/functions.html. - *Built-in Functions*.

https://docs.python.org/3/library/json.html. - *JSON encoder and decoder*.

[https://docs.python.org/3/library/json.html] [https://docs.python.org/3/library/functions.html]