REVISTA DO PROGRAMA DE PÓS-GRADUAÇÃO EM EDUCAÇÃO MATEMÁTICA DA UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL (UFMS)

ISSN 2359-2842 Volume 15, número 37 – 2022 DOI: 10.46312/pem.v15i37.13171

Simulando o Jogo da Corrida dos Cavalos: Elementos de Pensamento Computacional para Professores

Simulating the Horse Racing Game: Elements of Computational Thinking for Teachers

Sérgio Gonçalves de Sousa[1]

Leandro da Silva Machado[2]

Moisés Ceni de Almeida[3]

Leonardo Maricato Musmanno[4]

RESUMO

Neste trabalho, são expostos os resultados de uma investigação, em andamento, cujo objetivo é apresentar e explorar direcionamentos no que concerne à aproximação de professores, os quais ensinam Matemática, aos conceitos do pensamento computacional que possam ocasionar a Educação

[1] Possui Licenciatura em Matemática pela Universidade do Estado do Rio de Janeiro - UERJ; Bacharelado em Matemática pela Universidade Federal Fluminense - IM/UFF e Mestrado em Matemática pela Universidade Federal do Rio de Janeiro - IM/UFRJ. Atualmente cursa Doutorado em Matemática - IM/UFRJ e é Professor Assistente II na Universidade do Estado do Rio de Janeiro, departamento de Matemática e Desenho, Instituto de Aplicação da Uerj (CAp/Uerj). E-mail: sergio.sousa@uerj.br. ORCID: https://orcid.org/0000-0001-8717-8853.

[2] Mestrado Profissional em Matemática pelo Instituto Nacional de Matemática Pura e Aplicada (2015). Licenciatura em Matemática pela Universidade do Estado do Rio de Janeiro (2002) e Especialização em Matemática para Professores de Ensino Fundamental e Médio pela Universidade Federal Fluminense (2005). Atualmente é Professor Assistente I da Universidade do Estado do Rio de Janeiro (UERJ), lotado no Departamento de Matemática e Desenho do Instituto de Aplicação. Atua também na Rede Municipal de Duque de Caxias, como Professor do Ensino Fundamental II. E-mail: leandro3machado@gmail.com. ORCID: https://orcid.org/0000-0001-9335-3152.

[3] Licenciado em Matemática (UERJ), Mestre em Matemática (UFRJ) e doutor em Engenharia Mecânica (UERJ) é professor doutor do Instituto Federal do Rio de Janeiro. E-mail: moisesceni@gmail.com. ORCID: https://orcid.org/0000-0002-6170-2036.

[4] Professor Ajunto I – Departamento de Matemática e Desenho – CAp/Uerj. Possui graduação em Bacharelado em Matemática pela Universidade Federal Fluminense (2006), graduação em Licenciatura em Matemática pela Universidade Federal Fluminense (2010), mestrado em Ciências da Computação pela Universidade Federal Fluminense (2013) e doutorado em Ciências da Computação pela Universidade Federal Fluminense (2018). E-mail: leonardo.musmanno@gmail.com. ORCID: https://orcid.org/0000-0002-1657-7408.

Matemática Crítica. Em especial, a investigação se direciona ao pensamento algorítmico, por meio da simulação do “Jogo da Corrida dos Cavalos”, atividade lúdico pedagógica proposta por Skovsmose (2000). Logo, sob os pressupostos da Educação Matemática Crítica e, em alinhamento às ideias da Base Nacional Comum Curricular (BNCC), nós construímos um algoritmo, em forma de pseudocódigo, que permite a simulação do jogo. A apresentação da construção do algoritmo é feita de forma investigativa e exploratória, analisando gradativamente as estratégias e processos envolvidos. Por meio da investigação realizada, ressaltamos as potencialidades do pensamento computacional como ferramenta aliada numa possível ação pedagógica do professor que ensina Matemática.
PALAVRAS-CHAVE: Pensamento Computacional. Formação de Professores. Educação Matemática Crítica.

ABSTRACT

This work presents the results of an ongoing investigation, whose objective is to present and explore directions regarding the approach of teachers who teach Mathematics to the concepts of computational thinking, especially algorithmic thinking, which can lead to Critical Mathematics Education, through the simulation of the “Horse Racing Game”, a playful educational activity proposed by Skovsmose (2000). Under the assumptions of Critical Mathematics Education and in line with the ideas of the Common National Curriculum Base (BNCC), we built an algorithm, in the form of a pseudocode, which allows for the simulation of the game. The presentation of the construction of the algorithm is made in an investigative and exploratory way, gradually analyzing the strategies and processes involved. Through the investigation carried out, we emphasize the potential of computational thinking as a tool allied in a possible pedagogical action of the teacher who teaches Mathematics.
KEYWORDS: Computational Thinking. Teacher Education. Critical Mathematical Education.

Introdução

O movimento da Educação Matemática Crítica que emerge no Brasil, na década de 80, aponta para o importante papel que a Matemática deve desempenhar para promover competências democráticas em estudantes do ensino básico em uma sociedade, marcadamente, tecnológica. Essa competência democrática suplanta a capacidade de resolver problemas, no sentido de que as situações de ensino e aprendizagem devem ser abertas e capacitadoras, gerando espaço para troca de ideias, pesquisa e validação ou refutação de resultados.

Como produto do movimento, podemos citar os ambientes de investigação, que se constituem de um conjunto de metodologias que visam proporcionar ambientes de perscrutação em sala de aula, promovendo autonomia e buscando ressignificar as relações de poder no espaço escolar. Nesse contexto, a tecnologia é uma ferramenta que pode auxiliar o desenvolvimento de trabalhos pedagógicos e investigativos, contrapondo-se ao paradigma do exercício.

Dentre os aspectos da tecnologia, o pensamento computacional vem sendo objeto de estudo de pesquisadores em Educação Matemática (BARBOSA, 2019; SILVA *et al*., 2020). Ele possibilita uma maior capacidade em formular, analisar e resolver problemas complexos, podendo essa capacidade ser aplicada às mais variadas áreas do saber. Além disso, auxilia no processo de raciocínio lógico e na

alfabetização digital, bem como promove a autonomia na criação de tecnologias, permitindo que o cidadão deixe de ser apenas um consumidor de recursos digitais e se transforme em um produtor.

A Base Nacional Comum Curricular (BNCC) corrobora essa visão do uso de tecnologias em sala de aula, citando o pensamento computacional e, em particular, os algoritmos, ao propor que:

> os estudantes utilizem tecnologias, como calculadoras e planilhas eletrônicas, desde os anos iniciais do Ensino Fundamental. Tal valorização possibilita que, ao chegarem aos anos finais, eles possam ser estimulados a desenvolver o pensamento computacional, por meio da interpretação e da elaboração de fluxogramas e algoritmos (BRASIL, 2018, p. 513).

Para Borba e Penteado (2002), o caminho para o uso de tecnologias é bastante desafiador e:

> requer mudanças na maneira de interagir com os alunos, no planejamento e desenvolvimento das aulas, na sequência didática, na prontidão para lidar com incertezas, entre outras. As pesquisas mostram que nem sempre essas mudanças acontecem (BORBA; PENTEADO, 2002, p. 24).

Assim, ainda precisam emergir mais pesquisas para a popularização do uso de tecnologias entre professores, em especial do uso de pensamento algorítmico que, na visão dos autores, remete à formulação e reformulação de procedimentos e fluxogramas. Logo, envolve a análise de possíveis erros e a necessidade de experimentação e validação de resultados, de modo que se relaciona com a construção do conhecimento matemático.

Por esses motivos, o presente trabalho tem por objetivo apresentar direcionamentos que possibilitam ao professor de Matemática aproximar-se dos conceitos envolvidos em pensamento computacional, em especial, o pensamento algorítmico. Isso se concretiza por meio da construção de um algoritmo para simulação do “Jogo da Corrida dos Cavalos”, atividade lúdico-pedagógica voltada para o ensino de probabilidade, proposta por Skovsmose (2000).

Uma importante motivação para este artigo provém do fato de os autores já terem trabalhado anteriormente com o “Jogo da Corrida dos Cavalos”, sendo obtidos resultados pedagógicos positivos. Em Machado *et. al* (2020), os autores relataram as experiências realizadas com a atividade e mostraram de que modo o computador pode ser utilizado como uma ferramenta na investigação das questões matemáticas envolvidas no jogo.

> A atividade desenvolvida nos permitiu verificar que nas três turmas testadas, os alunos, de forma prática e lúdica, fixaram e expandiram a sua compreensão de conceitos centrais do tema, como: espaços amostrais, equiprobabilidade, frequência relativa x probabilidade, entre outros (MACHADO *et. al*, 2020, p. 249).

Em Musmanno *et. al* (2021), os autores apresentaram diversos cenários para investigação dentro do jogo, visando à utilização em sala de aula. Na ocasião, os autores implementaram uma simulação computacional do jogo para servir de suporte a possíveis outras investigações que possam surgir em seu contexto. Nos trabalhos anteriores, entretanto, não foi aprofundado o aspecto do pensamento computacional, isso é, os algoritmos implementados foram utilizados como uma estratégia cuja finalidade é a obtenção de respostas para problemas de difícil solução analítica, sem ser realizado nenhum tipo de análise sobre eles.

No presente artigo, aproximamos os professores das ideias envolvidas na implementação do algoritmo para simulação do “Jogo da Corrida dos Cavalos”, com o objetivo de que eles reconheçam as possibilidades didáticas dessa inserção. Portanto, acreditamos que os professores poderão se equipar dessas ferramentas para utilização em uma situação didática. Cabe ressaltar que não pretendemos (e nem seria possível) esgotar todas as possibilidades e potencialidades do pensamento algorítmico.

Assim, nesse trabalho, refletimos e argumentamos sobre a importância da inclusão do pensamento computacional na prática docente. Para isso, nos baseamos em nossos relatos e artigos anteriores e também em textos e produções científicas sobre pensamento computacional, produzindo um ensaio teórico acerca da temática, segundo as concepções de Fiorentini e Lorenzato (2006), obtendo como produto dessas reflexões uma nova abordagem do Jogo da Corrida dos Cavalos. Nesse sentido, é apresentada uma proposta concreta da construção de um algoritmo para o Jogo, em torno do qual discutimos conceitos e ideias do pensamento computacional, em particular do algorítmico, mostrando-o como possibilidade pedagógica para o ensino de matemática, proporcionando oportunidades para a realização de investigações dentro e fora da sala de aula.

Este artigo está organizado da seguinte maneira: na seção Pensamento Computacional, apresentamos a definição desse conceito e destacamos a importância de sua incorporação na educação básica; na seção O “Jogo da Corrida dos Cavalos” e trabalhos anteriores, apresentamos o jogo e os trabalhos anteriormente realizados pelos autores envolvendo essa atividade; na seção Explorando o Pensamento Computacional com o “Jogo da Corrida dos Cavalos”, são apresentados os aspectos

do jogo que podem ser explorados junto aos alunos. Finalmente, na seção Considerações finais, tecemos algumas reflexões sobre o trabalho.

Pensamento Computacional

Sabemos da enorme quantidade de recursos tecnológicos disponíveis atualmente. Os estudantes veem nesses recursos, principalmente, um local, uma fonte para realização de pesquisas e desenvolvimento de tarefas. É, nesse contexto, que surge para a escola o desafio de incorporá-los, utilizá-los e explorá-los de forma crítica, em particular o computador, como uma ferramenta metodológica nos processos de ensino e aprendizagem, de modo que os estudantes possam refletir, conjecturar e realmente aprender.

> A atividade matemática em sala de aula pode ser reorganizada por meio de tarefas baseadas no uso de tecnologias que permitem a elaboração, refutação, e/ou confirmação de conjecturas e a exploração de múltiplas soluções de problemas devido a forma aberta da tarefa. Esse tipo de abordagem baseada em investigação promove o surgimento da cultura do fazer, onde a ênfase é colocada em assumir riscos e aprender por meio de erros em uma comunidade colaborativa (SILVA *et al*, 2020, p. 3, tradução nossa)[5].

No atual ambiente tecnológico em que vivemos, é importante que os estudantes adquiram habilidades e competências que os permitam utilizar com destreza os recursos computacionais disponíveis. O pensamento computacional, como forma de pensamento estruturado, possibilita potencializar as habilidades de manipular e interpretar de forma crítica os recursos tecnológicos, em especial o computador e suas tecnologias.

Assim, o pensamento computacional é, segundo Wing (2006), uma representação do modo de pensar humano. Portanto, como tal, deve ser visto como uma habilidade intelectual básica do ser humano, assim como ler, escrever ou realizar cálculos matemáticos e, por isso, deve ser inserido na educação básica desde as séries iniciais.

França *et al.* (2015) reúnem diversos aspectos do pensamento computacional associados à compreensão e resolução de problemas, a saber:

> i) formular problemas de modo que seja possível usar o computador e outras ferramentas para ajudar a resolvê-los; ii) organizar e analisar dados de forma lógica; iii) representar dados através de abstrações,

[5] "The mathematical activity in classrooms can be reorganized through the development of hands-on tasks based on the use of technologies that allow the elaboration, refutation, and/or confirmation of conjectures and the exploration of multiple solutions of problems regarding the open-ended design of tasks. This kind of inquiry-based approach fosters the emergence of a maker culture, where an emphasis is placed on risk-taking and learning through mistakes in a collaborative community".

> tais como modelos e simulações; iv) automatizar soluções através do pensamento algorítmico; v) identificar, analisar e implementar as soluções possíveis com o objetivo de conseguir a combinação mais eficiente e eficaz de etapas e recursos; e vi) generalizar e transferir esse processo de resolução de problemas para uma grande variedade de problemas (FRANÇA *et al.*, 2015, p. 2).

Para Barbosa (2019), o pensamento computacional apresenta cinco aspectos ou habilidades, a saber: o pensamento algorítmico, decomposição e generalização, padrões e abstração, representação e automação, bem como avaliação.

É importante destacar que o pensamento computacional não representa uma habilidade somente da área da Computação. Ele está presente em diversas áreas do conhecimento como forma de pensamento estruturado e, com isso, promove a representação, a análise e a compreensão dos processos e/ou das situações existentes em um mundo estruturado pela tecnologia de forma articulada, investigativa e contextualizada. Nesse sentido, a BNCC sugere

> identificar, analisar, modelar e solucionar problemas complexos em diversas áreas da vida cotidiana, explorando de forma efetiva o raciocínio lógico, o pensamento computacional, o espírito de investigação e a criatividade (BRASIL, 2018, p. 475).

Por esse motivo, estimular o desenvolvimento do pensamento computacional nos estudantes é um tema que vem ganhando importância nos últimos tempos. A BNCC ratifica a importância da atividade matemática para o desenvolvimento do pensamento computacional ao afirmar que:

> Processos matemáticos como resolução de problemas, de desenvolvimento de projetos, de investigação e de modelagem podem ser citados como formas privilegiadas da atividade matemática. Esses processos de aprendizagem são potencialmente ricos para o desenvolvimento de competências fundamentais para o letramento matemático (raciocínio, representação, comunicação e argumentação) e para o desenvolvimento do pensamento computacional (BRASIL, 2018, p. 266).

Dentre os aspectos mencionados, o presente trabalho focou no desenvolvimento do pensamento algorítmico, definido por Sousa e Lencastre (2014) como “a expressão de soluções em diferentes passos de forma a encontrar a maneira mais eficaz e eficiente de resolver um problema”, e na decomposição.

O pensamento algorítmico está naturalmente presente no ensino dos conteúdos matemáticos, por exemplo, ao trabalhar com divisões mais longas (numéricas ou algébricas), nos processos de fatorações, na resolução por etapas de um problema complexo e etc. A decomposição, por sua vez, é usada como uma estratégia no ensino da matemática para compreensão de problemas complexos que apresentam inicialmente difícil resolução, permitindo que cada parte possa ser

observada separadamente, reduzindo, assim, a complexidade do que precisa ser analisado.

O pensamento algorítmico representa os passos lógicos de um determinado processo, problema ou situação, e indica todas as ações e decisões necessárias que devem ser executadas para se chegar à solução de um dado problema, tendo como produto final o que chamamos de algoritmo. Em um algoritmo, as instruções podem ser escritas em diagramas (fluxogramas), pseudocódigos (linguagem humana) ou em linguagem de programação. Para a BNCC, um algoritmo é:

> uma sequência finita de procedimentos que permite resolver um determinado problema. Assim, o algoritmo é a decomposição de um procedimento complexo em suas partes mais simples, relacionando-as e ordenando-as, e pode ser representado graficamente por um fluxograma (BRASIL, 2018, p. 269).

Como vemos, o pensamento algorítmico pode levar a uma melhor compreensão das ideias presentes na atividade matemática, agrupando e organizando as informações necessárias para obter a solução dos problemas ou de processos matemáticos, incluindo, passo a passo, todas as etapas para chegar na solução.

Dessa forma, o pensamento computacional é um aliado e, até mesmo, uma ferramenta metodológica no processo de ensino dos conteúdos e conceitos matemáticos, capaz de dinamizar o processo de ensino-aprendizagem e contribuir ativamente para o desenvolvimento do trabalho colaborativo, investigativo e interdisciplinar da Matemática, redimensionando, assim, a prática docente.

> As características do pensamento computacional aliadas ao processo das características do fazer e aprender matematicamente valorizam o desenvolvimento de ideias; a resolução de problemas; a reflexão, análise e descrição de hipótese; a formulação criativa de soluções para um dado problema; a construção e aprimoramento de estratégias, indo além da computabilidade; o incentivo à tomada de decisões; a compreensão de fenômenos globais e locais com uso da programação, etc (AZEVEDO; MALTEMPI, 2019, p. 3).

A BNCC preconiza estimular o pensamento computacional ao longo de todos os anos dos ensinos fundamental e médio como uma habilidade para o desenvolvimento da atividade matemática. Desse modo, o pensamento computacional deve fazer parte da formação inicial e continuada dos professores de Matemática, conforme destacam Silva *et al*. (2020): o pensamento computacional é “um componente importante e relevante dentro da formação dos professores de Matemática”.

Barbosa e Maltempi (2020) ressaltam que não basta somente o professor de Matemática conhecer os conceitos e as ideias do pensamento computacional sem construir uma articulação dele com os conceitos e conteúdos matemáticos. Portanto, é necessário sugerir e elaborar atividades formativas e continuadas de ensino para esse fim, de forma que os professores de Matemática possam incorporar o pensamento computacional nas suas práticas de ensino.

É também nesse sentido que propomos a construção dos algoritmos relatados na seção Pensamento Computacional com o “Jogo da Corrida dos Cavalos”: acreditamos que as referências aqui apresentadas demonstram a importância de aprender e desenvolver o pensamento computacional de professores e estudantes. Assim, os algoritmos construídos na seção mencionada aproximam o professor desse conteúdo, em especial do pensamento algorítmico, aumentando sua capacidade de compreender e construir algoritmos.

O "Jogo da Corrida dos Cavalos" e Trabalhos Anteriores

O “Jogo da Corrida dos Cavalos” é utilizado, neste trabalho, como suporte para a apresentação e discussão do pensamento computacional, em alinhamento com o movimento da Educação Matemática Crítica e com a BNCC. Em suma, o jogo consiste em 11 cavalos dispostos em um tabuleiro de 11 colunas numeradas de 2 a 12 e 6 linhas numeradas de 0 a 5, de modo que os cavalos ficam dispostos, inicialmente, na linha 0. Dois dados honestos, de 1 a 6, são lançados simultaneamente e a soma dos números das faces voltadas para cima é anotada. Nessa situação, o cavalo que estiver na coluna cujo número é igual à soma supracitada avança uma linha (vai para a linha imediatamente acima), ganhando o que chegar primeiro na linha 5. Utilizaremos o termo rodada do “Jogo da Corrida dos Cavalos” para fazer referência ao processo de lançar os dados e avançar uma linha com o cavalo correspondente à soma obtida e usaremos o termo partida para se referir a uma execução sucessiva de rodadas, terminada quando um dos onze cavalos é sorteado cinco vezes.

Em “Relato de Experiência: probabilidade no Ensino Médio" (Machado *et al.*, 2020), vemos como a experiência da realização do jogo em escolas públicas foi estimulante para os alunos do ensino básico, que puderam ressignificar suas aprendizagens de conceitos de Probabilidade e Estatística.

> As questões levantadas pelos alunos, durante a execução da atividade, serviram como um aprofundamento do conteúdo ensinado anteriormente. Como exemplo, a discussão acerca do cálculo da probabilidade de um cavalo ganhar a partida auxiliou no entendimento

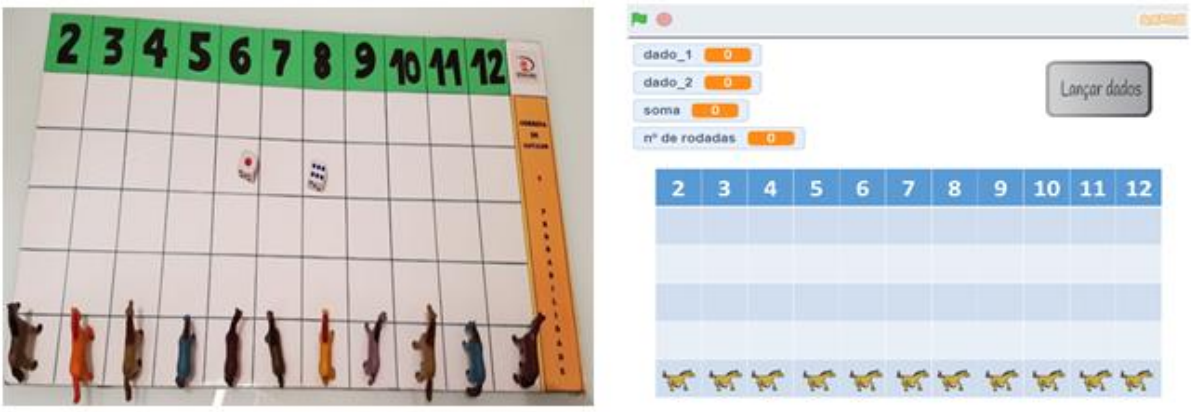
> dos princípios multiplicativos e aditivos, tópicos essenciais na aprendizagem de probabilidades. (MACHADO *et al.*, 2020, p. 249)

Mais recentemente, em “Educação Matemática crítica e uso de tecnologias: cenários para investigação com o jogo da corrida dos cavalos” (Musmanno *et al.*, 2021), os autores apresentaram diversos cenários para investigação a partir do “Jogo da Corrida dos Cavalos”, mostrando como o computador pode ser útil para a construção de uma atividade investigativa em sala de aula.

> Ao propor atividades investigativas com uso de recursos computacionais, buscamos sugerir novas formas de pensar e compreender as soluções dos problemas envolvidos no presente artigo, a fim de desenvolver uma autonomia crítica no pensar e no conjecturar, conduzindo, assim, os sujeitos envolvidos a uma Educação Matemática Crítica numa possível aplicação desses cenários em sala de aula. (MUSMANNO *et al.*, 2021, p. 23)

No contexto do trabalho de Musmanno *et al.* (2021), no entanto, os aspectos computacionais do “Jogo da Corrida dos Cavalos” foram apresentados diretamente, focando no desenvolvimento dos conteúdos matemáticos envolvidos no jogo. Para aquele trabalho, foram produzidas simulações em Scratch (uma versão lúdica) e C++ para explorar outros possíveis cenários de investigação.

Figura 1 – Tabuleiros do "Jogo da corrida dos cavalos” em tabuleiro físico e em versão digital produzida no Scratch.



Fonte: Elaborado pelos autores (2021)[6].

Como exemplo, no artigo mencionado, tratou-se a questão de determinar a probabilidade de um cavalo qualquer ser o vencedor em uma partida do jogo. Como se trata de um problema de difícil solução analítica, o computador é, nesse caso, um dos poucos recursos que podem ser utilizados para facilitar a resolução da questão. A resposta para essa pergunta, bem como para outras questões levantadas, foi obtida

[6] Todos os elementos computacionais utilizados nas análises (algoritmo de simulação e versão digital do jogo) estão disponíveis em nosso blog: https://jogodacorridadoscavalos.blogspot.com/.

por meio da simulação computacional e o sucesso obtido ao se utilizar esse recurso nos mostrou o potencial educacional da ferramenta. Entretanto, muitos professores de matemática ainda apresentam certo desconhecimento sobre o trabalho com o computador, tendo um conhecimento apenas rudimentar acerca dos algoritmos e do pensamento computacional, adquirido em alguma disciplina introdutória da sua formação inicial.

Assim, com o objetivo de apresentar, por meio da construção de um pseudocódigo da simulação do “Jogo da Corrida dos Cavalos”, elementos de pensamento computacional para professores de matemática, em especial o pensamento algorítmico, na seção Pensamento Computacional com o Jogo da Corrida dos Cavalos, serão desenvolvidas investigações e explorações para a construção de uma simulação computacional do jogo. Com isso, esperamos contribuir para que os docentes possam desenvolver um pouco mais de autonomia em trabalhos dessa categoria e eventualmente utilizar recursos semelhantes em suas aulas e pesquisas. Cabe ressaltar que não pretendemos –e nem seria possível– explorar todos os aspectos computacionais necessários à implementação de um jogo em uma linguagem de programação.

Pensamento Computacional com o Jogo da Corrida dos Cavalos

Esta seção está dividida da seguinte maneira: na subseção Conceitos Iniciais, são apresentados alguns conceitos elementares de algoritmos, necessários para a compreensão deste artigo; na subseção Construção do Pseudocódigo, realizamos uma construção gradual do pseudocódigo da simulação computacional do jogo; na subseção Versão Generalizada, estudamos a questão de como construir um algoritmo para uma versão generalizada do jogo, usando como base o pseudocódigo construído na subseção anterior.

Conceitos Iniciais

O primeiro passo na elaboração de um algoritmo é obter a compreensão exata de qual é o problema a ser resolvido, isso é, qual o objetivo final que se deseja atingir. Em seguida, supondo-se que já se saiba resolver o problema em questão, é preciso descrever, de forma clara e precisa, os passos para se chegar à solução. Depois, esses passos devem ser colocados em uma sequência lógica que, ao ser seguida, nos levará, de fato, à solução do problema.

O conceito de variável, no contexto de algoritmos, possui uma importância fundamental, pois um computador é essencialmente uma máquina de entrada e saída

de dados. Assim, dois tipos de dados podem ser definidos: constante e variável. O primeiro é um determinado valor fixo que não se altera até o término do programa e o segundo corresponde a uma posição, na memória do computador, a qual armazena um determinado dado que pode ser acessado e modificado ao longo do programa. Dependendo do tipo de dado que a variável armazena, ela pode ser classificada como numérica (quando armazena um número), caractere (usada para armazenamento de texto) ou lógica (quando assume apenas dois valores: Verdadeiro ou Falso).

As instruções presentes em um algoritmo podem ser de três tipos: sequenciais, comandos que representam ações imperativas, sem nenhum tipo de decisão; de decisão, que representam um desvio no fluxo normal do algoritmo, conforme o resultado de uma expressão lógica; e de repetição, que representam a execução repetitiva de comandos existentes em um desvio no fluxo normal de um programa, governada pelo resultado de uma expressão lógica (SOUZA *et al.*, 2006).

Nos algoritmos deste trabalho, iremos utilizar uma estrutura de decisão, Se-Então, utilizada quando existem instruções que só devem ser executadas se determinada condição for satisfeita. Em relação às estruturas de repetição, duas serão utilizadas: Enquanto-Faça, a qual permite que um certo trecho de programa seja executado enquanto uma certa condição for verdadeira (a estrutura Enquanto-Faça pode ser usada quando não soubermos exatamente quantas vezes o laço deve ser repetido); e a estrutura Para-Faça, que permite que um certo trecho de programa seja executado um determinado número de vezes.

Um vetor é uma variável composta homogênea unidimensional, formada por uma sequência de variáveis, todas do mesmo tipo de dados, com o mesmo identificador (mesmo nome) e alocadas sequencialmente na memória. Por exemplo, mais à frente, em nosso texto, iremos utilizar o vetor $contador[2\ldots12]$, que guarda, dentro de uma partida, quantas vezes cada cavalo foi sorteado (por exemplo, se em algum momento da partida tivermos $contador[3] = 2$, então o cavalo 3 já foi sorteado duas vezes).

Para finalizar a apresentação de conceitos, uma função é um tipo especial de procedimento em que, depois de executada a chamada, o valor calculado é retornado no nome da função, que passa a ser uma variável da expressão. As funções são chamadas dentro do corpo do programa principal como se fossem comandos. Após seu término, a execução continua a partir do ponto onde foi chamado.

Neste trabalho, para a representação dos algoritmos, iremos utilizar o pseudocódigo, uma forma genérica de escrever um algoritmo que utiliza uma

linguagem simples (nativa a quem o escreve, de forma a ser entendida por qualquer pessoa) sem necessidade de conhecer a sintaxe de nenhuma linguagem de programação.

Para mais informações sobre algoritmos, pseudocódigo e programação, cabe consultar (CORMEN *et al.*, 2012; SOUZA *et al.*, 2006). Ademais, em nosso blog, também pode ser encontrado um texto de revisão sobre os conceitos elementares de algoritmos.

Construção do Pseudocódigo

O objetivo desta subseção é apresentar uma construção gradual de um pseudocódigo que simule o "Jogo da Corrida dos Cavalos". Serão criados dois algoritmos: primeiramente, será desenvolvido um algoritmo que simula a realização de uma partida do jogo e, em seguida, um segundo algoritmo permitirá a execução sucessiva de um número qualquer de partidas, visto que, em uma simulação, em geral, deseja-se realizar um número grande de execuções, pois, assim, é possível obter mais dados ou informações sobre o problema.

A partir do "Jogo da Corrida dos Cavalos", vamos construir um algoritmo, em formato de pseudocódigo, para sistematizar o entendimento do jogo como uma sequência lógica de passos a serem executados. Para esse intuito, vamos determinar instruções claras que possam ser entendidas pelo computador, através de laços de repetição, como uma forma de sistematizar o jogo. Esse procedimento confere uma organização e representação ordenada, estimulando o reconhecimento de padrões e o pensamento matemático. Passemos, então, à construção do pseudocódigo de nosso primeiro algoritmo, o qual representará a execução de uma partida do jogo.

Em algoritmos, funções são, muitas vezes, usadas com o objetivo de dar mais organização a um programa. Assim, visando a uma melhor estruturação, iremos realizar a simulação de uma partida por meio de uma função, a qual indicaremos por "$Executa\_partida( )$". O que se espera que essa função devolva? Ao término de uma partida, duas informações devem ser armazenadas: o cavalo vencedor e a rodada em que a partida terminou. Portanto, após sua execução, a função deve retornar essas duas informações. Assim, para armazenar esses dados, criaremos as nossas primeiras variáveis: indicaremos por $vitorioso$ a variável que armazenará o cavalo vencedor da partida e $rodadas$ a que guarda a rodada em que a partida foi finalizada.

Na versão física do jogo, a cada rodada de uma partida, dois dados são lançados. Em relação a isso, iremos simular o sorteio desses dados por meio da

função "$aleat(v_1, v_2)$", que seleciona aleatoriamente um valor inteiro do intervalo $[v_1, v_2]$. Como estamos trabalhando com dados de 6 faces, temos que $v_1 = 1$ e $v_2 = 6$. Para armazenar os valores obtidos na função "$aleat$", utilizaremos as variáveis $dado\_1$ e $dado\_2$. A variável $soma$ irá armazenar a soma dos valores obtidos em $dado\_1$ e $dado\_2$.

Após a computação da soma das faces dos dados, o cavalo correspondente à soma obtida avança uma casa. É preciso, portanto, criar uma variável para representar esse avanço no pseudocódigo. Uma possível solução é engendrar, para cada cavalo, uma variável $contador\_n$, que armazena quantas vezes a soma n já foi sorteada, sendo essa aumentada em uma unidade cada vez que essa soma for obtida. Entretanto, proceder assim envolve a criação de 11 variáveis, uma para cada cavalo. Existe uma forma mais compacta de armazenar essas quantidades? Sim, e nessas ocorrências pode-se usar o conceito de vetor. Assim, para realizar o controle de quantas vezes cada cavalo foi sorteado, pode-se criar um vetor $contador[2 \ldots 12]$, sendo $contador[i]$ aumentado, em uma unidade, cada vez que a soma i for sorteada, para $2 \leq i \leq 12$.

De posse dessas variáveis, a estrutura de uma rodada fica determinada como indicado na Tabela 1:

Tabela 1 – Versão 1 de Executa_partida()

Algoritmo 1 – Versão 1 de Executa_partida()

```
1.      dado_1 ⟵ aleat(1,6);
2.      dado_2 ⟵ aleat(1,6);
3.      soma ⟵ dado_1 + dado_2;
4.      contador[soma] ⟵ contador[soma] + 1;
```

Fonte: Elaborado pelos autores.

Entretanto, uma partida do jogo não consiste em apenas uma rodada, mas sim numa sucessão de rodadas, concluída quando um dos cavalos avança 5 casas. Como representar essa sucessão de rodadas? Uma solução é criar um laço de repetição, terminado quando um dos contadores atinge o valor 5 (condição de parada do laço). Cabe mencionar que há mais de uma forma de verificar essa condição: uma delas é observar que as rodadas prosseguem enquanto nenhum contador atingir 5, assim, é natural utilizar o laço $enquanto$, com a condição de que o máximo dos valores do vetor $contador[2 \ldots 12]$ seja menor que 5. Quando essa condição for falsa, um dos contadores terá atingido o valor 5: isso significa que um cavalo venceu a partida e,

portanto, não há mais necessidade de executar uma rodada. O algoritmo apresentado na Tabela 2 ilustra a discussão:

Tabela 2 – Versão 2 de Executa_partida()

Algoritmo 1 - Versão 2 de Executa_partida()

1. enquanto $\max(\text{contador[j]}) < 5$ faça
2. $\text{dado\_1} \longleftarrow \text{aleat}(1,6)$;
3. $\text{dado\_2} \longleftarrow \text{aleat}(1,6)$;
4. $\text{soma} \longleftarrow \text{dado\_1} + \text{dado\_2}$;
5. $\text{contador[soma]} \longleftarrow \text{contador[soma]} + 1$;
6. fim enquanto

Fonte: Elaborado pelos autores.

A função $Executa\_partida()$ está quase finalizada. Entretanto, as duas informações fundamentais sobre a partida ainda não estão sendo armazenadas. Primeiramente, quantas rodadas ocorreram na partida? (em outras palavras, quantas vezes o laço $enquanto$ foi executado?). Para realizar esse controle sobre a quantidade de rodadas, iremos incluir a variável $rodadas$, já mencionada no início da subseção, que é incrementada em uma unidade cada vez que o laço enquanto é executado.

A segunda informação faltante é ainda mais relevante: qual cavalo foi o vencedor da partida? Após o término do laço enquanto, sabemos que um dos contadores atingiu o valor 5, mas ainda não está sendo verificado qual deles chegou a tal valor. Uma maneira simples de realizar essa verificação é criar um laço do tipo $para$, que percorrerá todo o vetor $contador[2 \ldots 12]$ e testará, para cada i, $2 \leq i \leq 12$, se $contador[i]$ é igual a 5. Em caso afirmativo, a variável $vitorioso$, também mencionada anteriormente, irá armazenar o cavalo vencedor.

Dessa forma, a versão final do Algoritmo 1 é apresentada pela Tabela 3:

Tabela 3 – Versão final de Executa_partida()

Algoritmo 1 - Executa_partida()

1. enquanto $\max(\text{contador[j]}) < 5$ faça
2. $\text{dado\_1} \longleftarrow \text{aleat}(1,6)$;
3. $\text{dado\_2} \longleftarrow \text{aleat}(1,6)$;
4. $\text{soma} \longleftarrow \text{dado\_1} + \text{dado\_2}$;
5. $\text{contador[soma]} \longleftarrow \text{contador[soma]} + 1$;
6. $\text{rodadas} \longleftarrow \text{rodadas} + 1$;
7. fim enquanto
8. para j $\longleftarrow$ 2 até 12
9. se $\text{contador[j]} = 5$ então
10. $\text{vitorioso} \longleftarrow$ j;

11. fim se
12. fim para
13. devolve vitorioso e rodadas;

Fonte: Elaborado pelos autores.

Pela Lei dos Grandes Números (WEN, 1991), quanto maior o número de repetições de um experimento, maior é a proximidade entre a frequência relativa e a probabilidade teórica de um evento. Por esse motivo, será necessária a simulação de um número grande de partidas. Assim, é preciso realizar a criação do segundo algoritmo que a permita. Na construção desse algoritmo, o usuário deve escolher o número de partidas a serem executadas. Dessa forma, iremos utilizar a variável "$TOTAL\_EXEC$" para armazenar esse número.

Uma vez escolhido o número de partidas, será importante conhecer a frequência de partidas vencidas por cada cavalo. Por exemplo, se um milhão de partidas do jogo forem executadas, qual a frequência relativa de vitórias do cavalo 7? E se aumentarmos o número de partidas para cinco milhões, há alguma mudança nessa frequência?

Como visto, a função $Executa\_partida()$ retorna o cavalo vencedor e a rodada em que a partida foi terminada. Essas informações precisam ser registradas após a execução de $Executa\_partida()$, nesse segundo algoritmo. Isso será feito armazenando-se o valor de $vitorioso$ na variável $vencedor$ e o valor de $rodadas$ armazenado em $rodada\_fim$. Essas operações estão indicadas na linha 4 do algoritmo 2.

Para completar esse algoritmo, ainda falta contabilizar quantas vitórias, dentre o total de $TOTAL\_EXEC$ partidas, cada cavalo obteve, e também quantas partidas terminaram em cada rodada.

Novamente, a ideia de vetores se mostra adequada ao nosso propósito. O vetor $Contador\_vencedor[2 \ldots 12]$ armazenará a informação sobre as vitórias dos cavalos: para cada cavalo i, $2 \leq i \leq 12$, o elemento $Contador\_vencedor[i]$ aumenta em uma unidade cada vez que o cavalo i é o vencedor de uma partida. Analogamente, o vetor $Contador\_rodadas[1 \ldots 45]$ registra quantas partidas terminaram na rodada k, com $1 \leq k \leq 45$ (na prática, $5 \leq k \leq 45$, pois não é possível que uma partida termine em menos de 5 rodadas).

Por último, é necessário criar uma variável que irá controlar o número de partidas que já foram simuladas. Representando por "i" essa variável e utilizando as

variáveis e vetores mencionados acima, podemos montar a estrutura desse segundo algoritmo, conforme ilustrado na Tabela 4:

Tabela 4 – Algoritmo 2, Simulação do Jogo da Corrida dos Cavalos

Algoritmo 2 – Simulação do Jogo da Corrida dos Cavalos

1. entrada: TOTAL_EXEC
2. i ⟵ 1
3. enquanto i ≤ TOTAL_EXEC faça
4. (vencedor, rodada_fim) ⟵ Executa_partida();
5. Contador_vencedor[vencedor] ⟵ Contador_vencedor[vencedor] + 1;
6. Contador_rodadas[rodada_fim] ⟵ Contador_rodadas[rodada_fim] + 1;
7. i ⟵ i + 1;
8. fim enquanto

Fonte: Elaborado pelos autores

Os aspectos do pensamento computacional mencionados nas seções anteriores podem ser facilmente identificados no processo de construção desses algoritmos. Por exemplo, para realizar a simulação de um grande número de partidas do jogo, separamos o problema em duas partes: a execução de uma partida, tarefa que foi confiada a uma função, e a criação de um laço para a execução de muitas partidas. Isso ilustra o aspecto da decomposição de um problema.

Versão Generalizada

Na subseção anterior, foi construída e analisada uma simulação do “Jogo da Corrida dos Cavalos”. Para essa simulação, nos baseamos nas regras do jogo proposto por Skovsmose (2000), em que existiam 11 cavalos e o cavalo vencedor precisava andar 5 casas. Suponha que, agora, desejamos analisar como implementar uma versão generalizada desse jogo, isso é, um jogo composto por m cavalos, sendo necessário andar n casas até a vitória (manteremos apenas 2 dados). O que deve ser modificado no algoritmo para executar essa versão generalizada?

Para construir um pseudocódigo para essa versão, podemos utilizar muitos passos dos algoritmos anteriores. Entretanto, algumas adaptações precisam ser realizadas. Primeiramente, os valores dos parâmetros m e n devem ser definidos pelo usuário e passados como entrada, como indicado no algoritmo Gen. 1 abaixo. Em seguida, esses valores devem ser armazenados nas variáveis m e n, conforme apresentado na Tabela 5:

Tabela 5 – Algoritmo Generalizado 1 – Simulação da versão generalizada do Jogo da Corrida dos Cavalos

Algoritmo Gen.1 – Simulação da versão generalizada do Jogo da Corrida dos Cavalos

```
1. Entrada: n_cavalos, n_casas
2. m ⟵ n_cavalos;
3. n ⟵ n_casas
4. i ⟵ 1
5. enquanto i ≤ TOTAL_EXEC faça
6.    (vitorioso, rodadas) ⟵ Executa_partida(m, n);
7.    Contador_vencedor[vitorioso] ⟵ Contador_vencedor[vitorioso] + 1;
8.    Contador_rodadas[rodadas] ⟵ Contador_rodadas[rodadas] + 1;
9.    i ⟵ i + 1;
10. fim enquanto
```

Fonte: Elaborado pelos autores

Observe também que a função que executava uma partida no Algoritmo 1, $Executa\_partida()$, não tinha nenhum parâmetro, uma vez que a definição de número de cavalos e casas era fixa. Nessa versão generalizada, entretanto, é preciso que a função seja informada sobre o número de cavalos e casas do jogo. Assim, a função de execução de uma partida é agora representada por $Executa\_partida(m, n)$, como indicado na linha 6.

Quando se modifica o número de cavalos, é preciso modificar também os dados utilizados. Para simplificar, vamos supor que o valor de m é ímpar. Anteriormente, os dados funcionavam selecionando um número aleatório entre 1 e 6. Agora, com m cavalos, precisamos que os dados sejam numerados de 1 até $(m+1)/2$, como indicado nas linhas 2 e 3 do Algoritmo Gen. 2, visto na Tabela 6:

Tabela 6 – Algoritmo Generalizado 2 – Executa_partida(m, n)

Algoritmo Gen. 2 - Executa_partida(m, n)

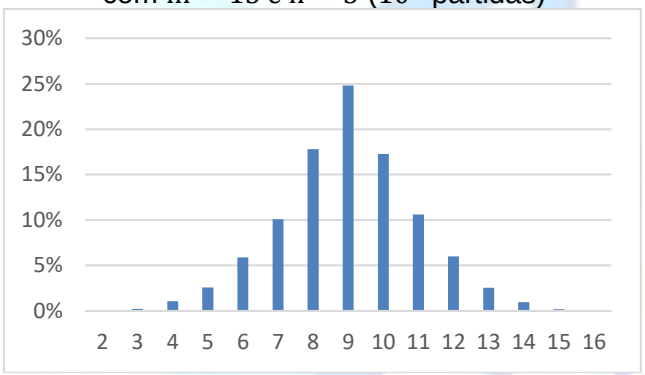
```
1. enquanto max(contador[j]) < n faça
2.    dado_1 ⟵ aleat(1, (m + 1)/2);
3.    dado_2 ⟵ aleat(1, (m + 1)/2);
4.    soma ⟵ dado_1 + dado_2;
5.    contador[soma] ⟵ contador[soma] + 1;
6.    rodadas ⟵ rodadas + 1;
7. fim enquanto
8. para j ⟵ 2 até m
9.    se contador[j] = n então
10.      vitorioso ⟵ j;
11.   fim se
12. fim para
13. devolve vitorioso e rodadas;
```

Fonte: Elaborado pelos autores

Também é necessário fazer mudanças nos contadores de cada cavalo. Anteriormente, a condição de parada do laço enquanto era representada por $(contador[j]) < 5$, uma vez que era preciso andar 5 casas para vencer uma partida. Agora, o parâmetro n representa o número de casas até a vitória, sendo preciso, portanto, modificar a condição de parada para $(contador[j]) < n$, como indicado na linha 1 do algoritmo Gen.2. Da mesma forma, o cavalo vencedor é aquele cujo contador atingir o valor n, como apresentado na linha 9.

Chamamos a atenção para o fato de que a implementação dessas generalizações permite a exploração de diversas questões probabilísticas. Em Musmanno *et al.* (2021), foram analisadas questões relativas à versão normal do jogo (isto é, com $m = 11\ e\ n = 5$). A generalização acima permite que sejam realizadas análises semelhantes nessas novas versões do jogo. Por exemplo, para o caso do jogo formado por 15 cavalos e 5 casas até a vitória ($m = 15\ e\ n = 5$), a implementação dos algoritmos generalizados nos permite verificar a frequência de vitórias de cada um dos cavalos, como indicado na Figura 2:

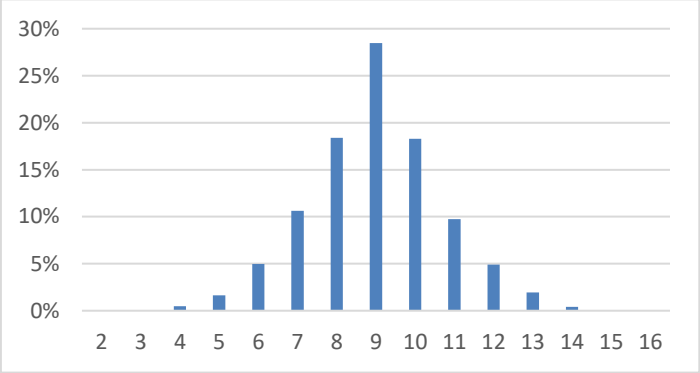
Figura 2 – Frequência de vitórias
com $\mathrm{m} = 15$ e $\mathrm{n} = 5$ (10^6 partidas)



Fonte: Elaborado pelos autores (2021)

Na Figura 3, são apresentadas as frequências de vitórias para cada um dos 15 cavalos, agora, entretanto, precisando andar 7 casas até a vitória.

Figura 3 – Frequência de vitórias
com $\mathrm{m} = 15$ e $\mathrm{n} = 7$ (10^6 partidas)



Fonte: Elaborado pelos autores (2021)

A partir desses gráficos, diversas análises podem ser realizadas. Por exemplo, os gráficos apresentados têm algum tipo de simetria? Caso tenham, porque essa simetria ocorre? O que explica o aumento da frequência de vitórias do cavalo 9 na Figura 3?

Dessa forma, as generalizações do “Jogo da Corrida dos Cavalos” permitem que novas investigações sejam realizadas, podendo se tornar uma ferramenta para o aprimoramento do pensamento computacional, bem como um convite à investigação de questões envolvendo probabilidade.

Considerações Finais

Os recursos tecnológicos estão cada vez mais presentes em nossa sociedade. É preciso, portanto, que esses sejam incluídos e utilizados a favor de alunos e professores também na educação básica. Na Educação Matemática, em particular, a tecnologia pode ser uma importante ferramenta para o desenvolvimento de trabalhos colaborativos e investigativos, favorecendo uma maior autonomia do pensar dos sujeitos envolvidos.

Neste trabalho, dirigimos nossa atenção para os aspectos do pensamento computacional e sua relação com a tecnologia. A exploração desse conceito se deu por meio da construção de algoritmos relacionados à atividade lúdico-pedagógica denominada "Jogo da Corrida dos Cavalos", proposta por Skovsmose (2000). Mais especificamente, foi realizada a construção de um algoritmo, em pseudocódigo, que permite a simulação desse jogo, assim como um algoritmo que simula uma versão generalizada do jogo, na qual o próprio usuário pode escolher o número de participantes e o número de casas até a vitória. Em ambas as construções, não apresentamos o algoritmo pronto, já acabado, mas sim procuramos apresentar o processo de pensamento envolvido na elaboração deles.

Com a crescente demanda pela inclusão do pensamento computacional no currículo escolar, os professores de Matemática, em especial, devem procurar adquirir

conhecimentos que os capacitem a ensinar esse conteúdo. Este artigo, ao apresentar de forma didática alguns elementos de pensamento algorítmico, oferece uma possibilidade, um convite, ao estudo do pensamento computacional.

Naturalmente, para que o professor esteja habilitado a utilizar conteúdos de pensamento computacional em sala de aula, é preciso realizar estudos mais aprofundados, os quais permitirão que ele possa transmitir esse conhecimento de forma segura em suas aulas. Considerando-se os benefícios pedagógicos mencionados na literatura, acreditamos que o esforço nesse sentido é válido e pode tornar a Matemática mais significativa para os estudantes de nossa sociedade altamente tecnológica.

Referências

AZEVEDO, Greiton Toledo de; MALTEMPI, Marcus Vinícius; Processo de Aprendizagem de Matemática à luz das Metodologias ativas e do Pensamento Computacional. **Ciência e Educação**, v. 26, p. 1-18, 2020.

BARBOSA, Luciana Leal da Silva; MALTEMPI, Marcus Vinícius; Matemática, Pensamento Computacional e BNCC: desafios e potencialidades de ensino e das tecnologias na formação inicial de professores. **Revista Brasileira de Ensino de Ciências e Matemática**, v. 3, n. 3, p. 748-776, 2020.

BARBOSA, Lara Martins; **Aspectos do Pensamento Computacional na Construção de Fractais com o software GeoGebra.** Dissertação de Mestrado em Educação Matemática. Universidade Estadual Paulista (Unesp), Instituto de Geociências e Ciências Exatas, Rio Claro: 2019.

BORBA, Marcelo de Carvalho, PENTEADO, Miriam Godoy; Pesquisas em informática e educação matemática. **Educação em Revista**, n.36, p. 239-253. 2002.

BRASIL. Ministério da Educação. Secretaria de Educação Básica. **Base Nacional Comum Curricular**: ensino médio. Brasília: MEC, 2018.

CORMEN, Thomas; LEISERSON, Charles; RIVEST, Ronald; STEIN, Clifford; **Algoritmos**: Teoria e Prática. 3a edição. Elsevier, 2012.

FIORENTINI, Dario; LORENZATO, Sergio; **Investigação em Educação Matemática**: percursos teóricos e metodológicos. Campinas: Autores Associados, 2006.

FRANÇA, Rozelma; TEDESCO, Patrícia; Explorando o pensamento computacional no ensino médio: do design à avaliação de jogos digitais. In: WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI), 23, 2015, Recife. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2015. p. 61-70.

FRANÇA, Rozelma; SILVA, Waldir; AMARAL, Haroldo José Costa do; Ensino de ciência da computação na educação básica: Experiências, desafios e possibilidades. In: **Anais do XX Workshop sobre Educação em Computação.** SBC, 2012.

MACHADO, Leandro da Silva; MUSMANNO, Leonardo Maricato; ALMEIDA, Moisés Ceni de; SOUSA, Sérgio Gonçalves de; Relato de Experiência: probabilidade no Ensino Médio. **Educação Matemática em Revista**, v. 25, n. 66, p. 239-251, 2020.

MUSMANNO, Leonardo Maricato; SOUSA, Sérgio Gonçalves de; ALMEIDA, Moisés Ceni de; MACHADO, Leandro da Silva; Educação Matemática crítica e uso de tecnologias: cenários para investigação com o jogo da corrida dos cavalos. **Revista de Ensino de Ciências e Matemática**, v. 12, n. 3, p. 1-24, 2021.

SILVA, Ricardo Scucuglia Rodrigues da; GADANIDIS, George; RONDINI, Carina Alexandra; BORBA, Marcelo de Carvalho; HUGHES, Janette; Sensitive-Computational Thinking of Pre-Service Mathematics Teachers on Nested Loops. **Perspectivas da Educação Matemática**, v. 13, n. 32, p. 1-18, 2020.

SKOVSMOSE, Ole. Cenários para investigação. Tradução de Jonei Cerqueira Barbosa. **Bolema**, Rio Claro, v. 13, n. 14, p. 66-91, 2000.

SOUSA, Rui Miguel; LENCASTRE, José Alberto; Scratch: uma opção válida para desenvolver o pensamento computacional e a competência de resolução de problemas. In: 2º ENCONTRO SOBRE JOGOS E MOBILE LEARNING, 2014, Braga. **Anais**... Braga: 2014, p.256 -267.

SOUZA, Marco Furlan de; GOMES, Marcelo Marques; SOARES, Márcio Vieira; CONCILIO, Ricardo; **Algoritmos e Lógica de programação**, 1a edição. São Paulo,Thomson Learning, 2006.

WEN, Liu; An Analytic Technique to Prove Borel's Strong Law of Large Numbers. **The American Mathematical Monthly**. Londres. v. 98, n. 2, p. 146-148, 1991.

WING, Jeannette; Computational Thinking. **Communications of the ACM**, v. 49, p. 33-35, 2006.

Submetido em junho de 2021.

Aceito em março de 2022.