TETRIS GAME

Felipe Gomes da Silva

[1]Engenharia da Computação - Universidade Estadual de Feira de Santana (UEFS)
Feira de Santana – BA, Brasil – 44036-900

felipegoomes8910@gmail.com

Abstract. *The large game publisher Blizzard Entertainment has opened a selection process for an internship at its company. This consists of developing a new version of the game Tetris, where the user must manipulate pieces of various shapes that fall from the top of a board and organize them in a way that completes lines and thus eliminates them to earn points. This article aims to describe the entire game creation process using the Python language.*

Resumo. *A Blizzard Entertainment grande editora de jogos, abriu um processo seletivo para estágio em sua empresa. Este consiste no desenvolvimento de uma nova versão do jogo Tetris, onde o usuário deve manipular peças de vários formatos que caem do topo de um tabuleiro, organizar de uma forma que completem linhas e assim eliminá-las para ganhar pontos. Esse artigo tem por objetivo descrever todo o processo de criação do jogo utilizando a linguagem Python.*

1. Introdução

O *Tetris*, jogo criado pelo matemático russo Alexey Pajitnov, foi feito em meio a Guerra Fria no Centro de Computação da Academia Russa de Ciências e inspirado num jogo grego chamado *Pentaminoes*, no qual as peças são formadas por cinco quadrados, já no *Tetris* as peças possuem apenas quatro quadrados, daí vindo a origem do seu nome que em grego significa quatro. O jogo veio se popularizar quando o designer Henk Rogers, o lançou no Japão para um aparelho da Nintendo, onde veio a vender três milhões de exemplares. Desde 1988 até o ano atual de 2024, o *Tetris* é sucesso de vendas, com mais de 520 milhões de exemplares vendidos em todo o mundo, segundo o site oficial do *Tetris* e conquista pessoas de todas as idades. O jogo pode parecer simples mas a estratégia e velocidade pode fazer toda a diferença durante a *gameplay.*

A proposta do jogo é fazer com que os usuários girem, movam e encaixem estrategicamente os *Tetriminos,* nome dado as peças do *Tetris*, que caem de uma matriz de 20 linhas e 10 colunas em uma velocidade crescente. O jogador deve tentar limpar linhas horizontais de blocos e acumular o máximo de pontos possíveis, mas se os *Tetriminos* ultrapassarem o limite no topo da matriz, o jogo acaba.

Diante disso, a proposta desse relatório é apresentar as fases de criação do jogo e a maneira como ele funciona. Ao decorrer dos tópicos será apresentada a estrutura do código feito na linguagem do *Python* e como ele funciona para que o usuário tenha uma experiência divertida.

2. Metodologia

2.1. Desenvolvimento nas sessões

Para a criação do projeto foi necessário fazer sessões com a turma de MI - Algoritmos. Diante da leitura do problema questões foram levantadas: 1) Como seria feita a modularização do código; 2) Como seriam colocados os blocos na matriz e como seria feito a eliminação de linhas; 3) Como mover os blocos na matriz; 4) A tabela ASCII poderia ser utilizada?.

Na primeira sessão foram decididos como funcionaria o código teste, a maneira como seria aplicado o bônus de linhas simultâneas eliminadas e no momento em que as linhas fossem eliminadas, uma nova linha vazia seria adicionada no topo pra preencher o local dessa linha removida.

O problema inicial foi majoritariamente a falta de conhecimento sobre matrizes e definições, algo que seria utilizado pra criação da tabela, peças e para funcionamento do código do jogo. Com o decorrer das sessões e realizando pesquisas, aspecto colocado como meta na primeira sessão, o desenvolvimento de cada componente foi melhorando.

Foi decidido nos encontros seguintes como cada peça deveria ficar após a rotação e como limitar o tabuleiro para que as peças não saíssem da matriz, algo que poderia ocasionar erros, assim afetando o funcionamento do código. Nos foi informado, então, a necessidade de implantação de uma bomba, que após o contato deveria explodir e apagar as peças ao seu redor, tudo dentro de um raio de alcance de uma matriz 3x3.

2.2. Definição de requisitos/funcionalidades

Os requisitos para rodar esse jogo é ter a versão *Python* 3.12.7 instalada em seu dispositivo e as bibliotecas curses, time e random instaladas da maneira correta (a forma de como instalar essas bibliotecas está no manual de instruções, item 3.1 do anexo III.) O primeiro passo foi criar a matriz que seria o tabuleiro dentro de uma função e importar a biblioteca *curses*, usada em grande parte do código pois ajuda a exibir e faz com que ele funcione de forma fluida. Após a criação da matriz, foi necessário definir as peças que seriam usadas no jogo e para isso a fim de representá-las, utilizaram-se quadrados azuis que foram armazenados dentro de listas de listas.

Para que as peças pudessem descer de forma aleatória do topo do tabuleiro, foi necessário a utilização da biblioteca *random* e para controlar a velocidade de descida foi necessária a biblioteca *time,* foram criadas definições para cada funcionalidade do jogo, desde a movimentação até a fixação de cada peça no tabuleiro.

Antes do looping principal do jogo, existe um *while* que gera o menu do *Tetris*, que dá como opção ao usuário iniciar um novo jogo ou encerrá-lo ali mesmo, nele existe uma verificação de erro, caso seja inserido um número ou algo diferente de 1 e 2, é exibida a mensagem: 'OPÇÃO INVÁLIDA' e pede para que sejam digitados os números corretos. Clicando 1, o jogo é inicializado por meio da função play_nojogo(), que puxa dois *loopings* construídos dentro dela, um definido como *while not* fim, que 'chama' as peças para o tabuleiro, verifica se as peças chegaram ao topo para assim definir o fim de jogo, a explosão da peça bomba e outro *looping* chamado *while not* parar que nele estão as funcionalidades do jogo e as maneiras de manipulação das peças do Tetris, logo após

esta tem um *if* que inicializa da maneira correta as funções do código principal, sem ela o código não funciona de maneira alguma.

Por fim, está a condicional que caso o usuário digite 2, exibe uma mensagem de finalização do código.

2.3. Ordem de Codificação

A ordem de codificação foi dividida em 5 blocos:

1 - Bloco de definição e criação de matrizes e peças: foram importadas as bibliotecas *curses, random* e *time.* Logo após isso, foi definido o tamanho do tabuleiro utilizando a *def* criar_tabuleiro, na qual é criada uma matriz de 20x10. Abaixo dessa definição foram definidos os *Tetriminos,* armazenando cada peça em listas e juntando todas essas em outra lista, chamada pedras que será utilizada na *def* pedra_aleatoria, que trabalha junto com a função *choice* da biblioteca *random* para puxar os *Tetriminos* de maneira aleatória no jogo. Para finalizar esse bloco está a peça bomba, que foi necessário utilizar uma função de tratamento de erro (*try* e *except*) para evitar problemas durante o código.

2 - Bloco de exibição: nele estão localizadas as funções aparecer_tab, que exibe um novo tabuleiro e apaga a cada interação para sempre manter o visor atualizado juntamente com colocar_pedra e limpar_pedra que fazem papéis fundamentais para manter a peça dentro do tabuleiro e atualizam cada manipulação de *Tetriminos*. Essas são essenciais para o funcionamento correto dos outros blocos.

3 – Bloco de movimentação e fixação: nesse conjunto de definições estão localizadas a *def* descer, que mantém o movimento de descida constante até a peça encontrar o fim do tabuleiro ou outra peça, fixação esta que foi definida na *def* colisao. Para rotacionar foi utilizada a função *zip*, que compacta todas as peças e além dela, o *reversed* que auxilia na rotação de cada peça. A última *def* desse bloco se chama mover, ela verifica se é possível movimentar a peça dentro dos limites do tabuleiro.

4 – Bloco de pontuação e dificuldade: a primeira definição desse bloco é a *def* eliminar_linhas, que verifica as linhas completas pelo usuário e assim as elimina.

Após apagar a(s) linha(as) do tabuleiro ela adiciona a mesma quantidade que foi eliminadas no topo, assim mantendo o tabuleiro inteiro. A última função desse bloco, def nova_vel, modifica a velocidade de caída das peças após o usuário chegar a 800 pontos, criando uma dificuldade de controle e forçando o usuário a utilizar novas estratégias.

5 – No último bloco está o código principal: nele se encontra o menu do jogo, que possui dois *looping while:* um principal e outro para caso o usuário digite uma opção não esperada pelo sistema. Ao digitar '1', o *Tetris* é iniciado por meio do *if* , que chama a função principal onde estão outros dois *while*. O primeiro é o *looping* das peças, nele está a explosão da bomba e o jogo é encerrado caso as peças passem o limite do topo do tabuleiro; o segundo *looping,* localizado dentro do *while not* parar, contém as funções de manipulação dos *Tetriminos*: movimentar para os lados, rotacionar e acelerar descida, teclas quais são capturadas por meio das funções *getch* e *ord*. Finalizando o bloco e executando corretamente o código, está a função *wrapper* do *curses*, sem essa função, o jogo não inicia. Após encerrar o jogo, digitando '2' no menu inicial, é exibida uma mensagem de despedidas ao usuário.

3. Resultados e Discussões

As dificuldades iniciais se basearam na falta de conhecimento de matrizes e modularização de código que, ao decorrer da primeira sessão, foi sendo resolvida com dicas e recomendações de vídeos para aprendizado. Após uma nova leitura do problema, foi possível notar a dúvida de como colocar as peças no tabuleiro e de como manipular cada peça da maneira correta.

Com o auxílio da turma de MI – Algoritmos, foi possível ter ideia de como manipular cada peça utilizando as bibliotecas *numpy* e *curses.* Algo que foi discutido foi o uso da segunda biblioteca citada, pois ela é completa e não precisaria de outra para complementar sua função, quem utilizasse *numpy* teria que instalar outra biblioteca chamado *keyboard* para manipular o movimento da peça e *OS* para atualizar a tela a cada ação. Próximo ao fim do prazo, houveram muitas dificuldades, que serão citadas no item 3.2 de testes e erros.

3.1. Manual de uso/entradas e saídas

Para o perfeito funcionamento desse jogo é necessária a instalação do *Windows-curses* na versão 2.3.3, abrindo o terminal/*CM*D e digitando *pip install Windows-curses*, assim como na imagem 1, além de ter a versão *Python* 3.12.7 instalada e estar com *Capslock* desativado.

Após iniciar o código do jogo, irá aparecer o menu de seleção com duas opções, uma para inicializar um novo jogo e outra para encerrar o programa (exemplo na imagem 2). Digitando 1 o Tetris será iniciado: para movimentar a peça é necessário apenas um clique para cada ação, seja para rotacioná-la ou mover-la para os lados e para baixo. As teclas definidas não devem ser pressionadas pois pode causar *bugs* na exibição da tela.

Abaixo do tabuleiro *Tetris* estão as instruções e alguns alertas para ter uma *gameplay* fluida e divertida. Clicando ‘w’ a peça irá rotacionar da direita para a esquerda em sentido horário, clicando em ‘a’ o usuário pode movimentar o *Tetriminos* para a esquerda, ‘d’ movimentar para direita e ‘s’ faz a peça descer 4 blocos abaixo, simulando uma queda mais rápida. Ao formar uma linha vertical de blocos azuis, ela será eliminada e contará 100 pontos para o usuário, que está sujeito a um bônus caso elimine duas, três ou quatro linhas simultâneas. Para deixar a *gameplay* mais desafiadora, quando a pontuação do jogador atingir 800 pontos, a velocidade de queda das peças aumentará para 0.1s. Caso sejam acumulados *Tetriminos* até o topo do tabuleiro de uma forma que não consiga mais empilha-lós, o jogo será finalizado.

3.2. Imagens

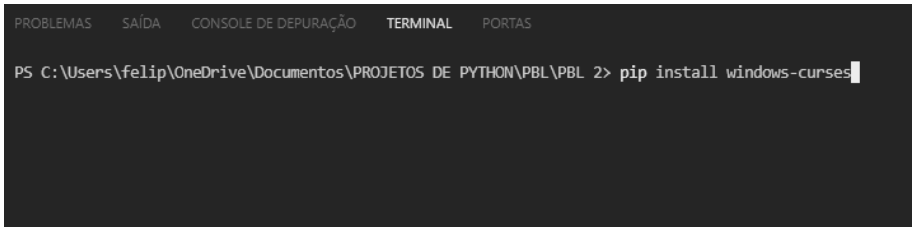

Figure 1. Instalação da biblioteca curses.

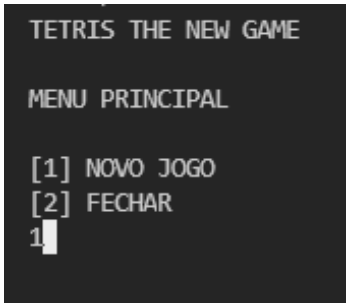

Figure 2. Exibição do tabuleiro.

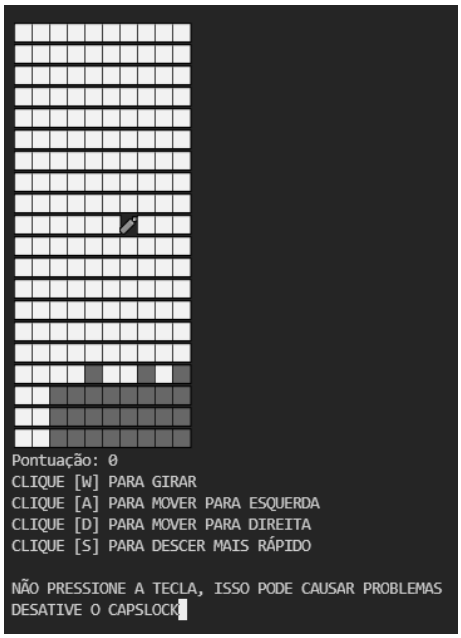

Figure 3. Exibição do tabuleiro.

3.3. Testes e Erros

O programa foi criado na versão 3.12.7, como sugerido nas sessões, por ser a última versão estável e com suporte para a biblioteca usada. Durante os testes, a biblioteca *curses* apresentou problemas, pois ela foi criada para o sistema operacional *Linux*, no entanto a versão utilizada nesse código foi a adaptação para *Windows*. No momento da instalação foi necessário reinstalar o *Python* pelo motivo da biblioteca não estar encontrando a IDE, havendo, portanto, a inserção no local correto, sendo possível a execução da biblioteca.

O programa apresenta um erro quando a tecla é pressionada: o jogo trava e retorna após alguns segundos, caso o usuário faça outros movimentos enquanto estiver travado, eles serão realizados com *delay*. Inicialmente as peças estavam atravessando ou ficando presas a matriz, atrapalhando e causando conflito no tabuleiro, por isso foi necessário criar uma verificação de limites, fazendo com que a peça possa ficar dentro da do tabuleiro.

Para a bomba foram testadas 3 formas lógicas, tanto para explosão tanto para gerar sua matriz, mas todas apresentavam um erro e foi necessário utilizar *try e except* e colocar o bloco que faria ela estourar dentro do *looping* principal, conforme imagem 4 do item 3.2. O jogo deve ser executado com terminal em tela cheia, para que o jogo rode da

maneira correta pelo *curses*. Caso apareça a mensagem de erro da imagem 5 do item 3.2, o usuário deve abrir o terminal da maneira correta e executar novamente.

```
posicionamento_y = 0 #Começa do topo
for i in range(len(tabuleiro)): #Explosão da da bomba
    for j in range(len(tabuleiro[i])):
        if tabuleiro[i][j] == '🧨':
            tabuleiro[i][j] = '💥'
```

Figure 4. Explosão da bomba.

```
try: #verifica os blocos dentro do alcance
    if tabuleiro[i+h][j+m] != '💥':
        tabuleiro[i+h][j+m] = '💥'
except IndexError:
    pass
```

Figure 5. Tratamento de erro na verificação.

```
MENU PRINCIPAL

[1] NOVO JOGO
[2] FECHAR
1
Traceback (most recent call last):
  File "c:\Users\felip\OneDrive\Documentos\PROJETOS DE PYTHON\PBL\PBL 2\Teste_do_TETRSI.py", line 225, in <module>
    curses.wrapper(play_nojogo)
  File "C:\Users\felip\AppData\Local\Programs\Python\Python312\Lib\curses\__init__.py", line 94, in wrapper
    return func(stdscr, *args, **kwds)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "c:\Users\felip\OneDrive\Documentos\PROJETOS DE PYTHON\PBL\PBL 2\Teste_do_TETRSI.py", line 193, in play_nojogo
    aparecer_tab(telinha, tabuleiro, total_pontos) # Mostra o tabuleiro
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "c:\Users\felip\OneDrive\Documentos\PROJETOS DE PYTHON\PBL\PBL 2\Teste_do_TETRSI.py", line 64, in aparecer_tab
    telinha.addstr (f'Pontuação: {total_pontos}\n'
_curses.error: addwstr() returned ERR
PS C:\Users\felip\OneDrive\Documentos\PROJETOS DE PYTHON\PBL\PBL 2>
```

Figure 6. Erro de minimização de tela.

4. Conclusão

O relatório contém a forma como esse código foi construído, visto que explica cada bloco de funções necessárias para o bom funcionamento. Nele contém um manual de instruções para que o usuário possa entender como funciona o Tetris construído na linguagem *Python*. O código está a disposição de melhorias de algumas funcionalidades, bem como a movimentação dos *Tetriminos*, deixando eles de uma forma mais fluida e retirando o travamento das peças quando as teclas de movimentação forem pressionadas. Possivelmente poderia ser utilizada a biblioteca *curses* ou *Keyboard e numpy* para manipulação das matrizes.

Outro ponto de melhoria seria exibir a próxima peça para o usuário e uma aba com as melhores pontuações feitas pelo jogador, fazendo com que ele buscar sempre bater o seu recorde.

5. Referências

Awari. (2023, Novembro 24). Função Len Python: Como usar a função len para contar caracteres em Python. Disponivel em: Https://Awari.Com.Br/Funcao-Len-Python-Como-Usar-a-Funcao-Len-Para-Contar-Caracteres-Em-Python/.

Bozza, C. (2011, Agosto 18). A história do Tetris. Disponivel em: Https://Www.Techtudo.Com.Br/Noticias/2011/08/Historia-Do-Tetris.Ghtml.

Guanabara, G. (2018, Agosto 1). Exercício Python 086 - Matriz em Python. Disponivel em:Https://Www.Youtube.Com/Watch?V=EGmlFdwD4C4.

Kuchling, A. M., Raymond, E. S. (2024, Agosto 28). Programação em Curses com Python. Disponivel em: Https://Docs.Python.Org/Pt-Br/3/Howto/Curses.Html.

Leonardo. (2023, Fevereiro 18). APRENDA 2 formas de gerar Matriz Python — [FÁCIL E RÁPIDO]. Disponivel em: Https://Www.Youtube.Com/Watch?V=W-Fn4r6r1gA.

LLC, T. H. (2024, Outubro 28). Sobre Tetris. Https://Tetris.Com/about-Us.

random — Gera números pseudoaleatórios. (2024, Agosto 28). Disponivel em: Https://Docs.Python.Org/Pt-Br/3.12/Library/Random.Html.