SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO – UFERSA
CENTRO MULTIDISCIPLINAR DE PAU DOS FERROS – CMPF

MARCOS ROBERTO EDUARDO DE ALBUQUERQUE

PROCESSO DE CONSTRUÇÃO DO JOGO TETRIS NO UNITY COMO INCENTIVO À APRENDIZAGEM DE PROGRAMAÇÃO

PAUDOS FERROS – RN
2021

MARCOS ROBERTO EDUARDO DE ALBUQUERQUE

PROCESSO DE CONSTRUÇÃO DO JOGO TETRIS NO UNITY COMO INCENTIVO À APRENDIZAGEM DE PROGRAMAÇÃO

Monografia apresentada à banca examinadora da Universidade Federal Rural do Semi-Árido como requisito para a obtenção do título de Bacharel em Ciência e Tecnologia

Orientador: Prof. Dr. Rodrigo Soares Semente

PAUDOS FERROS – RN
2021

A345p Albuquerque, Marcos Roberto Eduardo de.
Processo de construção do jogo Tetris no Unity como incentivo à aprendizagem de programação / Marcos Roberto Eduardo de Albuquerque. - 2021.
31 f. : il.

Orientador: Rodrigo Soares Semente.
Monografia (graduação) - Universidade Federal Rural do Semi-árido, Curso de Ciência e Tecnologia, 2021.

1. informática na educação. 2. ensino de programação. 3. desenvolvimento de jogos. I. Soares Semente, Rodrigo, orient. II. Título.

Ficha catalográfica elaborada por sistema gerador automáto em conformidade com AACR2 e os dados fornecidos pelo) autor(a).
Biblioteca Campus Pau dos Ferros
Bibliotecária: Erlanda Maria Lopes da Silva
CRB: 1115

MARCOS ROBERTO EDUARDO DE ALBUQUERQUE

PROCESSO DE CONSTRUÇÃO DO JOGO TETRIS NO UNITY COMO INCENTIVO À APRENDIZAGEM DE PROGRAMAÇÃO

Monografia apresentada à banca examinadora da Universidade Federal Rural do Semi-Árido como requisito para a obtenção do título de Bacharel em Ciência e Tecnologia

Defendido em: 16/ Novembro / 2021

Banca Examinadora

RODRIGO SOARES SEMENTE:04790852423

Assinado digitalmente por RODRIGO SOARES SEMENTE:04790852423
DN: cn=RODRIGO SOARES SEMENTE:04790852423, c=BR, o=ICPEdu, ou=UFERSA - Universidade Federal Rural do Semi-Arido, email=RODRIGO.SEMENTE@UFERSA.EDU.BR
Data: 2021.11.30 18:35:57 -03'00'

Prof. Dr. Rodrigo Soares Semente
Presidente

Profa. Dra. Sâmara de Cavalcante Paiva
Membro Examinador

PATRICK CESAR ALVES TERREMATTE:01333009437

Digitally signed by PATRICK CESAR ALVES TERREMATTE:01333009437
Date: 2021.12.01 00:28:55 -03'00'

Prof. Msc. Patrick Cesar Alves Terrematte
Membro Examinador

Digitally signed by MARCO DIEGO AURELIO MESQUITA:78702640244
Date: 2021.12.02 18:22:24

Prof. Msc. Marco Diego Aurélio Mesquita
Membro Suplente

Resumo

O processo de aprendizagem de lógica de programação pode ser tornar uma tarefa complicada para iniciantes no mundo da programação. Para superar essa dificuldade, este trabalho apresenta o uso de desenvolvimento de jogos como método para aumentar o interesse dos alunos na aprendizagem de programação. Para isso, foi proposto a gravação de vídeo aulas ensinando a implementar, passo a passo, uma versão do jogo Tetris, destacando os conceitos de programação usados no momento de criação do jogo. O jogo foi desenvolvido no ambiente de desenvolvimento de jogos do Unity. Uma série de 14 videoaulas foi gravada, e tais videoaulas foram disponibilizadas para professores de disciplinas de programação da UFERSA para que passassem aos alunos para que os mesmos executem o que está sendo passado nos vídeos. Por fim será aplicado um questionário a esses alunos visando verificar a eficácia desse método de aprendizagem de programação. Apesar de não ter havido respostas aos formulário, as videoaulas ainda continuarão disponíveis e poderão ser utilizadas como forma complementar de ensino.

Palavras-chave: Informática na Educação, Ensino de Programação, Desenvolvimento de Jogos.

Abstract

The process of learning programming logic can become a complicated task for beginners in the programming world. To overcome this difficulty, this work presents the use of game development as a method to increase the students' interest in learning programming. For this, it was proposed to record video lessons teaching how to implement, step by step, a version of the Tetris game, highlighting the programming concepts used when creating the game. The game was developed in the Unity game development environment. A series of 14 videotapes were recorded, and these videotapes were made available to teachers of programming courses at UFERSA so that they could show them to the students so that they could execute what is being shown in the videos. Finally, a questionnaire will be applied to these students to verify the effectiveness of this method of learning programming. Although there have been no answers to the form, the video classes will still be available and can be used as a complementary way of teaching.

Keywords: Informatics in Education, Teaching Programming, Game Development.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Nos cursos de graduação, principalmente nas áreas de engenharia e tecnologia, o processo de aprendizagem de programação pode se tornar uma tarefa difícil para alunos iniciantes. Para ajudá-los a superar esse desafio, muitos professores utilizam novos métodos de ensino que visam facilitar a compreensão de conceitos básicos de algoritmos e lógica de programação.

Como métodos de ensino é comum o uso de pseudo linguagem como o VisuAlg, que possibilita abordar inicialmente conceitos de lógica de programação de maneira mais amigável, em portugol (português estruturado) (COUTINHO, 2018). Outro meio de suporte ao ensino de programação é o ambiente de programação Scratch, desenvolvido no MIT como forma de estimular a aprendizagem de lógica de programação por meio de uma linguagem de blocos (MIT, 2021). No entanto, ainda é notório o baixo aproveitamento e desmotivação de alunos em disciplinas de programação.

Contudo, para superar esse obstáculo, este trabalho propõe aplicar o uso de desenvolvimento de jogos, em especial o jogo Tetris, como uma nova abordagem para estimular o interesse de alunos em aprender programação. Para isso, será usado o motor Unity (Unity Technologies, 2021) como ambiente de desenvolvimento, pois o mesmo possui uma ampla variedade de recursos, é multiplataforma, sendo suportado no Linux, Windows e MacOS e, além de ter uma versão gratuita, é um dos ambientes mais usados para o desenvolvimento de jogos, possuindo diversos fóruns na internet para se tirar dúvidas.

Assim, foi produzido uma série de vídeos tutoriais onde foi mostrado o desenvolvimento, passo a passo, do jogo Tetris, destacando os conceitos de programação usados em cada etapa. Os vídeos criados foram apresentados a professores das disciplinas de programação da Universidade Federal Rural do Semi-Árido (UFERSA), com o intuito de que repassem para seus alunos, para que tentem reproduzir o tutorial como uma forma complementar de aprendizagem.

1.1 PROBLEMATIZAÇÃO

Os índices de reprovação em disciplinas de algoritmos e lógica de programação ainda continua a ser um problema em muitas instituições de ensino superior. De acordo com Bosse e Gerosa (2015) o percentual médio de reprovação/trancamento na Universidade de São Paulo (USP) no período de 2010 a 2014 é de 29,31% (BOSSE, 2015). De acordo com os estudos de Filho et al (2018), constatou-se que há um aumento no índice de reprovação a cada repetência (FILHO et al, 2018).

Segundo Rolim (2019), na Universidade Federal Rural do Semi-Árido (UFERSA) campus Pau dos Ferros – RN, a taxa de insucesso entre os semestres 2015.2 e 2017.2 foi de 70,94%. Com isso, foi realizado um projeto de Pré-Algoritmos que visa melhorar o desempenho dos alunos. Nos semestres 2018.1 e 2018.2 apenas 20% dos alunos participaram do projeto com frequência maior ou igual a 50%, onde os mesmos alcançaram uma taxa final de sucesso de 76%. Apesar da baixa adesão, o aproveitamento geral dos alunos matriculados na disciplina foi de 50% (ROLIM, 2019). Para Rolim (2020), a taxa de insucesso entre os quatro períodos do projeto de Pré-Algoritmos foi de 52%, que ainda pode ser considerada alta (ROLIM, 2020). Nem todos os alunos da disciplina de Algoritmos participam do curso de Pré-Algoritmos, e seria interessante aplicar metodologias dentro da própria disciplina que ajudem a todos os alunos a aprenderem o conteúdo.

1.2 OBJETIVOS

- **GERAL**
    - Contribuir para o processo de ensino e aprendizagem de programação através do desenvolvimento de jogos baseado no projeto de implementação de um jogo Tetris.
- **ESPECÍFICOS**
    - Elaborar recursos educacionais em vídeo do desenvolvimento de uma versão do jogo Tetris no motor da Unity.
    - Abordar, durante a implementação do jogo, conceitos básicos de programação.
    - Divulgar e avaliar a integração do processo de desenvolvimento de jogos como metodologia de ensino de programação.

2 REVISÃO TEÓRICA

Na segunda parte foi discutido a respeito de diferentes trabalhos envolvendo o desenvolvimento de jogos como método para o ensino de programação. Neste capítulo, os conteúdos apresentados foram divididos em duas partes. Na primeira parte, foi apresentada uma breve história do Tetris, jogo escolhido para ser desenvolvido neste trabalho.

2.1 ORIGEM DO JOGO TETRIS

O jogo tetris tem sua origem em 1984 na União Soviética, no período da Guerra Fria, tendo sido desenvolvido pelo russo Alexey Pajitnov, que na época trabalhava com aperfeiçoamento de Inteligência Artificial (LICHAN, 2015). Na criação do game, ele se inspirou no jogo de quebra-cabeça chamado Pentaminó, criado pelo matemático norte-americano Solomon Golomb. O Pentaminó consistia de peças formadas por cinco quadrados e que podiam assumir 12 formas diferentes.

Alexey programava jogos com um foco psicológico nos computadores soviéticos para poder testar a capacidade deles. Para tentar implementar o Pentaminó no computador de uma maneira mais simples, Alexey decide retirar um quadrado de cada peça, formando peças de quatro quadrados, podendo assumir até sete formas distintas. O nome escolhido para o jogo deriva dos nomes tetramino (pois era formado por quatro quadrados) e tênis, o esporte favorito dele.

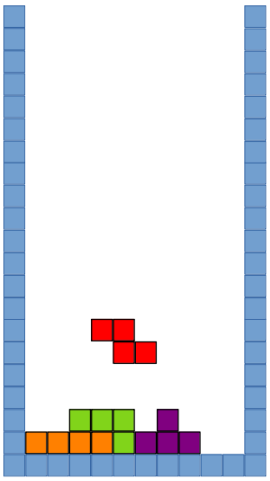

Figura 1: Jogo Tetris
Fonte: Autor.

Na Figura 1 tem-se uma ilustração do jogo Tetris, que consiste basicamente em empilhar os tetraminós que estão em queda de modo a preencher uma linha horizontal.

Após concluído, o jogo chamou a atenção no mundo inteiro e logo se tornou um sucesso. Enquanto que o foco de muitos jogos era destruições, matanças e explosões, "O *game* apela para um lado diferente do ser humano, o lado da construção", disse Alexey um ano depois (LICHAN, 2015).

Atualmente, segundo Izabella (2019),

> "Tetris" é considerado o jogo mais vendido da história, na frente de "Minecraft" e "Grand Theft Auto V". De acordo com a Tetris Company, foram mais de 500 milhões de downloads em aparelhos móveis e centenas de milhões de produtos vendidos em mais de 50 plataformas em 200 países.

2.2 DESENVOLVIMENTO DE JOGOS COMO METODOLOGIA DE ENSINO

Pontes (2013) considera o desenvolvimento de jogos como uma forma de estimular o aprendizado dos fundamentos de programação. O mesmo afirma que a "utilização de jogos no contexto educacional oferece ao professor uma interessante estratégia para o desenvolvimento da criatividade, aptidão e autonomia do aluno, propiciando a construção do aprendizado em um cenário interdisciplinar". Ele destaca ainda que os jogos em si não são os únicos responsáveis pelo aprendizado dos alunos, enfatizando a importância das reflexões a respeito do que foi aprendido e a forma como o aluno visualiza o conhecimento obtido sendo aplicado (PONTES, 2013).

Em seu trabalho, Coutinho (2018) ressalta a existência de várias ferramentas que objetivam apoiar o ensino de programação, tais como o Scratch e o VisuAlg, bem como o desenvolvimento de pequenos jogos como forma de ensinar lógica de programação e raciocínio lógico e matemático. O seu trabalho consistiu na utilização da ferramenta de desenvolvimento Game Maker para o ensino de lógica de programação em um curso de graduação em Sistemas de Mídias Digitais (COUTINHO, 2018).

Batista (2012) apresenta o *framework* JPlay como um recurso para auxiliar o ensino de lógica de programação com a implementação de jogos. Esse método foi

aplicado em uma turma de Análise e Desenvolvimento de Sistemas, onde foi ensinado conceitos da linguagem Java como classe, métodos e objetos, além de explanar a respeito do *framework* JPlay. Por fim, foi desenvolvido com a turma, o jogo Genius. Os alunos conseguiram desenvolver satisfatoriamente a parte gráfica e tiveram dificuldade na implementação dos métodos, porém notou-se que o desenvolvimento de um jogo estimulou os alunos na busca de soluções diante das dificuldades (BATISTA, 2012).

Falcão (2015) procura averiguar o uso do desenvolvimento de jogos com possível metodologia para o ensino de programação. Em seu trabalho, foi desenvolvida uma oficina para o ensino de programação com o desenvolvimento de jogos. Foi utilizado o motor de jogos Unity e linguagem C# com intuito de desenvolver uma versão do jogo Space Invaders. A oficina consistia em três partes: na primeira parte, foi apresentado o ambiente de desenvolvimento da Unity (considerada adequada por possuir versões grátis); na segunda parte ocorreu o desenvolvimento do jogo, onde novos conceitos de programação eram apresentados; por último, aplicou-se um questionários aos participantes da oficina, e também aos que não participaram. Como resultados, houve uma grande porcentagem de acertos, a respeito do conteúdo ensinado, por parte dos alunos que participaram da oficina quando comparado aos que não participaram.

Para estimular o interesse de alunos do curso introdutório de programação gráfica Jin e Chandramouli (2014), em seu trabalho, usam da prática de desenvolvimento de alguns jogos 2D, incluindo o jogo Pong com o intuito de ensinar declarações condicionais e a interação teclado/mouse; o jogo Tetris é desenvolvido visando reforçar a estrutura de dados do *array*, e o jogo Angry Bird é implementado para o ensino da programação usando bibliotecas de código aberto (JIN, 2014). O método usado se mostrou eficaz para ensinar conceitos de programação, diminuindo a complexidade da disciplina.

3 METODOLOGIA

Como forma de estimular a aprendizagem de programação, foi desenvolvido uma série de vídeos tutoriais do desenvolvimento de uma versão do jogo Tetris no Unity. A cada etapa do desenvolvimento foi explicado os principais conceitos de

programação usados na construção do jogo. Na produção dos vídeos tutoriais foram utilizados os softwares OBS Studio (OBS Project, 2021), para a gravação dos vídeos, e o KdenLive (KdenLive, 2021), para a edição e formatação.

Após a conclusão dos vídeos tutoriais, os mesmos foram passados para os professores das disciplinas de programação da UFERSA, no semestre 2021.1, com o intuito que repassem para seus alunos e estes assistam o vídeo tutorial e tentem reproduzir o que está sendo ensinado. Para que os vídeos pudessem ser acessados de modo fácil, eles foram postados em um canal no YouTube.

3.1 UNITY

Os jogos eletrônicos surgiram pouco tempo após a invenção dos computadores eletrônicos, seja para testar a capacidade dos computadores da época ou simplesmente como forma de entretenimento. Com o passar dos anos, os jogos eletrônicos foram se popularizando, criando assim uma nova demanda no mercado de tecnologia. Com isso, foram desenvolvidos diversos jogos em diferentes plataformas. Para o desenvolvimento de um jogo, é necessário além de um computador (hardware), de uma linguagem de programação para a criação do jogo (software).

À medida em que os computadores foram evoluindo, por consequência, os jogos também foram. No entanto, dependendo da complexidade de um jogo, desenvolvê-lo do zero pode se tornar uma tarefa muito árdua. Uma solução prática para esse problema, foi a criação de um ambiente que fornecesse os recursos necessários para se criar uma infinidade de jogos, chamada motor de desenvolvimento. A engine Unity (Unity Technologies, 2021), ilustrada na Figura 2, é uma poderosa ferramenta para o desenvolvimento de jogos em 2D ou 3D criada para esse propósito, trazendo assim uma série de vantagens, seja para desenvolvedores experientes ou para quem está começando, principalmente.

Entre essas vantagens, pode-se destacar a "*Asset Store*", que possui uma grande fonte de recursos como texturas, modelos, arquivos de áudios, extensões etc. Dessa forma, o desenvolvedor iniciante não precisa, obrigatoriamente, criar os próprios elementos do jogo. Outra vantagem muito importante, é que além do próprio Unity ser multiplataforma, estando disponível para Linux, macOS e Windows, com ele é possível desenvolver jogos para computadores, consoles, navegadores e, inclusive, para dispositivos móveis, onde se concentra a maior parte da demanda por jogos. Ademais, o Unity possui uma ampla gama de

recursos disponíveis na internet e na própria engine (com modelos básicos de jogos em 2D e 3D) para quem está começando na área de desenvolvimento de jogos. Por ser uma game engine popular, há diversos fóruns na internet para tirar dúvidas quando surgir algum problema no momento do desenvolvimento.

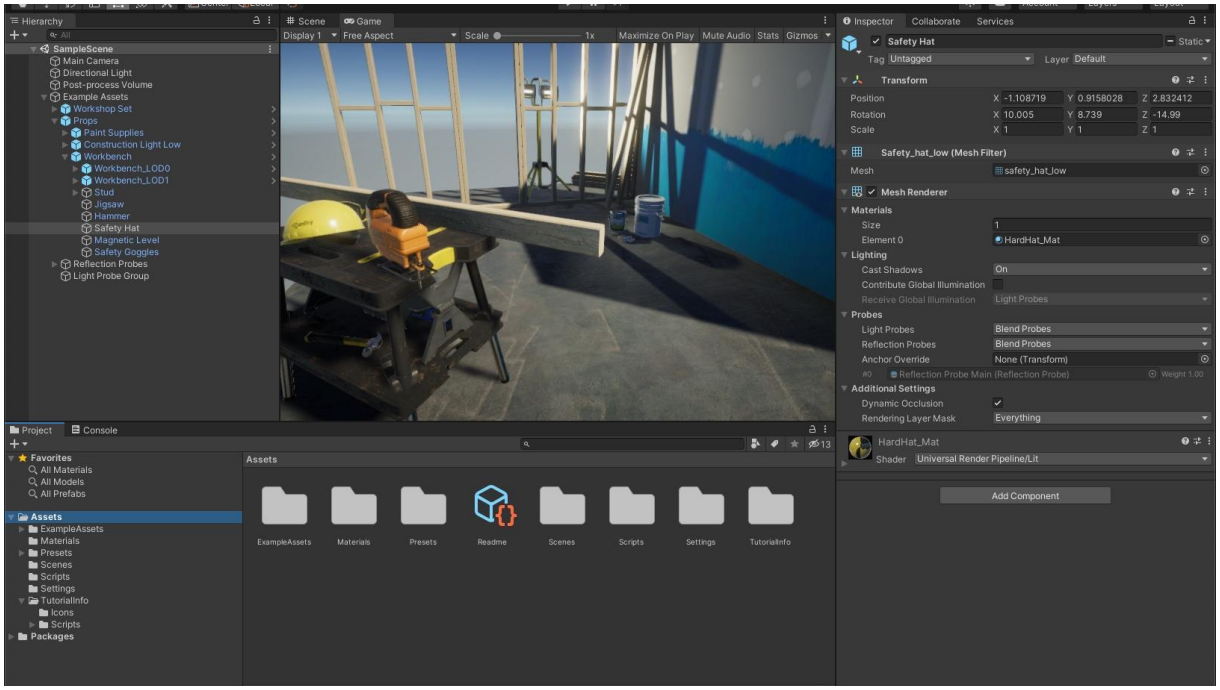

Figura 2: Game engine Unity
Fonte: Unity Technologies (2021)

Todos os aspectos de jogabilidade e funcionamento do Unity baseiam-se em três blocos de construção: GameObjects, componentes e variáveis (Unity Technologies, 2021). Os GameObjects são como elementos vazios, e somente após adicionarmos componentes, ele pode se tornar algo utilizável. Esses componentes caracterizam e dão controle a um GameObject, podendo por exemplo atribuí-lo uma propriedade de colisão, que por sua vez pode ser controlada a partir de variáveis. Além disso, podemos adicionar uma lógica maior de funcionamento a um GameObject usando scripts como componentes. Esses *scripts*, no Unity, podem ser programados em C# ou JavaScript, que são linguagens de alto nível e, consequentemente, fáceis de aprender.

4 RESULTADOS

No total, foram gravadas 14 videoaulas, desde a aula 00 até a aula 13, explicando passo a passo, o desenvolvimento de uma versão do jogo Tetris. Os vídeos tutoriais foram baseados numa playlist de vídeos do canal no YouTube chamado “CriaJogos” (CriaJogos, 2021). Na aula 00, foi introduzido a proposta de desenvolvimento do jogo em questão usando o ambiente de desenvolvimento do Unity, explicando primeiramente o que é o jogo Tetris, bem como sua mecânica de funcionamento e regras. Após isso, começou-se o processo de criação do cenário do jogo, Figura 3, onde foram usadas 8 figuras em forma de quadrados coloridos com dimensões 32x32 pixels (chamados *sprites*) para criar os tetraminós e a grade do jogo.

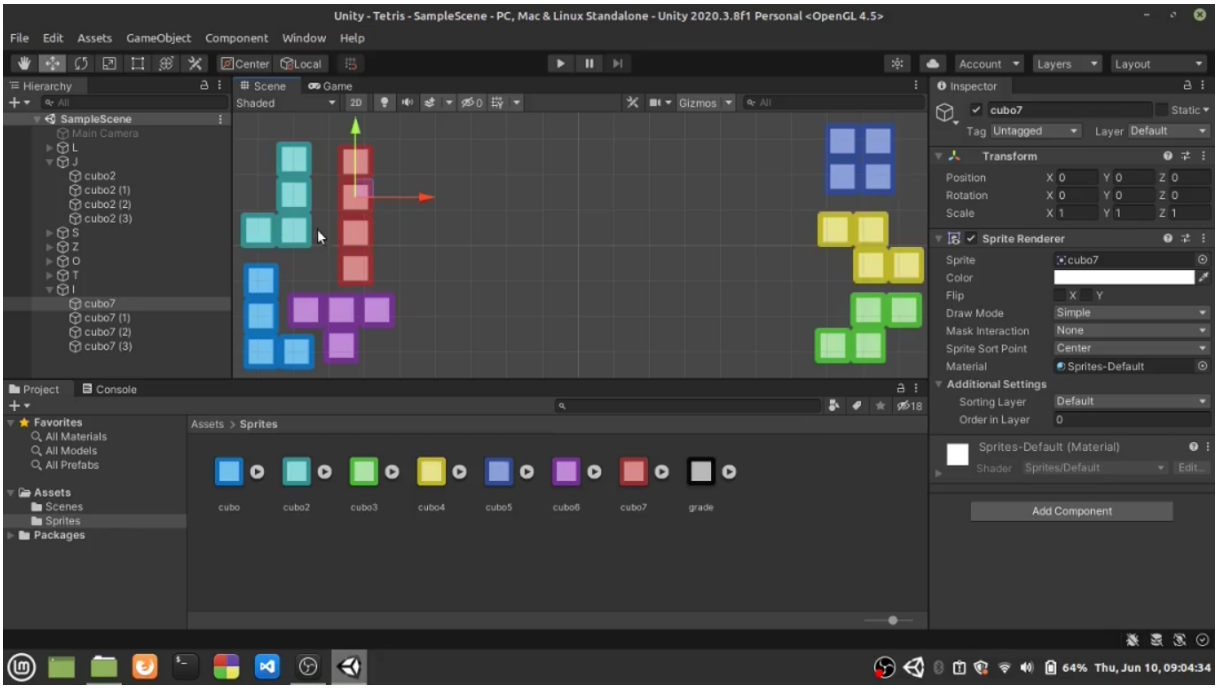

Figura 3: Criação do cenário do jogo, na aula 00.
Fonte: Autor.

Na aula 01, segundo vídeo tutorial, deu-se início a parte de programação propriamente dita, como pode ser visto na Figura 4. Visando dar movimento aos tetraminós criados na aula anterior, foi criado o primeiro script em C# (linguagem de programação usada) chamado “mov”. A partir disso, foi introduzido a estrutura básica de um código em C# no ambiente do Unity. Foi destacado inicialmente as bibliotecas que vêm por padrão ao se criar um novo script; a classe e seus métodos padrões *Start* e *Update*, enfatizando que o método *Start* é chamado no primeiro momento de execução e geralmente é usado para inicializar as variáveis

de instâncias criadas no decorrer do desenvolvimento do jogo, e que o método *Update* é chamado uma vez a cada frame, em um *loop* infinito, e que ele é usado para chamar os demais métodos. Depois de apresentado esses conceito básicos, deu-se início a criação das primeiras linhas de código dentro do método *Update*, usando a estrutura condicional IF para verificar se o jogador apertou em uma determinada tecla. Foi mostrado que se a condição fosse satisfeita, a instrução dentro da estrutura condicional IF seria executada, realizando um determinado movimento – para direita, esquerda, para baixo ou rotação – a depender da tecla que o jogador apertasse; e caso contrário, a instrução não seria executada.

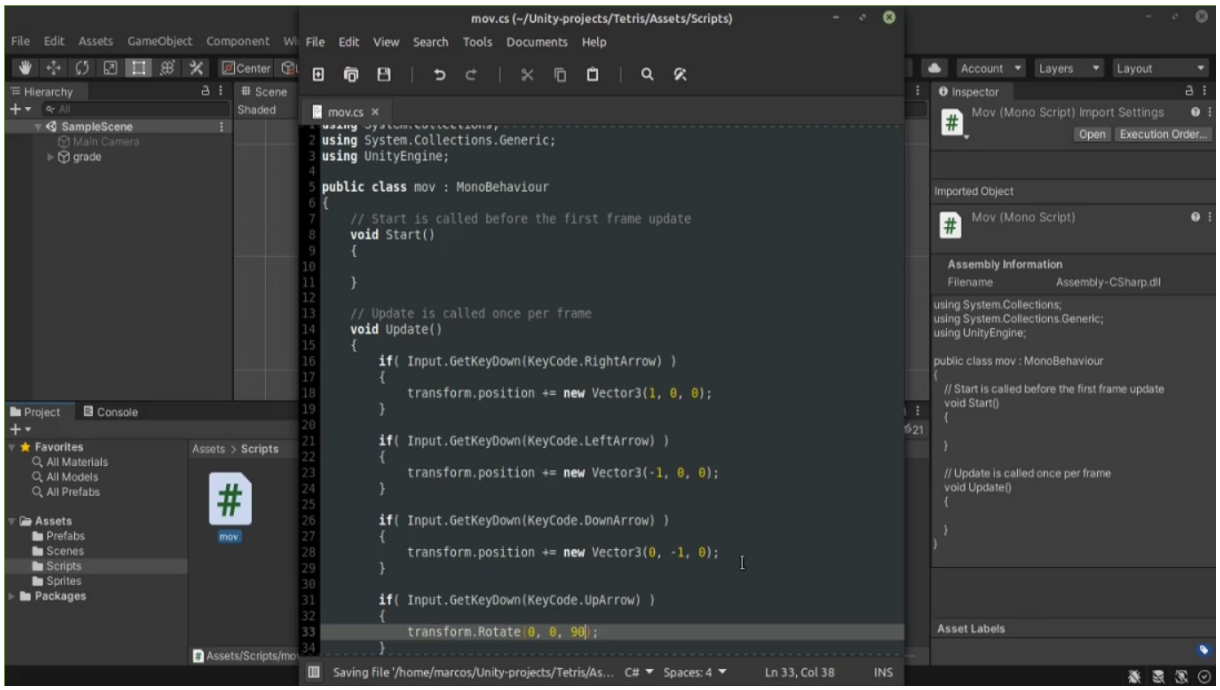

Figura 4: Implementação das primeiras linhas de código do jogo Tetris.
Fonte: Autor.

Para otimizar o movimento de rotação de cada tetramino, na aula 02 foi apresentada a implementação de um novo método chamado “checaRoda”, usado para verificar se um determinado tetraminó pode rodar ou não, e se o mesmo teria ou não uma rotação limitada. Para isso, ainda foi utilizado a estrutura condicional *IF*, só que de forma aninhada em conjunto com a condicional *ELSE*, que é executada caso a condição do *IF* com a qual está relacionada não for satisfeita. Além disso, foi demonstrado a declaração de variáveis do tipo booleano – que pode assumir o valor verdadeiro (*true*) ou falso (*false*) – e o uso do operador de negação “!”. Após a implementação do método “checaRoda”, mostrou-se como é feita uma chamada de método, chamando o método criado dentro do método *Update*.

Após a otimização da rotação dos tetraminós, o objetivo da aula 03 foi mostrar como configurar a grade (feita na aula 00) para impor limites aos movimentos dos tetraminós, para que cada tetraminó não atravesse a grade em nenhuma das direções. Para isso foi criado um novo script, chamado “gameManager”. Nesse script foi implementado o método chamado “dentroGrade” que verifica se o tetraminó está ou não respeitando os limites da grade. Nesse método foi apresentado o conceito de parâmetros, tipo de retorno booleano, uso do operador lógico “&&”, e uso de casting para converter variáveis. Ademais, foi elucidado como declarar uma instância da classe “gameManager” dentro da classe “mov”.

A aula 04 teve como foco a implementação do sistema de surgimento de um novo tetraminó aleatório logo após o tetraminó atual tocar no fundo da grade, bem como a implementação da queda automática do tetraminó. Para tal foi criado um novo script chamado *spawn*, e nele, foi explanado o conceito de arrays (vetores), que foi usado para armazenar cada tetraminó, assim como o uso de números aleatórios para selecionar aleatoriamente um novo tetraminó. Para programar a funcionalidade de queda automática do tetraminó, apresentou-se o uso do operador lógico OU (simbolizador por ||) e comparadores de magnitude “>=” dentro da estrutura condicional *IF* para condicionar se era uma queda automática ou se o jogador apertou a tecla de seta para baixo.

No jogo Tetris, quando o jogador aperte uma determinada tecla de movimento e continua pressionando-a, é comum que o tetraminó se desloque com uma determinada velocidade, sem que o jogador tenha que estar pressionando e soltando a tecla várias vezes para isso. Na aula 05 foi ensinado como realizar esse comportamento, usando e explicando o operador de atribuição “+=”. Outro comportamento que é característico do jogo Tetris é o fato de que quando um tetraminó atinge o chão da grade, esse tetraminó é desativado, e automaticamente outro é chamado no topo da grade. Para implementar esse sistema, na aula 06 foi criado um novo método chamado “proximaPeca” (“peca” é usado para se referir ao tetraminó) que instancia um novo tetraminó caso o tetraminó atual toque o fundo da grade.

Após a aula 06, surgiu um novo problema: quando um tetraminó jaz no fundo da grade, o tetraminó seguinte consegue sobrepor o anterior, o que não deveria acontecer. O correto é que o próximo tetraminó deva ser capaz de pousar em cima do tetraminó anterior, podendo ser empilhado. Essa funcionalidade foi implementada na aula 07, onde foi explanado o conceito de matrizes (*array* bidimensional) usando o laço de repetição *FOR* e *FOREACH*.

Feito isso, na aula 08 foi implementado o sistema que deleta as linhas horizontais cheias. Nessa aula, mostrou-se a criação de um conjunto de métodos que iriam realizar esse trabalho. Foi enfatizado a importância da modularização do código em diferentes métodos a fim de facilitar a manutenção, reutilização e legibilidade do código.

Nas videoaulas 09 e 10, respectivamente, foi ensinado como implementar um sistema de pontuação usando o operador de incremento "+=", e a criar um nível de dificuldade baseado na pontuação do jogo. O sistema que mostra na tela qual o próximo tetraminó a ser usado foi apresentado na aula 11. E por fim, na aula 12 foi ensinado como criar um botão para pausar e despausar o jogo usando a condicional IF e uma variável booleana, e na aula 13 mostrou-se como criar uma tela de "*game over*" e como chamá-la assim que o sistema detectar o fim do jogo, ou seja, quando houver uma grande pilha de tetraminós onde não é mais possível gerar novos tetraminós.

A playlist com todas as videoaulas gravadas pode ser acessada pelo link.

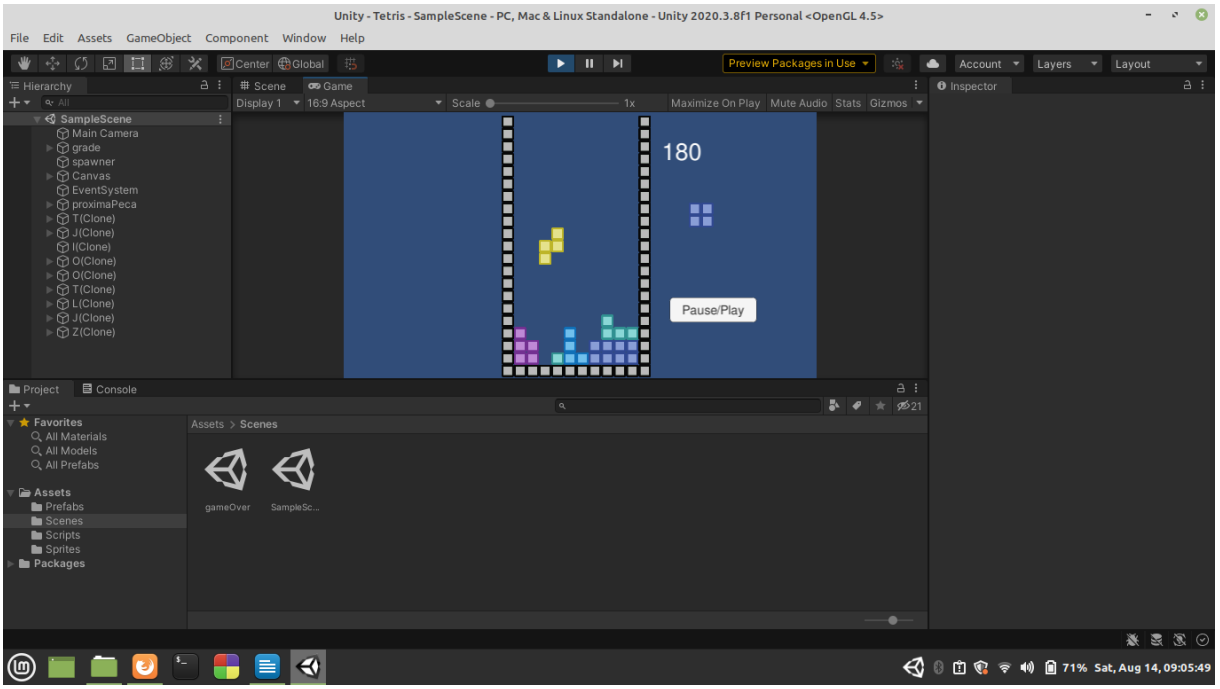

Figura 5: Versão final do jogo.
Fonte: Autor.

4.1 Discussão dos Resultados

Uma vez que as videoaulas foram postadas no YouTube em forma de playlist, o link para os vídeos foi enviado para o e-mail de 16 professores que ensinavam disciplinas de

programação na UFERSA. No e-mail foi pedido que o link fosse repassado para seus alunos para que eles reproduzissem as videoaulas.

As videoaulas tiveram em média 12 visualizações, com maior concentração de visualizações nas primeiras aulas. Com as aulas remotas, e em decorrência da pandemia causada pela Covid-19, houve dificuldades na divulgação das videoaulas, não sendo possível garantir a reprodução dos vídeos pelos alunos, havendo assim poucas visualizações. Ademais, em cada videoaula havia um formulário com duas questões simples para serem respondidas ao final de cada vídeo, perguntando se a videoaula ajudou a entender melhor algum conceito de programação, e se eles conseguiram assistir o até o final.

Além de ter sido informado aos professores a respeito do formulário na descrição das videoaulas, havia também uma breve mensagem na tela inicial de cada vídeo avisando sobre o link do formulário na descrição. No entanto, apesar das visualizações das videoaulas, não houve nenhuma resposta aos formulários. Isso pode ser explicado pelo fato de não ter sido esclarecido de forma explícita no final de cada vídeo que haveria um formulário.

5 CONSIDERAÇÕES FINAIS

Com os altos índices de reprovação em disciplinas de programação, novos métodos de ensino no intuito de melhorar esses índices são necessários. O uso de programação de jogos torna-se um bom método de ensino pois o aluno pode ver na prática, e de modo divertido, a aplicação dos conceitos de programação aprendidos em sala de aula.

Portanto, este trabalho visou aumentar o interesse dos alunos na aprendizagem de programação, usando como método de ensino o desenvolvimento de jogos. Uma série de videoaulas foram produzidas explicando conceitos de programação no decorrer da implementação do jogo, e tais videoaulas foram disponibilizados a professores da UFERSA para que eles passassem para seus alunos.

Notou-se que houve um problema na comunicação com os professores, onde o envio de um e-mail descrevendo a proposta deste trabalho não se mostrou suficiente para garantir que os alunos executassem as videoaulas.

Apesar da falta de respostas aos formulários propostos, a série de videoaulas ainda pode ser utilizada como forma complementar de ensino pelos professores de disciplinas de programação.

Para trabalhos futuros propõe-se um tutorial da implementação de um jogo mais simples e que esteja inserido em um tema mais atual e atrativo para alunos que estão iniciando em disciplinas de programação.

REFERÊNCIAS

BATISTA, Gleison B. BRASIL, Kayro Rafael F. Utilização do motor de jogos JPlay como ferramenta de auxílio ao ensino da lógica de programação. **Anais do VII Congresso Norte e Nordeste de Pesquisa e Inovação (CONNEPI)**, Tocantins, 2012. Disponível em: <https://propi.ifto.edu.br/ocs/index.php/connepi/vii/paper/view/1257>. Acesso em: 12 mar, 2021.

BOSSE, Yorah. GEROSA, Marco Aurélio. **Reprovações e trancamentos nas disciplinas de introdução à programação da universidade de são paulo**: um estudo preliminar. USP – São Paulo, 2015.

COUTINHO, Emanuel F. BONATES, Mara F. MOREIRA, Leonardo O. Relato sobre o uso de ferramenta de desenvolvimento de jogos para o ensino introdutório de lógica de programação. **WCBIE**, Fortaleza, p. 689-698, 2018.

CriaJogos. **Tutorial Unity 5 - TETRIS [COMPLETO]**. Disponível em: < shorturl.at/nEFQ8 >. Acesso em: 12 mar. 2021.

FALCÃO, Emerson Uriel. JÚNIOR, José Luiz Ungeritch. Desenvolvimento de jogos eletrônicos como metodologia de ensino de programação para alunos do curso de informática. Santa Catarina : **MICTI,** 2015.

FILHO, Romir Soares de Souza. et al. **Análises do índice de reprovação na disciplina de tecnologia da informação I da Universidade Federal de Juiz de Fora**. UFJF- MG, 2018.

IZABELLA, Fernanda. **"Tetris" faz 35 anos, e Alexey Pajitnov nem pensa em reclamar**. 2019. Disponível em: <https://www.uol.com.br/start/ultimas-noticias/2019/06/10/alexey-pajitnov-e-os-35-anos-de-tetris.htm>. Acesso em: 18 mar, 2021.

JIN, Ge. CHANDRAMOULI, Magesh. Development of casual 2-D game laboratory exercises in introductory computer graphics programming course. **IAJC**, Flórida, 2014.

KdenLive. **Libre Video Editor**. Disponível em: <https://kdenlive.org/en/>. Acesso em 29 out, 2021.

LICHAND, Michel. **De volta para o Arkade:** bloco a bloco, a história de Tetris. 2015. Disponível em: <https://www.arkade.com.br/volta-arkade-bloco-bloco-historia-tetris/>. Acesso em 18 mar, 2021.

OBS Project. OBS Studio. Disponível em: <https://obsproject.com>. Acesso em: 29 out, 2021.

Scratch Foundation. **Scratch**. Disponível em: <https://scratch.mit.edu>. Acesso em: 28 mar, 2021.

PONTES, Herleson Paiva. Desenvolvimento de jogos no processo de aprendizado em algoritmos e programação de computadores. **SBC – Proceedings of SBGames**: Fortaleza, 2013.

ROLIM, Reudismam. et al. Pré-Algoritmos – Ações de apoio à melhoria do ensino de graduação. **ECOP**, 2019.

ROLIM, Reudismam. Motivando os discentes e solucionando seus desafios de aprendizagem, um estudo do projeto de ensino pré-algoritmos. **ECOP**, 2020.

Unity Technologies. **Unity Engine**. Disponível em: <https://unity.com/pt>. Acesso em 29 out, 2021.

ANEXO A – Script mov.cs

Fonte: Canal no YouTube *CriaJogos*: Tutorial Unity 5 - TETRIS.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class mov : MonoBehaviour
{
    public bool podeRodar;
    public bool roda360;

    gameManager gManager;
    spawn gSpawn;

    public float queda;

    public float velocidade;
    public float timer;

    // Start is called before the first frame update
    void Start()
    {
        timer = velocidade;
        gManager = GameObject.FindObjectOfType<gameManager>();
        gSpawn = GameObject.FindObjectOfType<spawn>();
    }

    // Update is called once per frame
    void Update()
    {
        if(!gManager.pause)
        {
            if( gManager.pontoDificuldade > 1000 )
            {
                gManager.pontoDificuldade -= 1000;
                gManager.dificuldade += 0.5f;
            }

            if( Input.GetKeyUp(KeyCode.RightArrow) ||
Input.GetKeyUp(KeyCode.LeftArrow) || Input.GetKeyUp(KeyCode.DownArrow) )
                timer = velocidade;

            if( Input.GetKey(KeyCode.RightArrow) )
            {
                timer += Time.deltaTime;
```

```
        if(timer > velocidade)
        {
            transform.position += new Vector3(1, 0, 0);
            timer = 0;
        }

        if( posicaoValida() )
        {
            gManager.atualizaGrade(this);
        }
        else
        {
            transform.position += new Vector3(-1, 0, 0);
        }
    }

    if( Input.GetKey(KeyCode.LeftArrow) )
    {
        timer += Time.deltaTime;

        if(timer > velocidade)
        {
                transform.position += new Vector3(-1, 0, 0);
                timer = 0;
        }


        if( posicaoValida() )
        {
                gManager.atualizaGrade(this);
        }
        else
        {
                transform.position += new Vector3(1, 0, 0);
        }
    }

    if( Input.GetKey(KeyCode.DownArrow) ) // || Time.time - queda >= 1 )
    {
        timer += Time.deltaTime;

        if(timer > velocidade)
        {
                transform.position += new Vector3(0, -1, 0);
                timer = 0;
        }

        if( posicaoValida() )
```

```
                {
                    gManager.atualizaGrade(this);
                }
                else
                {
                    transform.position += new Vector3(0, 1, 0);
                    gManager.apagaLinha();

                    if(gManager.acimaGrade(this))
                    {
                        gManager.gameOver();
                    }

                    gManager.score += 10;
                    gManager.pontoDificuldade += 10;
                    enabled = false;
                    gSpawn.proximaPeca();
                }

                //queda = Time.time;
            }

            if( Time.time - queda >= ( 1 / gManager.dificuldade ) &&
!Input.GetKey(KeyCode.DownArrow))
            {
                transform.position += new Vector3(0, -1, 0);

                if( posicaoValida() )
                {
                    gManager.atualizaGrade(this);
                }
                else
                {
                    transform.position += new Vector3(0, 1, 0);
                    gManager.apagaLinha();

                if(gManager.acimaGrade(this))
                {
                    gManager.gameOver();
                }

                    gManager.score += 10;
                    gManager.pontoDificuldade += 10;
                    enabled = false;
                    gSpawn.proximaPeca();
                }

                queda = Time.time;
```

```
            }

            if( Input.GetKeyDown(KeyCode.UpArrow) )
            {
                checaRoda();
            }
        }
    }

    void checaRoda()
    {
        if( podeRodar )
        {
            if( !roda360 )
            {
                if( transform.rotation.z < 0 )
                {
                    transform.Rotate(0, 0, 90);

                    if( posicaoValida() )
                    {
                        gManager.atualizaGrade(this);
                    }
                    else
                    {
                        transform.Rotate(0, 0, -90);
                    }
                }
                else
                {
                    transform.Rotate(0, 0, -90);

                    if( posicaoValida() )
                    {
                        gManager.atualizaGrade(this);
                    }
                    else
                    {
                        transform.Rotate(0, 0, 90);
                    }
                }
            }
            else
            {
                transform.Rotate(0, 0, -90);

                if( posicaoValida() )
                {
```

```
                    gManager.atualizaGrade(this);
                }
                else
                {
                    transform.Rotate(0, 0, 90);
                }
            }
        }
    }

    bool posicaoValida()
    {
        foreach( Transform child in transform )
        {
            Vector2 posBloco = gManager.arredonda(child.position);

            if( gManager.dentroGrade(posBloco) == false )
            {
                return false;
            }

            if( gManager.posicaoTransformGrade(posBloco) != null &&
gManager.posicaoTransformGrade(posBloco).parent != transform )
            {
                return false;
            }
        }

        return true;
    }
}
```

ANEXO B – Script gameManager.cs

Fonte: Canal no YouTube *CriaJogos*: Tutorial Unity 5 - TETRIS.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class gameManager : MonoBehaviour
{
    public static int altura = 20;
    public static int largura = 10;

    public bool pause = false;

    public static Transform[,] grade = new Transform[ largura, altura ];

    public int score = 0;
    public Text textScore;

    public int pontoDificuldade;
    public float dificuldade = 1;

    void Update()
    {
        textScore.text = score.ToString();
    }

    public bool dentroGrade( Vector2 posicao )
    {
        return ( (int)posicao.x >= 0 && (int)posicao.x < largura && (int)posicao.y >= 0 );
    }

    public Vector2 arredonda(Vector2 nA)
    {
        return new Vector2( Mathf.Round(nA.x), Mathf.Round(nA.y) );
    }

    public void atualizaGrade(mov pecaTetris)
    {
        for(int y = 0; y < altura; y++)
        {
            for( int x = 0; x < largura; x++ )
            {
                if( grade[x, y] != null )
                {
```

```
                    if( grade[x, y].parent == pecaTetris.transform )
                    {
                    grade[x, y] = null;
                    }
                }
            }
        }

        foreach( Transform peca in pecaTetris.transform)
        {
            Vector2 posicao = arredonda(peca.position);

            if(posicao.y < altura)
            {
                grade[(int)posicao.x, (int)posicao.y] = peca;
            }
        }
    }

    public Transform posicaoTransformGrade(Vector2 posicao)
    {
        if( posicao.y > altura - 1 )
        {
            return null;
        }
        else
        {
            return grade[(int)posicao.x, (int)posicao.y];
        }
    }

    public bool linhaCheia( int y )
    {
        for( int x = 0; x < largura; x++ )
        {
            if( grade[x, y] == null )
            {
                return false;
            }
        }

        return true;
    }

    public void deletaQuadrado( int y )
    {
        for(int x = 0; x < largura; x++)
        {
```

```
            Destroy(grade[x, y].gameObject);

            grade[x, y] = null;
        }
    }

    public void moveLinhaBaixo(int y)
    {
        for(int x = 0; x < largura; x++)
        {
            if(grade[x, y] != null)
            {
                grade[x, y-1] = grade[x, y];
                grade[x, y] = null;

                grade[x, y-1].position += new Vector3(0, -1, 0);
            }
        }
    }

    public void moveTodasLinhasBaixo(int y)
    {
        for(int i = y; i < altura; i++)
        {
            moveLinhaBaixo(i);
        }
    }

    public void apagaLinha()
    {
        for(int y = 0; y < altura; y++)
        {
            if(linhaCheia(y))
            {
                deletaQuadrado(y);
                moveTodasLinhasBaixo(y+1);
                y--;
                score += 100;
                pontoDificuldade += 100;
            }
        }
    }

    public void pausaJogo()
    {
        if(pause)
        {
            pause = false;
```

```
        }
        else
        {
            pause = true;
        }
    }

    public bool acimaGrade( mov peca )
    {
        for(int x = 0; x < altura; x++)
        {
            foreach( Transform cubo in peca.transform )
            {
                Vector2 posicao = arredonda(cubo.position);

                if(posicao.y > altura - 1)
                {
                    return true;
                }
            }
        }

        return false;
    }

    public void gameOver()
    {
        SceneManager.LoadScene("gameOver");
    }
}
```

ANEXO C – Script spawn.cs

Fonte: Canal no YouTube *CriaJogos*: Tutorial Unity 5 - TETRIS.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class spawn : MonoBehaviour
{
    public int proxPeca;

    public Transform[] criaPecas;
    public List<GameObject> mostraPeca;

    // Start is called before the first frame update
    void Start()
    {
        proxPeca = Random.Range(0, 7);
        proximaPeca();
    }

    public void proximaPeca()
    {
        Instantiate(criaPecas[proxPeca], transform.position, Quaternion.identity);

        proxPeca = Random.Range(0, 7);

        for(int i = 0; i < mostraPeca.Count; i++)
        {
            mostraPeca[i].SetActive(false);
        }

        mostraPeca[proxPeca].SetActive(true);
    }
}
```