

Probabilidades na Matrix: Os tabuleiros das somas ocultas

Ricardo C. Oliveira

Departamento de Ciências Exatas Curso de Engenharia de Computação - Universidade
Estadual de Feira de Santana (UEFS)
Avenida Transnordestina, s/n - Novo Horizonte - CEP 44036-900 - Feira de Santana -
Bahia, Brasil

rickoliver001@hotmail.com

Abstract. *This report aims to describe the construction and development of a board game whose users must guess, through deductions and probabilities, the sums of its rows and columns. In addition, it reports what the project requirements were and how all the difficulties encountered in solving the problem were overcome, using the Python programming language for this purpose. Therefore, it was possible to build the code complying with all the requirements imposed by the system.*

Resumo. *O presente relatório tem como objetivo descrever a construção e desenvolvimento de um jogo de tabuleiros cujos usuários devem adivinhar, por meio de deduções e probabilidades, as somas de suas linhas e colunas. Além disso, relata quais foram os requisitos do projeto e a forma de como foram superadas todas as dificuldades encontradas na resolução da problemática, utilizando da linguagem de programação Python para tal finalidade. Diante disso, foi possível a construção do código acatando todos os requisitos impostos pelo sistema.*

1. Introdução

Redescobrimos a simulação da realidade em que está inserido, Neo imediatamente redireciona suas atenções ao encontro de seus fiéis e leais companheiros. Porém, nessa desafiadora viagem pelos confins da chamada Matrix, encontra diversas provações perigosas impostas pelo seu maior rival, o vírus Agente Smith. Seu inimigo mortal o prende em um labirinto de enigmas, onde é posto em provação por variados desafios.

Logo após passar por quase todos os desafios, Neo se depara com o último deles, a criação do Jogo das Somas Esquecidas. Já na etapa final, com seus poderes totalmente esgotados pelo cansaço, Neo decide procurar ajuda no mundo real, onde a autoridade do Agente Smith não alcança. Almejando resolver por completo esse desafio, Neo pede ajuda aos estudantes de Engenharia de Computação da Universidade Estadual de Feira de Santana.

Nesse contexto, acatando ao pedido d'O Escolhido, foi construído o programa que trata do referente Jogo das Somas Esquecidas, utilizando conceitos de linguagem de programação para tal fim. Assim, Neo conseguirá vencer o seu arqui-inimigo, o Agente Smith.

2. Metodologia

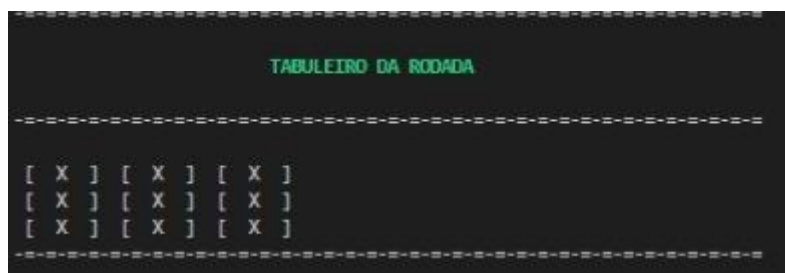
Nas sessões tutoriais que ocorreram no período de construção do código, foram descobertos diversos desafios para a resolução do problema. Inicialmente, os integrantes das sessões tiveram dificuldade em entender o problema em si, tendo que relê-lo diversas vezes para o seu entendimento completo. Para aprimorar o conhecimento sobre o que o problema requisitava, os participantes do trabalho jogaram uma partida do jogo na sessão tutorial, assim, foram descobertas várias questões sobre regras e casos específicos do jogo.

Com o decorrer do tempo, as sessões tornaram-se muito mais produtivas com relação a qual lógica seria implementada para o solucionamento das questões encontradas. Na etapa de descobrimento das funcionalidades que seriam utilizadas, os integrantes sentiram uma certa dificuldade em perceber como elas ajudariam na resolução do problema, visto que, vários conceitos não eram do conhecimento de todos os integrantes. Com a contribuição de cada um nas sessões, o conhecimento acerca das “novas” funcionalidades foi expandido, que por sua vez, gerou como consequência uma variabilidade de ideias para a solução de cada parte do problema.

Contudo, apesar das dificuldades iniciais, todos os integrantes conseguiram absorver bastante experiência e conhecimento das sessões tutoriais. Além do desenvolvimento individual, o entrosamento da equipe também foi aprimorado nesse trajeto, gerando uma maior sinergia para experiências futuras. Ao final do trabalho, cada código apresentava uma particularidade diferente, porém, todos orientados para a mesma finalidade, a resolução do problema. Dito isso, a evolução do conhecimento da equipe foi bastante proveitosa e construtiva.

Nesse contexto, é importante salientar quais foram os requisitos impostos pelo problema em pauta. O Jogo das Somas Esquecidas consiste em um jogo de tabuleiros jogado por dois jogadores, que por sua vez, podem escolher entre dois modos de jogo: com um tabuleiro, ou com dois tabuleiros. O jogo apresenta também três níveis de dificuldade: fácil; médio; e difícil. Para o nível fácil, o tabuleiro no qual os jogadores irão realizar as suas jogadas terá uma área de 3×3 , ou seja, três linhas e três colunas. Caso o nível escolhido seja médio, o tabuleiro terá uma área 4×4 , e para o nível difícil, uma área 5×5 . Além dessas escolhas, o jogo apresenta dois modos de término de partida: por número de rodadas (que deve ser um número ímpar); ou por tabuleiro totalmente revelado.

No princípio, os tabuleiros são preenchidos com números inteiros aleatórios de acordo com um dado intervalo, que por sua vez, varia de acordo com o nível escolhido. Para o nível fácil, o intervalo se configura entre 1 e 30. Para o nível médio, entre 1 e 60, e para o difícil, entre 1 e 100. Para todos os níveis do jogo, não é permitida a repetição desses números no tabuleiro. Os valores dos tabuleiros devem permanecer ocultos aos jogadores, mostrando somente o tabuleiro vazio (Figura 1). As somas de todos os elementos de cada linha devem permanecer ocultos ao lado de cada uma delas, assim como, as somas dos elementos de cada coluna permanecem abaixo.



TABULEIRO DA RODADA								
[X]	[X]	[X]
[X]	[X]	[X]
[X]	[X]	[X]

Figura 1 - Tabuleiro com valores ocultos

No início de cada rodada, cada um dos jogadores deve escolher a linha ou coluna no tabuleiro geral (ou no seu tabuleiro individual) em que quer chutar o somatório de seus elementos. Logo após essa escolha, inserir o valor de seu chute. A partir disso, pontua o jogador que mais se aproximou da soma da linha ou coluna escolhida por ele. O jogador que pontuou terá uma ou mais casas reveladas no tabuleiro de acordo com a precisão de seu chute (Figura 2). Em caso de empate, os dois terão uma ou mais casas reveladas.



TABULEIRO DA RODADA								
[X]	[X]	[X]
[11]	[X]	[X]
[X]	[X]	[X]

Figura 2 - Tabuleiro com uma casa revelada

Nessa configuração, as casas a serem reveladas vão depender do chute do jogador que pontuou (ou dos dois em caso de empate). Caso o chute seja maior que a soma da linha ou da coluna escolhida, a casa a ser revelada será a que possuir o maior número da localização escolhida. Caso o chute seja menor que a soma, a casa revelada será a que possuir o menor número naquela localização. Além disso, caso o chute seja exatamente a soma, a linha ou coluna da escolha do jogador será inteiramente revelada. Os jogadores recebem a sua pontuação de acordo com o número de casas reveladas no tabuleiro. Ao ser atingido o critério de encerramento da partida, o jogador vencedor será consequentemente o que obteve a maior pontuação, ou seja, o maior número de casas reveladas.

A cada rodada, deve ser apresentado aos jogadores, os tabuleiros em que serão realizadas as jogadas. Além disso, um placar parcial contendo o número de casas reveladas pelos dois até aquela rodada e um histórico contendo informações das rodadas anteriores, como os seus chutes e se eles foram maiores, menores ou exatamente a soma das localizações escolhidas. Paralelamente, as jogadas realizadas devem ser validadas para que os jogadores escolham sempre linhas e colunas válidas na configuração escolhida por eles.

Diante disso, é relevante apresentar a forma de como o programa foi construído e quais funções estão presentes nele. O código em sua composição final foi separado em 4 módulos além do arquivo principal. Cada módulo possui uma finalidade, sendo eles: exibições, validações, escolhas do jogador e processamento. No arquivo principal são

encontradas sequências de chamadas de funções presentes nos módulos, para a manutenção do funcionamento das partidas no jogo. As funções utilizadas no programa foram divididas nesses módulos por funcionalidades específicas, assim, o tratamento de erros presentes no código foi facilitado, já que era fácil identificar quais áreas estavam com anormalidades a partir dos módulos.

Sob uma perspectiva geral, o código principal se divide em três partes: antes do início da partida, durante e ao final dela. Todas as ações são realizadas por meio de funções próprias.

Primordialmente, antes do início da partida, o programa possui um menu de escolhas para o conhecimento da configuração das partidas. Para cada caso de configuração existe uma área específica referente àquele estilo de jogo. Após essas escolhas, o fluxo do código é direcionado para a formação das estruturas que serão utilizadas no jogo, como a formação dos tabuleiros e a inserção dos números aleatórios nas casas a serem reveladas. É importante salientar que, como o jogo é constituído por tabuleiros, o código possui foco na utilização de matrizes, portanto, utilizou-se esse tipo de estrutura de dados para a construção dos mesmos.

Após a formação de tais estruturas, o fluxo do código se direciona para o início da partida em si. Diante disso, o programa recebe as entradas dos jogadores para a comparação de qual deles se aproximou mais da soma da localização escolhida, ou seja, qual jogador pontuou. Seguindo adiante, o programa segue para a lógica de descobrir se o chute do jogador vencedor foi maior ou menor que a soma, ou exatamente ela. Todas as informações do vencedor da rodada vão sendo armazenadas, dados como: qual jogador pontuou; localização de onde chutou; valor do chute; se o seu chute foi maior ou menor que a soma, ou exatamente ela; etc. Após isso, o programa se dirige ao bloco de substituições dos valores ocultos (elementos presentes nos tabuleiros apresentados aos jogadores) pelos valores reais (elementos presentes nos tabuleiros ocultos aos jogadores). Nessa etapa, ocorre a substituição desses valores de acordo com as informações armazenadas anteriormente, pois, nelas vão estar contidas as instruções de quais números vão ser substituídos nos tabuleiros da partida.

Logo após isso, a contagem de pontos é realizada de acordo com o número de casas que foram reveladas no tabuleiro apresentado aos jogadores. Com as informações que foram armazenadas, é feito o histórico das rodadas (Figura 3), que por sua vez é apresentado a cada rodada, exibindo essas informações. O programa se repete nessa configuração, por meio de laços de repetição, até que o critério de término escolhido seja atingido.

```
-----
                                HISTÓRICO DAS RODADAS
-----
RODADA: 1
- JOGADOR 1 chutou 20 para a L1 e foi MENOR que a soma!
- JOGADOR 2 chutou 30 para a L2 e foi MAIOR que a soma!
RODADA: 2
- JOGADOR 1 chutou 40 para a L3 e foi MENOR que a soma!
- JOGADOR 2 chutou 50 para a L3 e foi MENOR que a soma!
-----
```

Figura 3 - Histórico de jogadas

Ao final da partida, o fluxo do código se direciona para a apresentação dos tabuleiros com suas respectivas somas, assim como, a exibição de qual jogador foi vitorioso na partida. Essas duas ações são realizadas por meio de *print's* configurados de acordo com o fim da partida.

Vale ressaltar que as validações para as entradas dos jogadores no programa são implementadas só quando há interação com os usuários. No presente código, as validações são utilizadas no menu de escolhas inicial, no número de rodadas (caso a escolha de término seja essa), e nos chutes realizados pelos jogadores, impossibilitando-os de escolherem alguma opção inválida.

Outro ponto importante para o entendimento do desenvolvimento do código é sua ordem de codificação. Em primeiro plano, a atenção foi voltada para a construção do menu de escolhas da configuração das partidas. Assim, o foco foi se reduzindo para configurações mais restritas de acordo com essas escolhas. Após isso, o foco foi redirecionado para a formação das estruturas que seriam utilizadas no jogo, como os tabuleiros.

Logo após, foi importante dar ênfase em como definir qual jogador se aproximou mais da soma e se seu chute foi maior, menor ou exato. Com isso, o código teria o conhecimento de qual elemento seria substituído no tabuleiro com os valores ocultos. A partir disso, o foco foi concentrado em como as substituições ocorreriam nos tabuleiros, sendo criada uma lógica para tal fim. Posteriormente, foi necessário implementar a lógica de pontuação dos jogadores para o encerramento da partida funcionar por completo. A partir disso, foi construído o histórico utilizando todas as informações do vencedor armazenadas durante a partida.

Mais adiante, foram criadas as exibições do tabuleiro com as somas ao lado de cada linha e abaixo de cada coluna, além da apresentação de qual jogador foi vitorioso na partida. Ao final da construção de todas as funções referentes a essas ações, foram implementadas todas as validações de entradas realizadas pelos jogadores, para a prevenção de erros no programa.

Alguns casos especiais de pontuação foram deixados para a etapa final do código, pois, em sua maioria, não influenciavam em seu funcionamento parcial. Portanto, na etapa final de codificação, a atenção foi direcionada para esses casos específicos, como empates e pontos duplicados no tabuleiro.

Para o desenvolvimento das soluções implementadas no programa, foi utilizado o sistema operacional Windows 10, juntamente com a linguagem de programação Python manipulada através do editor de código-fonte Visual Studio Code.

3. Resultados e Discussões

O programa deve ser utilizado de acordo com as opções que são apresentadas aos seus usuários. Inicialmente, é preciso inserir como os jogadores querem ser referenciados no jogo, podendo ser quaisquer caracteres de sua escolha. Logo após, deve-se escolher a configuração da partida a ser jogada dentre as opções apresentadas no menu. Adiante, independente da opção de término escolhida, as jogadas a serem inseridas pelos jogadores no decorrer da partida serão apenas duas: localização e chute. Inicialmente, deve-se escolher a localização em que vai ser realizado o chute, por exemplo: "L1". Após a escolha da localização no tabuleiro, é necessário realizar o chute para aquela

linha ou coluna. Assim, o programa irá processar as escolhas e chutes dos jogadores para revelar quem foi o vencedor da rodada, e, além disso, revelar casas nos tabuleiros de acordo com essas respostas. O código segue esse fluxo até a partida terminar, revelando quem foi o vencedor da partida.

Para uma experiência fluida com o programa, é importante definir quais são as entradas esperadas de seus utilizadores. Inicialmente, caso os usuários não insiram nenhum caractere no campo dos nomes dos jogadores, o código apresentará uma mensagem de erro informando que essa ação é necessária.

Já no menu de escolhas, caso não sejam escolhidas opções válidas no mesmo, mensagens de erro serão exibidas até que as entradas sejam aceitas. Caso o modo de término da partida escolhido no menu seja o por número de rodadas, o valor a ser inserido deve ser um número inteiro positivo ímpar (um dos requisitos solicitados pelo problema). Caso essa condição não seja atendida, o programa apresenta uma mensagem de erro até que ela seja acatada. Contudo, embora nesse caso o critério de encerramento seja por número de rodadas, há um limite máximo de rodadas que pode ser inserido. No código, esse limite é de 99 rodadas (particularidade individual discutida em sessão tutorial). Porém, mesmo que o número de rodadas ainda não tenha sido atingido, se os tabuleiros estiverem completos nesse trajeto, a partida será finalizada.

Na etapa de escolha da localização na qual será realizado o chute, é necessário que os jogadores escolham opções válidas nas configurações do tabuleiro escolhido, caso contrário, mensagens de erro serão apresentadas até que sejam escolhidas as opções possíveis. Além disso, é importante salientar que, as jogadas serão somente possíveis em linhas e colunas que ainda não foram totalmente reveladas na partida. Essa decisão foi tomada a partir de discussões nas sessões tutoriais realizadas durante o período de confecção do código.

Já na etapa de chute dos jogadores, os valores inseridos devem ser somente números inteiros maiores ou iguais a zero. Caso essa condição não seja acatada, o programa novamente apresenta uma mensagem de erro até que seja inserida uma entrada válida pelos usuários.

Atribuindo a mesma importância das entradas, as saídas possuem papel fundamental no entendimento das partidas do jogo ao decorrer das rodadas. Em sua maioria, além das mensagens de erro referenciadas anteriormente, as saídas presentes no programa são exibições das informações captadas durante as rodadas da partida. Após as escolhas serem realizadas pelos jogadores por meio do menu inicial, os tabuleiros (ou somente um, para o modo com um tabuleiro) da partida são exibidos. Logo após os chutes serem realizados, o programa realiza o processamento dessas informações e exibe quem foi o jogador que pontuou na rodada. A partir dessas informações, o código realiza essa ação tomando como parâmetro a aproximação dos chutes dos jogadores, com relação a soma da linha ou coluna escolhida por eles. Em seguida, é exibido o placar da rodada, que é processado a partir do número de casas reveladas no tabuleiro.

Após essas saídas, o histórico é apresentado aos jogadores, contendo as informações captadas e armazenadas durante as rodadas que já aconteceram até o momento. Seguindo o fluxo de processamento do código, são exibidos os tabuleiros com os valores substituídos. Esses números são substituídos de acordo com a precisão

do chute dos jogadores, como explicado anteriormente. A partir disso, os jogadores podem efetuar as demais jogadas das rodadas restantes.

Ao final da partida, são exibidos os tabuleiros com as somas ao lado de cada linha e abaixo de cada coluna. Além disso, também é apresentado qual jogador venceu a partida, ou, em caso de empate, os dois jogadores. Concluindo assim, as saídas presentes no código.

Para compreender o desenvolvimento do código no decorrer do período de sua confecção, é de suma importância conhecer quais foram os testes efetuados nele. De início, os testes se centravam nas escolhas das opções presentes no menu inicial, tendo que analisar se as configurações feitas por meio dele estavam corretas. Posteriormente, foi necessário realizar testes na formação dos tabuleiros com a utilização de matrizes, já que o conhecimento sobre essa estrutura de dados ainda era básico. Os testes realizados nessa fase se configuravam em inserir os números aleatórios, sem sua repetição, na matriz dos tabuleiros, além de criar o número de linhas e colunas adequado para cada nível de jogo.

Logo após os testes realizados na formação dos tabuleiros, o foco foi direcionado em definir qual jogador se aproximava mais da soma. Com isso, os testes realizados consistiram em inserir números iguais e diferentes nos chutes, almejando prever quais erros ocorreriam ao fazer isso.

Seguindo com a construção do código, os testes passaram a ser efetuados na etapa das substituições dos valores nos tabuleiros. Nesses testes, eram analisados se as casas reveladas condiziam com as respostas dos jogadores. Diante disso, todas as medidas foram tomadas para que fossem substituídos os valores corretos, isso foi possível através da colaboração do integrante Pedro Henrique nas sessões tutoriais, portanto, faz-se necessário dar-lhe o devido crédito no presente relatório.

Após os testes efetuados nas substituições, a atenção foi direcionada para a lógica de pontuação dos jogadores, onde foi necessário criar uma configuração para esse quesito. Nesse contexto, os casos de testes consistiam em inserir exatamente a soma das linhas ou colunas, assim como, acertar somente duas casas no tabuleiro, para observar qual seria a pontuação recebida nesses casos. Com o decorrer dos testes realizados, foi possível a implementação de um sistema de pontuações condizente com os resultados obtidos no jogo.

Durante esse trajeto, houveram diversos erros na construção do código, por isso, é importante apresentá-los para uma melhor percepção do seu desenvolvimento e otimização.

Inicialmente, o código apresentava diversos erros de lógica provenientes da construção dos tabuleiros. Geralmente, esses erros eram do tipo “*ValueError*” e “*IndexError*”, pois a manipulação para a formação dos tabuleiros era totalmente incorreta e falha. Porém, essas adversidades foram sanadas para o desenvolvimento do programa.

Logo após esses percalços, já na etapa intermediária de confecção do código, a maioria dos erros eram provenientes das substituições dos valores reais nos tabuleiros apresentados aos jogadores. Inicialmente, os valores substituídos nos tabuleiros apresentavam incongruência. Diante disso, os erros mais comuns encontrados eram os

do tipo “*IndexError*”, pois a manipulação dessas substituições ocorriam de forma irregular, substituindo valores errados e “quebrando” o código. Contudo, logo após discussões de ideias nas sessões tutoriais, foram implementadas lógicas que contornavam esses erros provenientes das substituições irregulares.

Já na etapa final, restavam apenas os erros de pontuações irregulares dos jogadores. Inicialmente, os jogadores recebiam somente um ponto por rodada, independente de quantas casas fossem reveladas. Nesse caso, ocorriam duas anormalidades no código: a contagem irregular de pontos e o não atendimento das condições de encerramento das partidas em sua totalidade. Ao obter o conhecimento desses erros, foi implementado um sistema de pontuação adequado que contabilizava corretamente os pontos recebidos, incluindo pontos duplicados em casos de empate. Nesses casos específicos, somente uma casa é revelada, porém, os dois jogadores recebem pontuação (discussão realizada em sessões tutoriais), visando sanar essa adversidade, o novo sistema de pontuação foi criado e implementado. Em síntese, o problema foi sanado para um melhor funcionamento do código, solucionando assim, os erros encontrados no período de confecção do mesmo.

4. Conclusão

Dado o presente problema, foi possível desenvolver e aprender variadas ferramentas e funcionalidades da linguagem Python. Todos os requisitos foram cumpridos adequadamente ao decorrer do desenvolvimento do projeto.

Em síntese, as áreas que podem receber melhora no código são as partes visuais apresentadas no terminal, e também a otimização do programa em si, promovendo uma melhor rapidez no processamento de dados realizado. Funcionalidades e implementações a mais realizadas no código foram: a inserção de cores no terminal e a exibição dos tabuleiros resultantes da partida com suas respectivas somas. Essas funcionalidades extras foram implementadas para tornar o código mais completo, e, além disso, deixá-lo mais intuitivo visualmente para os usuários.

O trajeto realizado tanto individualmente quanto pela equipe foi totalmente proveitoso para o conhecimento de novas funções presentes na linguagem de programação utilizada. Tornando assim, a experiência construtiva e elucidativa.

5. Referências

Exercício Python #086 - Matriz em Python, Gustavo Guanabara, Curso em Vídeo, publicado em 01 de agosto de 2018 na plataforma YouTube. Disponível em: [Exercício Python #086 - Matriz em Python - YouTube](#)

Python #11 - Listas, João Paulo Just Peixoto, Professor Just, publicado em 18 de maio de 2020 na plataforma YouTube. Disponível em: [Python #11 - Listas - YouTube](#)

Curso Python #11 - Cores no Terminal, Gustavo Guanabara, Curso em Vídeo, publicado em 16 de agosto de 2017 na plataforma YouTube. Disponível em: [Curso Python #11 - Cores no Terminal - YouTube](#)