

Tabuleiro de números

Problema 3 - MI Algoritmos

Maria Clara N Ramos

¹Departamento de Exatas – Universidade Estadual de Feira de Santana (UEFS)
Caixa Postal: 44.036-900 – Feira de Santana – BA – Brazil

m clara9593@gmail.com

Abstract. *This report refers to the production of software developed in the interpretation language “python 3.12”, aiming to answer the demand of the Games League of the IEE (Institute of Electrical and Electronic Engineers) of the State University of Feira de Santana (UEFS). is looking for members, and suggested a challenge to spark interest among Computer Engineering students. It was proposed to create a board game with numbers executed in this product through the matrix data structure, which forms the basis of the game, and continues among others such as repetitions and file processing compositions.*

Resumo. *Este relatório se refere à produção do software desenvolvido na linguagem de interpretação “python 3.12”, objetivando atender a demanda da Liga de Jogos do IEE (Institute of Electrical and Electronic Engineers) da Universidade Estadual de Feira de Santana . Tal ramo se encontra em busca de membros, e sugeriu um desafio para despertar interesse aos alunos de Engenharia de Computação. Foi proposta a criação de um jogo de tabuleiro com números executada neste produto através da estrutura de dados matriz, na qual é formada a base do jogo, e prossegue por entre outras como repetições e composições de tratamento de arquivos.*

1. Introdução

Se faz evidente a forte presença dos jogos eletrônicos- de quaisquer gênero- no século XXI, sejam eles apenas voltados ao lazer ou de teor competitivo. Existem também simulações de jogos físicos como xadrez e ludo, com turnos alternados onde cada jogador precisa traçar uma estratégia de vitória analisando a jogada do oponente.

Visando novos ingressos ao time, a Liga de Jogos do ramo IEE da UEFS propôs a criação do “Tabuleiro de números”. Tal projeto diz a respeito de um jogo competitivo ambientado em um tabuleiro $N \times N$, onde é indicado a cada jogador quais sequências numéricas devem ser formadas dentro das casas definidas, conforme objetivo proposto (por meio de sorteio). Foi solicitado aos alunos de Engenharia de Computação da Universidade Estadual de Feira de Santana, um software que possibilite essa gameplay de modo a cumprir com os requisitos evidenciados nas pré definições, (como os tamanhos de tabuleiro disponíveis e possibilitar salvamento de partida e de vencedores anteriores). Além disso, deve obedecer às regras e execuções de jogo, relacionadas a validação da sequência e uso de habilidade extra, por exemplo.

A solução desenvolvida foi um código fonte escrito em *python*, que cria uma interface interativa no terminal, na qual os jogadores recebem seus objetivos e cores

específicas, possibilitando a identificação de turnos, marcados pela escolha de uma casa, um número e, se habilitada, uma jogada especial. Caso o usuário decida interromper o jogo, é informado que ação deve ser executada para salva-lo, e o mesmo pode escolher continuar. Além disso, há uma opção no menu que ordena os jogadores por nome conforme o número de vitórias.

2. Metodologia e Fundamentação Teórica

Para obtenção dos resultados foram aplicados conhecimentos construídos durante as sessões PBL (Ensino Baseado em Problemas), tanto através das discussões em grupo como por intermédio do tutor Ângelo; pesquisas na documentação de bibliotecas da linguagem; em sites de ensino como w3schools e aulas de algoritmos ministradas pela professora Michele.

Durante as sessões, foi discutido a respeito da estilização do jogo e modularização do código, visando tornar ambos mais organizados e de fácil interpretação, como também da necessidade da realização de testes durante o desenvolvimento. Ainda almejando tais propriedades, se tornou de consenso geral a possibilidade de uso de bibliotecas, como *tabulate*, *winsound*, *curses*, *inquire*. Não obstante, foram definidos como fatos:

- Ganha quem cumprir o objetivo primeiro, este, que é sorteado, e pode ser o mesmo para os dois jogadores
- O empate deve ser anunciado quando as casas do tabuleiro acabam sem que ambos cumpram o objetivo;
- O uso da jogada especial (apaga uma sequência no tabuleiro e depois permite jogada convencional) deve ser habilitada antes da partida, caso desejar, e só pode ser utilizada uma vez por jogador;
- O número de casas da tabela é dado por N^2 , sendo que N é uma ordem definida com base na dificuldade escolhida pelo usuário;
- O software deve ter a possibilidade de trabalhar com dois arquivos diferentes, um direcionado ao salvamento do jogo e outro ao ranking de *players*;
- Foi liberado o uso de classes e *try except*, porém deve-se priorizar esse último de modo compacto e quando foge do controle do programador;
- Os números escolhidos devem ser limitados aos que ainda não foram jogados, e à quantidade de casas da tabela;
- Deve-se considerar a diagonal secundária na validação de vitória;

2.1. Definição de Requisitos

As funcionalidades do sistema requisitadas foram:

1. Opções de menu principal:
 - 1.1. Seleção de dificuldade do jogo com base no tamanho;
 - 1.2. Seleção de jogada especial antes da partida;
 - 1.3. Exibição de objetivo dos jogadores;
2. Jogadas alternadas:
 - 2.1. Diferenciação visual do tabuleiro para cada jogador;
 - 2.2. Verificação de jogadas inválidas (casas ou números);

3. Aplicação da jogada especial, que apaga uma sequência e permite jogada convencional em seguida;
4. Verificação do resultado do jogo (vitória ou empate);
5. Ranking dos jogadores;
6. Salvar e recuperar um jogo antes do final da partida;
7. Salvar e recuperar ranking dos jogadores;
8. Modularização adequada considerando possibilidades de upgrade futuro;
9. Uso de sons e cores para conceber melhor experiência;
10. Realização de testes;
11. Seguir padrões de boa prática;

2.2. Descrição de alto nível

O programa não possui interface, ofertando apenas interação direta através do terminal onde é exibido o Menu Principal (desenvolvido na função *main()*). Tal funcionamento se reparte em três possíveis escolhas (Figura1-Menu Principal): Continuar; Novo Jogo e Ranking. Por trás de cada seleção do menu, ocorre a chamada de *funções* correspondentes, elencadas dentro do código em 3 seções diferentes.

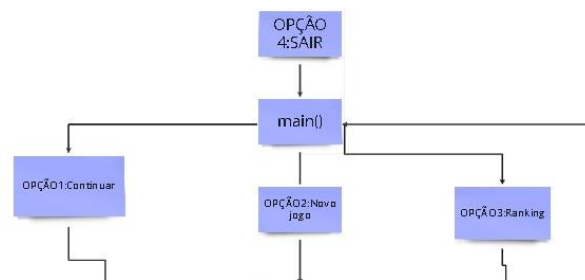


Figura 1. Menu principal

No que concerne à segunda opção do Menu (Figura1-Menu Principal), os jogadores são direcionados à escolha da dificuldade, que definirá o tamanho da tabela a ser utilizado como parâmetro na função *jogada(tamanho)*. Dentro desta são solicitadas as funções e parâmetros: *tabela(tamanho); jogadores(); especial(tabela_jogo, tamanho, casas, jogadas, tabelas); atualiza_tabela(tabela, tabela_jogo, tabela_verificar, tamanho, casa, num, num_verif); save(dados); verificar_jogada(tabela_verificar, tamanho, objetivo, jogador); verificar_objetivo(sequencia_objetivo)*. Após sua execução, o software retorna a função *main()*.

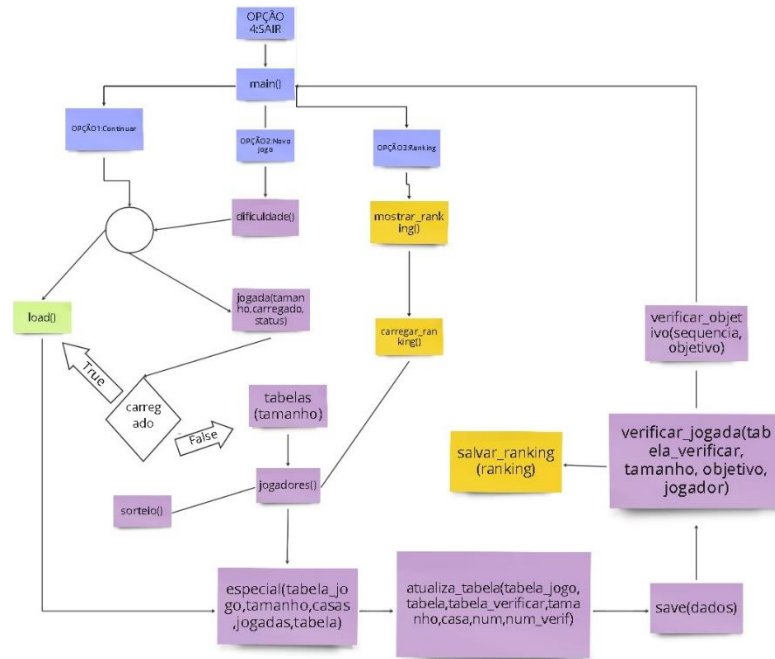


Figura 2. Diagrama de blocos “Novo jogo”

Em se tratando da sequência das funções ordenadas no último parágrafo, bem como evidenciadas acima (Figura 2. Diagrama de blocos “Novo jogo”), segue uma breve descrição de ocorrências que correspondem às suas chamadas:

1. Seleção de dificuldade;
2. Exibição de regras de jogo e orientação de salvamento;
3. Solicitação de nome de jogador;
 - a. Fornecimento dos objetivos;
4. Escolha de habilidade especial;
5. Exibição da tabela e entrada de casa e número (e/ou especial) de modo alternado
6. Anúncio de empate ou vitória
7. Menu principal ou (saída)

O estado do jogo só é salvo caso o usuário digite S no campo de casa - mencionado na etapa 2- com auxílio da função *save(dados)*. Nessa situação, os dados de variáveis e tabelas são alocados para um dicionário de mesmo nome, e o software ignora a execução dos itens 6 e 7 acima, retornando ao Menu principal. Para prosseguir a partir da última partida, é preciso selecionar a opção “Continuar” (Figura 2. Diagrama de blocos “Novo jogo”).

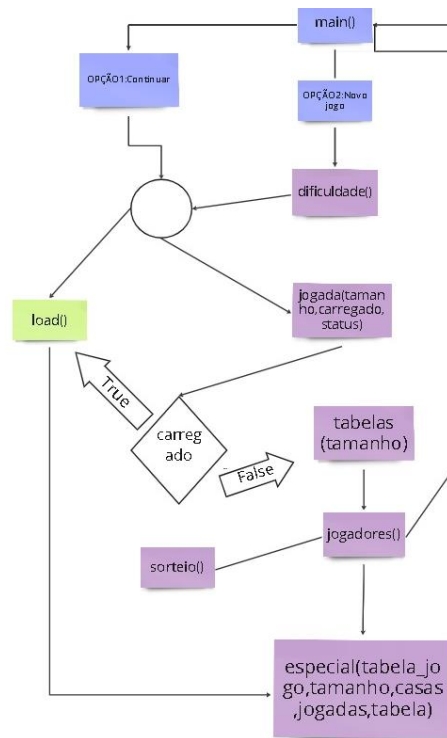
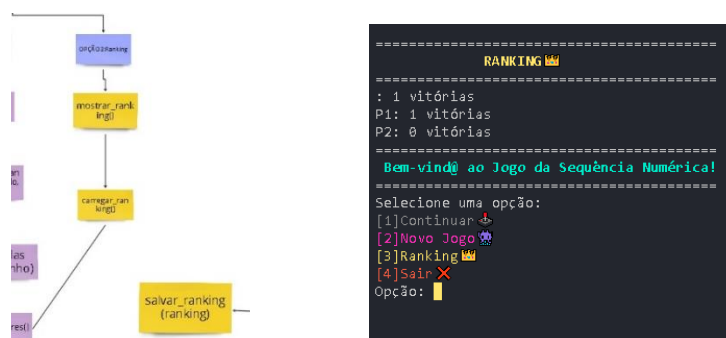


Figura 3. “Continuar”

Seguindo por esse caminho-indicado por valor *True* atribuído à “*carregado*”(variável de valor *booleano*)-as variáveis e estruturas de dados são carregadas através da função *load()*, evitando repetição das funções *tabela(tamanho)* e *jogadores()*. A função jogada é executada a partir da chamada da função *especial()*, após atualizar o jogador quanto ao status da partida passada (objetivos de cada um); e antes de exibir a tabela visual conforme últimas jogadas. A partir deste ponto, o prosseguimento do jogo se faz da mesma maneira que quando é escolhida a opção “Novo jogo”, descrita anteriormente e evidenciada acima (Figura 3. “Continuar”) e na Figura 2. Diagrama de blocos “Novo jogo”



```

=====
RANKING 🏆
=====
: 1 vitórias
P1: 1 vitórias
P2: 0 vitórias
=====
 Bem-vind@ ao Jogo da Sequência Numérica!
=====
Selecione uma opção:
[1]Continuar ➡
[2]Novo Jogo 🎮
[3]Ranking 🏆
[4]Sair ✖
Opção: 3
  
```

Figura 4.Ranking

Ao selecionar a opção 3, é chamada a função *mostrar ranking()*, que lê e formata o dicionário *ranking* criado caso não hajam partidas prévias, ou carregado com auxílio da função *carregar_ranking()*. O mesmo é atualizado através da função *salvar_ranking()*, chamada em *jogada(tamanho)*.

A opção 4 apenas sai de um *loop* com *while*, e não retorna ao Menu.

2.3. Ordem de Codificação

A ordem de codificação foi norteadada pelo menu. A partir dele foram listadas as funções necessárias a serem desenvolvidas. Inicialmente foi pensado nas definições prévias(sorteio,jogadores), seguido da “gameplay”, que envolve criação das tabelas, jogabilidade,inclusão de verificações de entrada e de condições de jogo e jogada especial. Posteriormente, foi desenvolvida a seção “ranking e save” que trata da criação,manipulação e salvamento de arquivos. A função save não foi completamente concebida no final,uma vez que algumas de suas definições dialogam com o estado do jogo (*jogada(tamanho)*).

O produto foi construido na linguagem *Python*, na versão 3.12, em um notebook com sistema operacional Windows 10,através da ferramenta Visual Studio Code.

3. Resultados e Discussões

3.1 Manual de uso

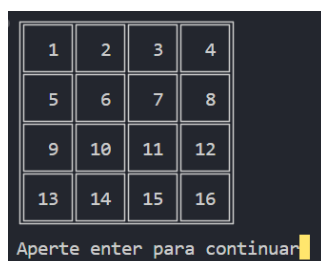
Para utilizar o programa,é necessário um interpretador do Python instalado no dispositivo e da biblioteca *tabulate*,que pode ser instalada utilizando o comando *pip install tabulate* no prompt de comando (*cmd*).

Ao iniciar a execução,após abertura do arquivo .py, o usuário deve selecionar uma opção no Menu (Figura 1.Menu principal).Caso a primeira seja escolhida,existem duas possibilidades:

- i. Não há jogo salvo
- ii. Há jogo salvo

Para a primeira ocorrência,é mostrada a mensagem “Nenhum jogo salvo encontrado. Aperte ENTER para voltar ao menu principal.”,e é preciso optar pelo “2-Novo jogo” (Figura 2. Diagrama de blocos “Novo jogo”).Em seguida o usuário deve seguir os seguintes passos:

1. Apertar a tecla do número que corresponde à dificuldade (1-Fácil,2-Médio,3-Difícil)
2. Ler as regras e apertar ENTER
3. Digitar nome do jogador 1,apertar ENTER para ver objetivo,apertar ENTER para passar para o próximo
4. Realizar novamente o passo 3 para jogador 2
5. Apertar a tecla S seguida de ENTER caso desejar modo com habilidade especial,se não,apenas enter



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Aperte enter para continuar

Figura 5. Tabela nível médio

Após seguir este passo a passo, é exibida uma tabela com o número de casas enumeradas (Figura 5. Tabela nível médio), de acordo com a escolha de dificuldade, o usuário deve apertar ENTER para prosseguir para escolha de casa, onde digita um número existente na tabela e pressiona enter novamente, seguindo o mesmo passo para número que deseja inserir na casa. Ambas as entradas devem ser de números inteiros, caso contrário o usuário é questionado novamente.

```
=====
REGRAS
=====
-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 baix
o) 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.

-Para salvar o jogo aperte 'S' no campo 'casa'

Aperte ENTER para continuar
```

Figura 6. Regras

O usuário 2 deve realizar a mesma ação. Esse *loop* finaliza ao cumprir um dos objetivos ou ao ocupar todas as casas (Figura 6. Regras). Ao chegar em um ganhador, é exibida uma mensagem no formato “O jogador (jogador vencedor) ganhou! Formando a sequência [sequência] cumprindo o objetivo de formar números (OBJETIVO)” e retorna ao menu principal. Para uso da habilidade especial, o usuário deve digitar E seguido de ENTER no campo de casa, se modo de jogo habilitado anteriormente.

1	2	

Turno do jogador 1: Escolha uma casa entre 1 e 9: E

Deseja apagar:

1-Linha

2-Coluna

3-Diagonal

Deseja apagar:

1-Linha 1

2-Linha 2

3-Linha 3

4-Linha 4

5-Linha 5

Turno do jogador 1: Escolha uma casa entre 1

Figura 7. Especial

É preciso escolher onde a sequência que deseja deletar se localiza. Se selecionada uma linha ou coluna inexistente, a opção retorna até que uma válida seja indicada. O jogador que optou por apagar realiza sua jogada convencional (Figura 7. Especial).

```
=====
STATUS
=====
O status da partida passada é:



|  |   |   |
|--|---|---|
|  | 2 | 3 |
|  |   |   |
|  |   |   |



Aperte enter para continuar
```

Figura 8. Status

Caso haja jogo salvo (ii), basta repetir o processo descrito no início do manual de uso-passos 3 a 5- com a diferença de que não há necessidade de inserir o nome,e no lugar tabela enumerada (Figura 5. Tabela enumerada),fica a representada na Figura 8.Status. O último jogo fica salvo como *savegame.pkl*.

No que se trata da escolha de opção 3 (Figura 4. Ranking),é exibido ao usuário um Ranking com nomes de players anteriores,ao lado da quantidade de vitórias em uma mesma linha e ordenadas de modo crescente Logo após, retorna ao menu principal.O arquivo com o ranking é salvo com nome de *ranking.pickle*.

Já a opção 4-Sair (Figura 2. Diagrama de blocos “Novo jogo”) fecha o programa e não exhibe mais o Menu Principal.

3.2 Testes e erros

A respeito dos testes, foram direcionados principalmente ao preenchimento da tabela,visando verificar se as saídas correspodiam aos números fornecidos, as sequências retiradas da mesma eram processadas de modo correto e se o jogo retorna quando elas se encaixam em algum objetivo.Além disso foram analisados se os dados de ranking e save eram realmente armazenados no dicionário e os arquivos criados,ou lidos caso já existissem.

4. Conclusão

Tendo em vista a solução proposta, é possível concluir que o produto cumpre com os requisitos, na medida em que segue todas as orientações especificadas no tópico 2.1- Definição de Requisitos. Além dessas especificações,o jogo possui sonorização simples,aplicada com auxílio da biblioteca nativa *winsound*, bem como detalhes de estilização como cores e emojis no Menu principal.

Porém,ainda se fazem necessárias modificações, objetivando garantir uma melhor experiência ao usuário:

1. Nova ambientação por meio de interfaces gráficas;
2. Jogabilidade:
 - a. Menu com acesso permitido durante todo o jogo;
 - b. Consideração de uma sequência decrescente e diagonal na vitória,para objetivos pares e ímpares.

5.Referências

FOUNDATION, P. S. Python 3.12.3 documentation. 2024. Disponível em: <https://docs.python.org/3/>. Acesso em: 9 jul. 2024.Foundation, P. S. (2024). Python 3.12.3 documentation. <https://docs.python.org/3/>.

LINUX CONSOLE. Um algoritmo geral para salvar e carregar dados em Python. Disponível em: <https://pt.linux-console.net/?p=25529#:~:text=Um%20algoritmo%20geral%20para%20salvar%20e%20carregar%20dados,a%20vari%C3%A1vel%20do%20arquivo%20usando%20pickle.load%20%28file%29.%20>. Acesso em: 9 jul. 2024.Sergey.(2022)Tabulate documentation.[tabulate · PyPI](https://pypi.org/project/tabulate/).

PYPI. Tabulate: Pretty-print tabular data in Python, a library and a command-line utility.
Disponível em: <https://pypi.org/project/tabulate/>. Acesso em: 9 jul. 2024.Linux
Console-net.[Como usar o Pickle para salvar e carregar variáveis em Python? \(linux-console.net\)](#)