


AS STEM E O PENSAMENTO COMPUTACIONAL: RESOLVENDO DESAFIOS DA VIDA REAL NO ENSINO SUPERIOR

STEM AND COMPUTATIONAL THINKING: SOLVING REAL-LIFE PROBLEMS IN HIGHER EDUCATION

PENSAMIENTO STEM Y COMPUTACIONAL: SOLUCIÓN DE PROBLEMAS DE LA VIDA REAL EN LA EDUCACIÓN SUPERIOR

Maria Cristina Costa* 

Sandra Gaspar Martins** 

António Domingos*** 

RESUMO

A promoção da educação STEM (Science, Technology, Engineering, and Mathematics) tem sido cada vez mais defendida, não só para motivar os estudantes para estas áreas, mas também para os preparar para os desafios do mundo real. Em particular, tem sido recomendada para promover a aprendizagem da matemática, muitas vezes acusada de contribuir para o problema da insuficiência de licenciados em áreas relacionadas com as STEM. Por outro lado, o Pensamento Computacional é um processo cognitivo que pode ser aplicado num contexto STEM e que requer capacidade de resolução de problemas, assim como pode contribuir para a aprendizagem da matemática. Este artigo apresenta o projeto CarRace que foi implementado numa turma de matemática de alunos de engenharia informática, entre os anos letivos 2017/18 e 2021/22; dele participaram um total de 581 estudantes. Com uma metodologia qualitativa, seguindo um paradigma interpretativo, conclui-se que o projeto promoveu o Pensamento Computacional nos estudantes, num contexto de educação STEM. Os resultados obtidos mostram a importância de implementar este tipo de abordagem com estudantes do ensino superior, nomeadamente estudantes de engenharia.

Palavras-chave: Educação matemática. Educação STEM. Ensino Superior. Pensamento computacional. Engenharia.

* Doutorada em Ciências da Educação pela Universidade Nova de Lisboa (UNL). Docente do Instituto Politécnico de Tomar (IPT), Tomar, Portugal. Endereço para correspondência: IPT, Estrada da Serra, Campus da Quinta do Contador 2300-313 Tomar, Portugal. Membro do centro de investigação em Cidades Inteligentes do IPT e colaboradora no Centro Interdisciplinar de Ciências Sociais (SICS.NOVA) da Universidade Nova de Lisboa (UNL). Diretora da Academia da Ciência, Arte e Património (www.academiacap.ipt.pt). E-mail: ccosta@ipt.pt.

** Doutorada em Ciências da Educação pela Universidade Nova de Lisboa (UNL). Docente do Instituto Superior de Engenharia de Lisboa (ISEL), Lisboa, Portugal. Endereço para correspondência: Rua Conselheiro Emídio Navarro 1, 1959-007 Lisboa, Portugal. Membro do centro de investigação Centro Interdisciplinar de Ciências Sociais (SICS.NOVA) da Universidade Nova de Lisboa (UNL). E-mail: sandra.gaspar.martins@isel.pt.

*** Doutorado em Ciências da Educação pela Universidade Nova de Lisboa (UNL). Docente da Faculdade de Ciências e Tecnologia (FCT) da Universidade Nova de Lisboa (UNL), Caparica, Portugal. Endereço para correspondência: Faculdade de Ciências e Tecnologia, 1959-007 Caparica, Portugal. Membro do centro de investigação Centro Interdisciplinar de Ciências Sociais (SICS.NOVA) da Universidade Nova de Lisboa (UNL). E-mail: amdd@fct.unl.pt.

ABSTRACT

The promotion of STEM (Science, Technology, Engineering, and Mathematics) education has been increasingly advocated, not only to motivate students towards these areas, but also to prepare them for the challenges of the real world. In particular, it has been recommended to promote the learning of mathematics, often accused of contributing to the problem of a shortage of STEM graduates. On the other hand, Computational Thinking is a cognitive process that can be applied in a STEM context and requires problem-solving skills, as well as contributing to mathematics learning. This paper presents the CarRace project that was implemented in a math class of computer engineering students between the academic years 2017/18 and 2021/22, with a total of 581 students taking part. With a qualitative methodology, following an interpretive paradigm, it is concluded that the project promoted Computational Thinking in students, in a STEM education context. The results show the importance of implementing this type of approach with higher education students, particularly engineering students.

Keywords: Mathematics education. STEM education. Higher education. Computational thinking. Engineering.

RESUMEN

Se ha abogado cada vez más por la promoción de la educación STEM (Ciencia, Tecnología, Ingeniería y Matemáticas), no sólo para motivar a los estudiantes hacia estas áreas, sino también para prepararlos para los desafíos del mundo real. En particular, se ha recomendado promover el aprendizaje de las matemáticas, a menudo acusadas de contribuir al problema de la insuficiencia de graduados en áreas relacionadas con STEM. Por otro lado, el Pensamiento Computacional es un proceso cognitivo que se puede aplicar en un contexto STEM y que requiere habilidades de resolución de problemas, además de contribuir al aprendizaje de las matemáticas. Este artículo presenta el proyecto CarRace, que se implementó en una clase de matemáticas de estudiantes de ingeniería informática, entre los cursos 2017/18 y 2021/22, con un total de 581 estudiantes participantes. Utilizando una metodología cualitativa, siguiendo un paradigma interpretativo, se concluye que el proyecto promovió el Pensamiento Computacional en los estudiantes, en un contexto de educación STEM. Los resultados obtenidos muestran la importancia de implementar este tipo de enfoque con estudiantes de educación superior, particularmente estudiantes de ingeniería.

Palabras clave: Educación matemática. Educación STEM. Educación Superior. Pensamiento computacional. Ingeniería.

1 INTRODUÇÃO

Existe um consenso na comunidade internacional sobre a importância de promover a educação STEM (Science, Technology, Engineering, and Mathematics), não só para motivar os alunos a aprender sobre estas áreas do conhecimento, mas também para os preparar para os desafios do mundo real (STOHLMANN, 2018). No entanto, a literatura refere que os estudantes têm escassez de competências nos domínios STEM e que a matemática contribui para o problema da insuficiência de licenciados em STEM (BESWICK; FRASER, 2019). Portanto, é crucial encontrar abordagens para enfrentar essas questões, nomeadamente destacando o papel

da Matemática em STEM (STOHLMANN, 2018).

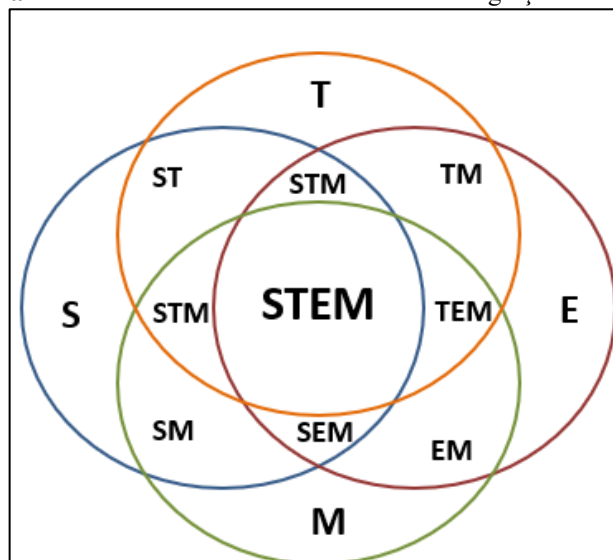
Por outro lado, o Pensamento Computacional (PC) é um processo cognitivo que pode ser aplicado a domínios STEM que requerem competências de resolução de problemas, podendo também contribuir para ajudar a aprender matemática (LI *et al.*, 2020; LU *et al.*, 2022). Em particular, os estudantes de engenharia informática necessitam de ter competências de programação, nomeadamente recorrendo à matemática. Este artigo apresenta o projeto CarRace que foi implementado, por uma professora de matemática, numa turma de cálculo matemático de alunos de engenharia informática, entre os anos letivos 2017/18 e 2021/22. Participaram no projeto um total de 581 alunos. O objetivo do projeto era fazer com que um carro percorresse um circuito utilizando a matemática num código Python (BEAZLEY, 2006). A questão de investigação que orienta este estudo é: *O projeto CarRace promove o Pensamento Computacional nos estudantes no contexto da educação STEM?*

Nas secções seguintes, começa-se por apresentar o quadro teórico com foco na educação STEM e no Pensamento Computacional. Posteriormente, descreve-se o projeto em causa e a metodologia de investigação. Por fim, apresenta-se a análise e discussão de dados e finaliza-se com as conclusões.

2 REFERÊNCIAL TEÓRICO

Apesar de não haver um consenso na comunidade internacional sobre a definição de STEM, a tendência mais recente é para uma abordagem cada vez mais interdisciplinar incluindo Ciência, Tecnologia, Engenharia e Matemática (AGUILERA *et al.*, 2021; COSTA *et al.*, 2020; LI *et al.*, 2020). Neste sentido, Costa e Domingos (2023) defendem uma abordagem integrada das STEM, onde as várias áreas disciplinares incluídas no acrónimo surgem no decorrer da resolução de problemas que surgem em contextos da vida real (Figura 1).

Figura 1 – As áreas incluídas no acrónimo e a integração das STEM.



Fonte: Costa & Domingos (2023, p. 90)

A Figura 1 apresenta diferentes combinações das quatro áreas disciplinares (Ciências, Tecnologia, Engenharia e Matemática) que surgem na literatura, estando no centro a dimensão STEM que resulta da integração das referidas áreas. Neste contexto, entende-se por abordagem integrada das STEM, aquela em que as tarefas introduzidas ou trabalhadas incluam conteúdos das quatro áreas disciplinares de forma integrada, na sequência da resolução de problemas da vida real (COSTA *et al.*, 2022).

A matemática desempenha um papel crucial na educação STEM, com cada vez mais autores a defenderem que lhe deve ser dada maior relevância neste contexto (STOHLMANN, 2018). No entanto, também é acusada de contribuir para a falta de profissionais em STEM, devido ao facto de o seu ensino ser pouco inspirador (BESWICK; FRASER, 2019). Assim, a educação STEM pode ser uma forma de inovar na educação matemática (FITZALLEN, 2015), bem como de melhorar o desempenho nesta disciplina (STOHLMANN, 2018). Além, disso, também poderá ajudar à compreensão de fenómenos da vida real, motivando assim os alunos para a sua importância (COSTA *et al.*, 2022). Becker e Park (2011) referem que as abordagens que envolvem a integração de tópicos relacionados com as STEM têm efeitos positivos no desempenho dos alunos, com melhores resultados ao nível do ensino básico.

São cada vez mais as referências sobre a necessidade de adaptar o ensino às novas gerações, as quais cresceram com cada vez mais recursos tecnológicos, sendo de esperar que tenham a capacidade de resolver problemas reais, tal como se espera, por exemplo, de um

engenheiro (RESNICK *et al.*, 2009). Neste contexto, a programação assume um papel cada vez mais relevante, não só nas profissões mais recentes, mas também a nível da aprendizagem desde o ensino básico (HEINTZ *et al.*, 2017; RESNICK *et al.*, 2009). Além disso, programação pode também ser uma forma de motivar os estudantes para a aprendizagem da matemática (ABRAMOVICH, 2013; JOHNSON, 2000), daí ser recomendada a sua inclusão no currículo das escolas (ZIATDINOV; VALLES, 2022).

De facto, a programação surge frequentemente para resolver problemas da vida real, em particular em cursos de engenharia, onde são necessárias competências de pensamento lógico, crítico e computacional, sendo esta uma forma de motivar os estudantes para a aprendizagem (MALICKY *et al.*, 2006), assim como também para os preparar as carreiras futuras (DE VERE, 2009). Por sua vez, a programação está relacionada com o pensamento computacional que envolve um raciocínio com abordagens analíticas e algorítmicas (BOCCONI *et al.*, 2016; WEINTROP *et al.*, 2016; WING, 2006). Neste contexto, a Matemática e as ciências da computação ligam-se através de algoritmos com sequências de instruções criadas de forma estruturada e respeitando diversos passos para cumprir objetivos (ALBUQUERQUE, 2022).

Figura 2 – Pensamento computacional.



Fonte: Elaborada pelos autores do artigo.

A Figura 2 apresenta algumas fases do pensamento computacional (WEINTROP *et al.*, 2016; WING, 2006), em que destacamos a abstração, a decomposição, o reconhecimento de padrões e a algoritmia. A abstração compreende a identificação de normas gerais a adotar, a decomposição envolve a sintetização de um dado problema e sua decomposição em passos mais simples, o reconhecimento de padrões permite a identificação de regularidades ou relações entre objetos, e a algoritmia cria passos para resolver problemas. A estas quatro fases, ainda se acrescenta a depuração, que tem a ver com a identificação e correção de erros de resolução, os quais têm a ver com a verificação e otimização da solução obtida (WING, 2006).

Com efeito, com base neste enquadramento, o Pensamento Computacional é promovido se o estudante conseguir aplicar estas fases no decorrer de tarefas de resolução de problemas, por exemplo na aula de matemática (WEINTROP *et al.*, 2016).

3 METODOLOGIA

Esta secção está dividida em duas subsecções. Na primeira dá-se conta do projeto desenvolvido com os estudantes. Na segunda, é referida a metodologia usada na investigação.

3.1 O projeto CarRace

Nesta secção começa-se por dar conta do projeto desenvolvido com os estudantes, seguido dos conteúdos matemáticos trabalhados.

O projeto CarRace mostra uma aplicação da matemática a um problema de engenharia com recurso à programação em Python. Neste sentido, os conteúdos matemáticos foram trabalhados de forma interdisciplinar com a programação, através da aprendizagem com recurso a problemas reais. O projeto colocado aos estudantes consistia em fazer um carro realizar uma trajetória num dado circuito a ser escolhido, respeitando algumas regras. A Figura 3 apresenta o exemplo de um circuito apresentado aos estudantes. Para o desenvolvimento dos projetos individuais, cada um deveria escolher um circuito ao seu critério, desde que incluísse uma parte de uma reta ou de uma parábola e também uma parte de um círculo ou uma elipse, assim como escolher um carro para percorrer a pista.

Figura 3: Imagem de um carro a percorrer um circuito.



Fonte: Elaborada pelos autores do artigo

O objetivo do projeto era fazer o carro executar a trajetória do circuito escolhido, usando parametrizações, num ficheiro Python, fornecido pelo professor, que continha um código no qual um carro percorre a trajetória cujo exemplo é dado na Figura 3. Os estudantes já tinham aprendido a utilizar a linguagem Python e Pygame numa outra unidade curricular de programação, pelo que já os tinham instalados no seu computador. A professora que implementou o CarRace com os estudantes na aula de matemática explicou como funcionava o programa Python, que servia de exemplo para o projeto proposto, numa apresentação com cerca de 30 minutos.

Posteriormente, a professora solicitou aos estudantes darem início ao projeto nos 60 minutos seguintes. Para desenvolverem o projeto, eles precisavam interpretar o código e conseguir adaptar as parametrizações, de forma a que o carro executasse a referida corrida, de acordo com as regras estabelecidas. Por exemplo, o carro tinha que percorrer partes de círculos, retas, elipses, parábolas, presentes no circuito escolhido sem sair da referida pista. Isto só era possível se os estudantes conseguissem dominar os conteúdos matemáticos em causa, identificando-os no código dado e reescrevendo o código apresentado a título de exemplo.

Uma vez realizado o projeto, os estudantes tinham que entregar à professora ficheiros com o carro, o circuito e o ficheiro Python que faziam com que o carro descrevesse um movimento correto na pista escolhida. Além disso, deviam produzir um vídeo com o carro a percorrer a pista. No final é feita uma discussão do projeto com o professor da unidade curricular em causa de forma a aferir a aprendizagem sobre o mesmo.

3.2 Metodologia de investigação

Para responder à questão de investigação, optámos por uma metodologia qualitativa com uma abordagem interpretativa (COHEN; LAWRENCE; KEITH, 2007). Os dados foram recolhidos a partir dos projetos desenvolvidos e entregues pelos alunos, assim como por meio da observação participante por parte do professor da unidade curricular durante o desenvolvimento do projeto e a discussão do mesmo.

Os participantes são estudantes da unidade curricular de Matemática Aplicada à Engenharia (Análise), da Licenciatura em Engenharia Informática, do Instituto Superior de Engenharia de Lisboa, pertencente ao Instituto Politécnico de Lisboa, que realizaram o projeto CarRace desde o ano letivo 2017/18 a 2021/22 (Tabela 1). Como se pode verificar, estiveram

inscritos 581 estudantes ao fim dos cinco anos letivos, e 430 entregaram o projeto finalizado.

Foi realizada uma análise documental a partir dos trabalhos entregues pelos estudantes, durante os referidos anos letivos, procurando aferir as suas aprendizagens na realização dos mesmos, no que diz respeito aos conteúdos de matemática usados para resolver o problema colocado no CarRace.

Tabela 1 – Número de alunos inscritos que entregaram o CarRace

Alunos	2017	2018	2019	2020	2021	Total
Inscritos	110	113	123	121	114	581
CarRace entregue	95	86	99	62	88	430

Fonte: elaborada pelos autores.

Na Tabela 1, encontra-se o número de alunos inscritos em cada um dos anos letivos, assim como o número de alunos que entregaram o projeto.

4 ANÁLISE E RESULTADOS

No projeto CarRace, o desafio consistia em fazer um carro percorrer um circuito, a escolher ou desenhar pelos estudantes, utilizando parametrizações inseridas num código Python. Trata-se, portanto, de um problema da vida real, nomeadamente de um problema de engenharia, cuja resolução envolve uma abordagem interdisciplinar não só com recurso à matemática e à programação, mas também a conceitos de física, entre outros.

A programação em Python foi abordada recorrendo à biblioteca Pygame, indicada para desenvolver jogos. Relativamente à matemática, são usadas parametrizações de linhas em \mathbb{R}^2 , nas quais as velocidades dos carros têm que ser adequadas, de forma a que os mesmos andem mais devagar nas curvas e com maior velocidade nas retas. Na parametrização das elipses e circunferências usa-se a parametrização recorrendo a coordenadas polares, as quais precisam de aprender, dado que os estudantes ainda não tinham estudado este conteúdo.

Abaixo, estão exemplificadas parametrizações de um segmento de reta, de uma parte de uma circunferência, de uma parte de uma elipse e de uma parte de uma parábola.

No caso do segmento de reta, tem-se a função f de \mathbb{R} em \mathbb{R}^2 , com

$$f(t) = (250, 340 - 250(t - 4)), \text{ onde } t \in]4; 4.4].$$

Sabemos que para calcular a velocidade se utiliza a derivada

$$f'(t) = (0, -250)$$

e depois calcula-se a norma:

$$\|(0, -250)\| = \sqrt{0^2 + (-250)^2} = 250$$

O que nos permite concluir que a velocidade do carro é constante e igual a 250 unidades.

No caso de parte de uma circunferência, tem-se, por exemplo, a função f de \mathbb{R} em \mathbb{R}^2 , com

$$f(t) = (355 + 110 \times \cos(-2 + t), 190 + 110 \times \sin(-2 + t)), \text{ onde } t \in]0.5; 3.64].$$

Para calcular a velocidade temos que calcular a derivada

$$f'(t) = (-110 \sin(-2 + t), 110 \cos(-2 + t))$$

depois, calculando a norma:

$$\begin{aligned} \|(-110 \sin(-2 + t), 110 \cos(-2 + t))\| &= \\ &= \sqrt{110^2 (\sin(-2 + t))^2 + 110^2 (\cos(-2 + t))^2} \\ &= \sqrt{110^2} = 110. \end{aligned}$$

Portanto a velocidade na circunferência é constante e igual a 110 unidades.

No caso de uma parte de uma elipse, um pouco mais complexa, tem-se, por exemplo, a função f de \mathbb{R} em \mathbb{R}^2 , com

$$f(t) = (355 + 110 \times \cos(-2 + t), 190 + 150 \times \sin(-2 + t)), \text{ onde } t \in]0.5; 3.64],$$

Para calcular a velocidade começamos por calcular a derivada,

$$f'(t) = (-110 \sin(-2 + t), 150 \cos(-2 + t))$$

depois, calculando a norma:

$$\begin{aligned} \|(-110 \sin(-2 + t), 150 \cos(-2 + t))\| &= \\ &= \sqrt{110^2 (\sin(-2 + t))^2 + 150^2 (\cos(-2 + t))^2} \\ &= \sqrt{12100 (\sin(-2 + t))^2 + 22500 (\cos(-2 + t))^2} \\ &= \sqrt{12100 (\sin(-2 + t))^2 + (12100 + 10400) (\cos(-2 + t))^2} \\ &= \sqrt{12100 (\sin(t - 2))^2 + 12100 (\cos(t - 2))^2 + 10400 (\cos(t - 2))^2} \\ &= \sqrt{12100 + 10400 (\cos(t - 2))^2} \end{aligned}$$

Como $0 \leq (\cos(x))^2 \leq 1$,

$$110 = \sqrt{12100} \leq \sqrt{12100 + 10400 (\cos(t - 2))^2} \leq \sqrt{22500} = 150$$

Portanto a velocidade na elipse varia entre 110 e 150 unidades.

Agora num exemplo de uma parábola:

$$f(t) = (100t, 100(t - 30)^2), \text{ onde } t \in [10, 20]$$

Logo

$$f'(t) = (100, 200(t - 30))$$

Donde

$$\|f'(t)\| = \sqrt{100^2 + 200^2 (t - 30)^2} = \sqrt{10000 + 40000 (t - 30)^2}$$

Como $t \in [10, 20]$, então $t - 30 \in [-20, -10]$, logo $(t - 30)^2 \in [100, 400]$. Portanto,

$$\sqrt{10000 + 40000 \times 100} \leq \sqrt{10000 + 40000 \times (t - 30)^2} \leq \sqrt{10000 + 40000 \times 400}$$

ou seja,

$$2002 \leq \sqrt{10000 + 40000 \times (t - 30)^2} \leq 4001$$

Portanto a velocidade nesta parábola varia entre 2002 e 4001 unidades.

Há também que fazer com que o tempo continue, ou seja, que seja definido em intervalos consecutivos de tempo, no sentido em que se uma parametrização é feita, por exemplo, com tempo t de 0 a 10, a segunda tem que começar em $t = 10$. Assim, por exemplo, se tivermos duas parametrizações de retas:

$$f(t) = (0, t), t \in [0, 2]$$

$$g(t) = (t, 5t + 1), t \in [5, 8]$$

Para que o tempo possa ser contínuo, a segunda parametrização terá que passar nos mesmos pontos, mas com o tempo a começar em 2. Ora, como o intervalo $[5, 8]$ tem amplitude 3, então o novo intervalo terá também amplitude 3, ou seja, será $[2, 5]$.

Quando temos que $t \in [5, 8]$, para passarmos a estar no intervalo $[2, 5]$ teremos que ter $t + 3$, ou seja, na parametrização $g(t) = (t, 5t + 1), t \in [5, 8]$ substituiremos t , por $t + 3$, ficando com

$$\overline{g}(t) = (t + 3, 5(t + 3) + 1), t \in [2, 5].$$

Assim, as novas parametrizações, em tempo contínuo serão:

$$f(t) = (0, t), t \in [0, 2]$$

$$\overline{g}(t) = (t + 3, 5(t + 3) + 1), t \in [2, 5].$$

No projeto, a adequação da velocidade tem que ser feita em conjunto com o tempo contínuo.

Abaixo, damos conta de um diálogo que decorreu entre a professora e um dos estudantes, o qual ilustra uma das dificuldades que são usuais nos estudantes.

Aluno A: Professora, já fiz o carro a percorrer o circuito. Agora queria adequar as velocidades para mais rápido nas retas e mais lento nas curvas. Pode ajudar-me?

Professora: Sim, posso. Vamos refletir. Vamos começar pela tua primeira parametrização:

$$F(t) = (t, 5t - 3), t \text{ in } [0, 6]$$

Como fazemos para que o carro vá ao dobro da velocidade?

Aluno A: hummm... não sei...

Professora: Quanto tempo é que o carro vai necessitar?

Aluno A: 3 unidades.

Professora: Então o intervalo vai ser qual?

Aluno A: [0, 3]

Professora: Exato! Então como vamos fazer para o carro passar nos mesmos pontos com t entre 0 e 3?

Aluno A: ora como $F(t) = (t, 5t - 3)$, t pertence a $[0, 6]$ teria que ser $F(t) = (2t, 5 \cdot (2t) - 3)$, t in $[0, 3]$.

Professora: Isso! E agora, a parametrização seguinte tem que ser alterada?
Aluno A: sim. Ui... vou ter que alterar todas as parametrizações, não é?
Professora: Sim, de facto, tem que ser.
(Diálogo entre professor e aluno, 2022).

Como se pode constatar no diálogo acima, a dificuldade do estudante foi ultrapassada com a ajuda da professora.

Segue-se um exemplo de um circuito utilizado por um aluno (Figura 4) que utiliza a regra de incluírem pelo menos uma parte de uma reta ou parábola e uma parte de uma circunferência ou elipse.

Figura 4 – Exemplo de um circuito escolhido pelos alunos.



Fonte: Trabalho de um aluno.

A Figura 5 mostra um exemplo de Código que foi desenvolvido por um dos estudantes, para dar resposta ao problema colocado.

Figura 5: Exemplo de código Python.

```
def velocidade (t):  
    if t==0:  
        vel = 0  
    if 0<t<=2.5:  
        vel = 81.6  
    if 2.5<t<=5:  
        distancia = (posicaoy(t)-posicaoy(t-0.1))/(posicaox(t)-posicaox(t-0.1))  
        if distancia < 0:  
            distancia = -distancia  
        vel = distancia/0.1  
    if 5<t<=7.5:  
        vel= 20  
    if 7.5<t<=10:  
        distancia = (posicaoy(t)-posicaoy(t-0.1))/(posicaox(t)-posicaox(t-0.1))  
        if distancia < 0:  
            distancia = -distancia  
        vel = distancia/0.1
```

Fonte: Trabalhos entregues pelos estudantes.

Vamos analisar com mais detalhe dois projetos, entregues pelos estudantes, que cumpriram ou cumpriram em parte o objetivo pretendido. A Figura 6 é um exemplo em que o estudante escolheu uma pista complexa, dado que continha várias curvas.

Figura 6 – Pista escolhida pelo aluno para fazer o seu carro percorrer o circuito, com um circuito complexo.



Fonte: Trabalhos entregues pelos estudantes.

Na imagem dos circuitos, o eixo dos xx é representado na horizontal e a crescer da esquerda para a direita; o dos yy encontra-se na vertical mas cresce de cima para baixo; estando a origem do sistema no canto superior esquerdo. O estudante tem que ser capaz de entender o sistema de eixos, assim como deve conseguir fazer o carro andar mais depressa nas retas e mais devagar nas curvas.

Na Figura 7, o estudante fez as parametrizações para que o seu carro saísse da partida (ponto (354,120)), depois seguisse em reta para a direita (354+294t,120), em seguida percorreu duas meias elipses em sentido contrário uma da outra.

Figura 7 – Parametrização realizada pelo aluno para que o carro percorra o seu circuito.

```
def parametrizacao (t):  
    if t==0:  
        resultado=(354,120)  
    if 0<t<=1:  
        resultado=(354+294*t,120)  
    if 1<t<=4.14:    #(1+pi)  
        resultado=(648+75*cos(t+3.71),188+68*sin(t+3.71))  
    if 4.14<t<=7.77:  
        resultado=(601+99*cos(t-3.06),322-75*sin(t-3.06))
```

Fonte: Trabalhos entregues pelos estudantes

Também se avalia se a velocidade é calculada em cada instante e se aparece no ecrã, como é o caso da Figura 6.

A Figura 8 mostra um código que permite calcular a velocidade em cada momento, enquanto o carro circula na pista anterior (Figura 6), onde a velocidade é adaptada de acordo com o percurso em causa.

Figura 8: Exemplo de código Python para calcular a velocidade em cada trecho.

```
def velocidade (t):  
    if t==0:  
        vel = 0  
    if 0<t<=2.5:  
        vel = 81.6  
    if 2.5<t<=5:  
        distancia = (posicaooy(t)-posicaooy(t-0.1))/(posicaoax(t)-posicaoax(t-0.1))  
        if distancia < 0:  
            distancia = -distancia  
        vel = distancia/0.1  
    if 5<t<=7.5:  
        vel= 20  
    if 7.5<t<=10:  
        distancia = (posicaooy(t)-posicaooy(t-0.1))/(posicaoax(t)-posicaoax(t-0.1))  
        if distancia < 0:  
            distancia = -distancia  
        vel = distancia/0.1
```

Fonte: Trabalhos entregues pelos estudantes.

O estudante em causa teve que fazer parametrizações adequadas à pista escolhida. Podemos ver as primeiras parametrizações que ele fez na Figura 7. No instante inicial ($t = 0$), o carro está parado no ponto de coordenadas (354,120), depois o carro avança em linha reta, mantendo o valor de y em 120 e fazendo o valor de x variar entre os valores pretendidos, mantendo uma velocidade de 294 unidades. Em seguida, o carro percorre meia elipse (daí o valor de π nos comentários), são usadas coordenadas polares e a velocidade deve ser menor por se tratar de uma curva. De facto, podemos verificar que a derivada da função é $(-75\sin(t + 3.71), 68\cos(t + 3.71))$.

Portanto, a velocidade é:

$$\begin{aligned} & \sqrt{75^2\sin^2(t + 3.71) + 68^2\cos^2(t + 3.71)} \\ &= \sqrt{5625\sin^2(t + 3.71) + 4624\cos^2(t + 3.71)} \\ &= \sqrt{1001\sin^2(t + 3.71) + 4624\sin^2(t + 3.71) + 4624\cos^2(t + 3.71)} \\ &= \sqrt{1001\sin^2(t + 3.71) + 4624} \end{aligned}$$

Ou seja, está, quando muito, entre 68 e 75 unidades, o que faz com que a curva seja percorrida muito mais devagar do que a reta, tal como pretendido. A curva seguinte é um pouco

mais de meia elipse ($7.77 - 4.14 = 3.63$) e a sua velocidade está entre 75 e 99 (calcula-se de forma análoga), o que podemos dizer que é semelhante à velocidade na curva anterior. Este aluno calcula a velocidade para cada instante (usando, para isso, derivadas), e apresenta-a no canto inferior esquerdo (ver Figura 6).

Neste exemplo, a depuração é evidenciada, dado que o estudante teve que adaptar o código ao novo circuito, de forma a cumprir com as instruções previamente definidas para a realização do projeto.

Outro exemplo de um circuito, também bastante complexo, por ter muitas linhas incluindo várias curvas, é apresentado na Figura 9.

Figura 9 – Circuito escolhido pelo aluno para o carro percorrer a trajetória.



Fonte: Trabalhos entregues pelos estudantes.

Neste exemplo, pode verificar-se, na Figura 10, que a parametrização está correta. Inicialmente, o carro está parado no ponto (244, 296), depois percorre um segmento de reta vertical (com x constante, com o valor de 244) para cima (o valor de y diminui à medida que o tempo passa) a uma velocidade de 81.6 unidades, segue-se um arco de elipse com velocidade entre 10 e 17 (menor do que a da reta, como se pretende), segue-se um segmento de reta horizontal mas com velocidade de 20 (devia ter uma velocidade maior – esta linha não obteve

a cotação total), por fim um arco de elipse à velocidade entre 25 e 30 unidades. Este aluno sabe parametrizar, mas teve dificuldade em adequar as velocidades.

Figura 10 - Parametrizações realizadas pelo aluno.

```
def parametrizacao (t):  
    if t==0:  
        resultado=(244,296)  
    if 0<t<=2.5:  
        resultado=(244, 296-81.6*t)  
    if 2.5<t<=5:  
        resultado=(234-10*cos(t), 92+17*sin(t))  
    if 5<t<=7.5:  
        resultado=(234-(20*(t-5)), 75)  
    if 7.5<t<=10:  
        resultado=(184+30*cos(t+0.3),92-25*sin(t-0.5))
```

Fonte: Trabalhos entregues pelos estudantes.

No entanto, calcula bem as velocidades e apresenta-as no canto superior esquerdo do ecrã (Figura 9). No caso das retas, é imediato calcular a velocidade, portanto ele simplesmente indica o seu valor. No entanto, para as curvas, o método que ele utiliza para as calcular é muito interessante e pouco vulgar, faz uma aproximação numérica da derivada, como se pode ver na Figura 11.

Figura 11 – Uma forma pouco usual do aluno calcular a velocidade em cada instante.

```
def velocidade (t):  
    if t==0:  
        vel = 0  
    if 0<t<=2.5:  
        vel = 81.6  
    if 2.5<t<=5:  
        distancia = (posicaooy(t)-posicaooy(t-0.1))/(posicaoax(t)-posicaoax(t-0.1))  
        if distancia < 0:  
            distancia = -distancia  
        vel = distancia/0.1  
    if 5<t<=7.5:  
        vel= 20  
    if 7.5<t<=10:  
        distancia = (posicaooy(t)-posicaooy(t-0.1))/(posicaoax(t)-posicaoax(t-0.1))  
        if distancia < 0:  
            distancia = -distancia  
        vel = distancia/0.1
```

Fonte: Trabalhos entregues pelos estudantes.

Há ainda outros alunos que utilizam bibliotecas do Python para calcular as derivadas (Figura 12).

Figura 12: O cálculo da velocidade em cada instante.

```
position = parametrizacao(t)
velocidade_inst = derivative(t, tipo="s")
velocidade = math.sqrt(((velocidade_inst[0])**2)+((velocidade_inst[1])**2))
```

Fonte: Trabalhos entregues pelos estudantes.

Como se pode constatar nos exemplos acima apresentados, os estudantes desenvolveram o pensamento computacional no decorrer das diversas etapas do projeto. A abstração surgiu na interpretação do código e na sua adaptação para resolver o problema. Também conseguiram sintetizar e resolver o problema decompondo-o em passos mais simples (decomposição). Conseguiram, por exemplo, relacionar o circuito com as parametrizações a usar, assim como adequar as velocidades, surgindo assim a identificação de padrões. Foram ainda capazes de criar diversos passos de forma a cumprir sequências de instruções, de forma a resolver o problema (algoritmia).

Apesar de nem sempre a depuração ser visível nos trabalhos acima apresentados (dado que refletem o produto final do trabalho realizado pelos estudantes), no decorrer do apoio à resolução do projeto, assim como na discussão final, a professora observou, como seria de esperar numa atividade de programação, que os estudantes identificaram e corrigiram erros até solucionarem o problema proposto.

Finalmente, na Tabela 2, apresentamos os conteúdos relacionados com STEM que estão incluídos no projeto.

Tabela 2 – Conteúdos de STEM incluídos no projeto CarRace.

Ciência	Tecnologia	Engenharia	Matemática
Física	Computador	Design	Funções de IR para IR ² , parametrizações em IR ² , ajustamento dos parâmetros
Velocidade	Python	Programação	Funções lineares, quadráticas e trigonométricas
Caminho	Calculadora		Retas, 16arabolas, circunferências, elipses
	Vídeo		Coordenadas polares, derivadas, norma de um vetor

Fonte: elaborada pelos autores.

No final da experiência, os estudantes foram convidados a escrever alguns comentários sobre o projeto realizado. Entre os vários comentários, destacamos alguns dos mais positivos, tais como “Adorei realmente ver matemática ser utilizada para alguma coisa útil”, “Fez pela primeira vez aplicar conteúdos matemáticos na criação de algo novo”, “Deu-me uma motivação extra”, “Ajudou imenso a compreender a matéria de parametrizações”, “É uma excelente forma

de usar matemática com programação”, “Cadeira fez-me ver a matemática de maneira diferente”, “Devia-se fazer mais vezes”.

Foram muitos os comentários semelhantes aos que se destacam acima, pelo que a experiência foi bastante positiva. Na verdade, muitos estudantes referiram que pela primeira vez compreenderam a necessidade de se usar matemática, nomeadamente na programação. Também disseram que esta abordagem os ajudou a compreender a matemática, em particular a matéria das parametrizações.

Tivemos também comentários menos positivos: “Não tinha os conhecimentos necessários para corrigir (...) erros”, “Dificuldade na programação”, “Não dominava a linguagem Python”. Como se pode verificar, estes comentários menos positivos têm a ver com conhecimentos de programação. De facto, para implementar com eficácia uma abordagem interdisciplinar, é necessário um domínio dos conceitos e procedimentos das várias áreas curriculares. Desta forma, esta que é uma das principais dificuldades apontadas para uma implementação com eficácia desta abordagem por parte dos professores (COSTA *et al.*, 2023), também parece ser uma das dificuldades do ponto de vista dos estudantes, quando estes não dominam os conceitos de outras unidades curriculares.

A resolução do projeto envolveu o raciocínio com abordagens analíticas e algorítmicas, que estão relacionadas com a Pensamento Computacional (BOCCONI *et al.*, 2016; WEINTROP *et al.*, 2016; WING, 2006). Os alunos tiveram de interpretar o código em Python e adaptá-lo ao circuito que escolheram, respeitando as instruções de incluir algum tipo de curvas que relacionaram com a matemática. Além disso, tiveram de utilizar parametrizações para ajustar a velocidade do carro ao circuito, tal como ilustrado acima.

Em resumo, a maioria dos estudantes considerou esta abordagem muito positiva, tendo destacado o papel da matemática e que esta passou a ser vista como “útil” e “de maneira diferente”. Em particular, verificou-se que alguns alunos não dominavam bem a programação, nomeadamente a linguagem de programação em Python, o que pode trazer dificuldades acrescidas à tarefa.

5 CONCLUSÕES

São cada vez mais as referências sobre a necessidade de usar abordagens de ensino mais eficazes para motivar os estudantes para a aprendizagem, assim como para os preparar para a vida profissional, nomeadamente através do recurso a problemas reais (DE VERE, 2009;

MALICKY *et al.*, 2006; SAVERY, 2006). Por exemplo, a educação STEM é cada vez mais sugerida, havendo uma tendência mais recente para uma abordagem mais integrada (COSTA; DOMINGOS, 2020; 2023), sendo em particular referido que se deve dar protagonismo à matemática (BESWICK; FRASER, 2019; STOHLMANN, 2018).

A tecnologia está incluída nas STEM e também é sugerido que esta seja integrada com a matemática, podendo a programação ser uma forma de motivar os estudantes para a aprendizagem da matemática (ABRAMOVICH, 2013; JOHNSON, 2000).

Com base na análise de dados realizada na secção anterior, verificou-se que o projeto CarRace promoveu o Pensamento Computacional em estudantes de engenharia no contexto da educação STEM (Figuras 4 a 12, Tabela 2) porque exigia competências de resolução de problemas e também ajudava a aprender matemática tal como sugerido por alguns autores (LI *et al.* 2020; LU *et al.*, 2022). De facto, os alunos utilizaram a matemática num contexto de programação e STEM e desenvolveram Pensamento Computacional como a abstracção, a decomposição, a identificação de padrões, a algoritmia e a depuração para realizar as tarefas solicitadas (WING, 2006).

Em particular, foram utilizados vários conteúdos matemáticos, tais como: Funções de \mathbb{R} em \mathbb{R}^2 (parametrizações), incluindo funções lineares, quadráticas e trigonométricas; coordenadas polares; curvas como retas, parábolas, círculos e elipses; derivadas e norma de um vetor, ajuste de parâmetros, entre outros (Tabela 2), o que os fez compreender a importância da matemática na programação e para resolver problemas do mundo real.

Para além de se verificar que o projeto implementado promoveu o Pensamento Computacional nos estudantes envolvidos, os resultados obtidos também mostram a importância de implementar este tipo de abordagem com estudantes do ensino superior, nomeadamente estudantes de engenharia. No entanto, alguns mostraram dificuldades relacionadas com a programação, o que pode comprometer a eficácia de implementação de uma abordagem interdisciplinar. Apesar disso, como se verificou na secção anterior, a grande maioria dos estudantes reconheceram que esta abordagem os ajudou a compreender melhor a matemática, assim como a sua importância para resolver problemas da vida real.

Tal como vários autores, concluímos que esta pode ser uma abordagem promotora de um melhor desempenho académico, e motivadora da aprendizagem da matemática, assim como pode preparar melhor os estudantes para futuras profissões (HUMBLE, 2021; DE VERE, 2009; MALICKY *et al.*, 2006; SAVERY, 2006).

REFERÊNCIAS

- ABRAMOVICH, Sejei. Computers in Mathematics Education: An Introduction. **Computers in the Schools**, v. 30, n. 1-2, p. 4–11, jan. 2013. <https://doi.org/10.1080/07380569.2013.765305>
- AGUILERA, D; Lupiàñez José Luis; VILCHEZ-GONZÁLEZ, José Miguel; PARALES-PALACIOS, Francisco Javier. In Search of a Long-Awaited Consensus on Disciplinary Integration in STEM Education. **Mathematics**, v. 9, n. 6, p. 597, jan. 2021. <https://doi.org/10.3390/math9060597>
- ALBUQUERQUE, Carlos. Pensamento Computacional e Matemática. **Educação e Matemática**, n. 162, p. 31–38, 2021. <https://em.apm.pt/index.php/em/article/view/2742>
Acesso em: 16 nov.2023.
- BEAZLEY, David. **Python essential reference**. Sams Publishing, 2006. <https://www.informit.com/store/python-essential-reference-9780672328626>. Acesso em: 16 nov.2023.
- BECKER, Kurt. Henry.; PARK, Kyungsuk. Integrative Approaches among Science, Technology, Engineering, and Mathematics (STEM) Subjects on Students’ Learning: A Meta-Analysis. **Journal of STEM Education: Innovations and Research**, v. 12, n. 5, 6 jun. 2011. <https://www.jstem.org/jstem/index.php/JSTEM/article/view/1509>. Acesso em: 16 nov.2023.
- BESWICK, Kim.; FRASER, Sharon. Developing mathematics teachers’ 21st century competence for teaching in STEM contexts. **ZDM**, set. 2019. <https://doi.org/10.1007/s11858-019-01084-2>.
- BOCCONI, Stefania; CHIOCCARIELLO, Augusto; DETTORI, Giuliana; ENGELHARDT, Katja. **Developing Computational Thinking in Compulsory Education - Implications for policy and practice**. Luxembourg: Publications Office of the European Union, 2016. Disponível em: <https://ideas.repec.org/p/ipt/iptwpa/jrc104188.html>. Acesso em: 16 nov.2023.
- COHEN, Louis Cohen; MANION, Lawrence Manion; MORRISON, Keith. **Research Methods in Education**. [s.l.] Routledge, 2002. <https://doi.org/10.4324/9780203224342>.
- COSTA, Maria Cristina; DOMINGOS, António. Teachers’ Professional Knowledge to Develop STEM Integrated Tasks. **Pedagogika**, v. 149, n. 1, p. 82–104, maio, 2023. <https://doi.org/10.15823/p.2023.149.4>
- COSTA, Maria Cristina; DOMINGOS, António.; TEODORO, Vítor. Promoting Integrated STEM Tasks in the Framework of Teachers’ Professional Development in Portugal. **Advances in STEM education**, p. 511–532, jan. 2020. https://doi.org/10.1007/978-3-030-52229-2_27
- COSTA, Maria Cristina; DOMINGOS, António Manuel Dias; TEODORO, Vítor Duarte; VINHAS, Élia Maria Rodrigues Guedes. Teacher Professional Development in STEM Education: An Integrated Approach with Real-World Scenarios in Portugal. **Mathematics**, v. 10, n. 21, p. 3944, 24 out. 2022. <https://doi.org/10.3390/math10213944>

DE VERE, Ian. **Developing creative engineers: a design approach to engineering education**. Disponível em:

<https://www.designsociety.org/publication/28948/Developing+creative+engineers%3A+a+design+approach+to+engineering+education> . Acesso em: 28 set. 2023.

FITZALLEN, Noleine. **STEM Education: What Does Mathematics Have to Offer?** [s.l.] Mathematics Education Research Group of Australasia, 2015.

<https://eric.ed.gov/?id=ED572451> . Acesso em: 16 nov.2023.

HEINTZ, Frederik; MANNILA, Linda; NORDÉN Lars-Ake; REGNELL, Björn. Introducing Programming and Digital Competence in Swedish K-9 Education. **Lecture Notes in Computer Science**, p. 117–128, 1 jan. 2017. https://doi.org/10.1007/978-3-319-71483-7_10

HUMBLE, Niklas. The use of Programming Tools in Teaching and Learning Material by K-12 Teachers, 2021. <https://www.diva-portal.org/smash/get/diva2:1606871/FULLTEXT01.pdf>

JOHNSON, David C. **Education and Information Technologies**, v. 5, n. 3, p. 201–214, 2000. <https://doi.org/10.1023/A:1009658802970>

LI, Yeping; SCHOENFLD, Alan H. DUSCHL, Richard A. Computational Thinking Is More about Thinking than Computing. **Journal for STEM Education Research**, v. 3, n. 1, p. 1–18, abr. 2020. <https://doi.org/10.1007/s41979-020-00030-2>

LU, C. MACDONALD, R; CUTUMISU, M. A scoping review of computational thinking assessments in higher education. **Journal of Computing in Higher Education**, v.34, p. 416-461, jan. 2022. <https://doi.org/10.1007/s12528-021-09305-y>

MALICKY, D. M.; LORD, S. M.; HUANG, M.; Problem, Project, Inquiry, Or Subject Based Pedagogies: What To Do? **Proceedings of ASEE Annual Conference**, Chicago, IL, set. 2006. <https://doi.org/10.18260/1-2--1019>

RESNICK, M.; MALONY, J.; MOROY-HERNÁNDEZ, A.; RUSK, N.; EASTMOND, EV.; MILLNER, A.; ROSENBAUM, E. Scratch: programming for all. **Communications of the ACM**, v. 52, n. 11, p. 60-67, nov. 2009. <http://doi.acm.org/10.1145/1592761.1592779>

SAVERY, John. R. Overview of problem-based learning: Definitions and distinctions. **Interdisciplinary Journal of Problem-Based Learning**, v.1, n.1, p. 9-20, maio. 2006. <https://doi.org/10.7771/1541-5015.1002>

STOHLMANN, M. A vision for future work to focus on the “M” in integrated STEM. **School Science and Mathematics**, v. 118, n. 7, p. 310–319, out. 2018. <https://doi.org/10.1111/ssm.12301>

WING, J. M. Computational thinking. **Communications of the ACM**, v. 49, n. 3, p. 33, mar. 2006. <https://doi.org/10.1145/1118178.1118215>

WEINTROP, D. et al. Defining Computational Thinking for Mathematics and Science Classrooms. **Journal of Science Education and Technology**, v. 25, n. 1, p. 127–147, out. 2015. <https://doi.org/10.1007/s10956-015-9581-5>

ZIATDINOV, R.; VALLES, J. R. Synthesis of Modeling, Visualization, and Programming in GeoGebra as an Effective Approach for Teaching and Learning STEM Topics. **Mathematics**, v. 10, n. 3, p. 398, jan. 2022. <https://doi.org/10.3390/math10030398>

APÊNDICE 1 – INFORMAÇÕES SOBRE O MANUSCRITO

AGRADECIMENTOS

Não se aplica.

FINANCIAMENTO

Não se aplica.

CONTRIBUIÇÕES DE AUTORIA

Resumo/Abstract/Resumen: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Introdução: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Referencial teórico: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Análise de dados: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Discussão dos resultados: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Conclusão e considerações finais: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Referências: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Revisão do manuscrito: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

Aprovação da versão final publicada: Maria Cristina Costa, António Domingos, Sandra Gaspar Martins

CONFLITOS DE INTERESSE

Os autores declararam não haver nenhum conflito de interesse de ordem pessoal, comercial, acadêmico, político e financeiro referente a este manuscrito.

DISPONIBILIDADE DE DADOS DE PESQUISA

O conjunto de dados que dá suporte aos resultados da pesquisa foi publicado no próprio artigo.

PREPRINT

Não publicado.

CONSENTIMENTO DE USO DE IMAGEM

Não se aplica.

APROVAÇÃO DE COMITÊ DE ÉTICA EM PESQUISA

Não se aplica.

COMO CITAR - ABNT

COSTA, Maria Cristina; MARTINS, Sandra Gaspar; DOMINGOS, António. As STEM e o pensamento computacional: resolvendo desafios da vida real no ensino superior. **REAMEC – Rede Amazônica de Educação em Ciências e Matemática**. Cuiabá, v. 11, n. 1, e23100, jan./dez., 2023. <https://doi.org/10.26571/reamec.v11i1.16743>

COMO CITAR - APA

Costa, M. C., Martins, S. G., Domingos, A. (2023). As STEM e o pensamento computacional: resolvendo desafios da vida real no ensino superior. *REAMEC - Rede Amazônica de Educação em Ciências e Matemática*, 11(1), e23100. <https://doi.org/10.26571/reamec.v11i1.16743>

LICENÇA DE USO

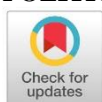
Licenciado sob a Licença Creative Commons [Attribution-NonCommercial 4.0 International \(CC BY-NC 4.0\)](https://creativecommons.org/licenses/by-nc/4.0/). Esta licença permite compartilhar, copiar, redistribuir o manuscrito em qualquer meio ou formato. Além disso, permite adaptar, remixar, transformar e construir sobre o material, desde que seja atribuído o devido crédito de autoria e publicação inicial neste periódico.



DIREITOS AUTORAIS

Os direitos autorais são mantidos pelos autores, os quais concedem à Revista REAMEC – Rede Amazônica de Educação em Ciências e Matemática - os direitos exclusivos de primeira publicação. Os autores não serão remunerados pela publicação de trabalhos neste periódico. Os autores têm autorização para assumir contratos adicionais separadamente, para distribuição não exclusiva da versão do trabalho publicado neste periódico (ex.: publicar em repositório institucional, em site pessoal, publicar uma tradução, ou como capítulo de livro), com reconhecimento de autoria e publicação inicial neste periódico. Os editores da Revista têm o direito de realizar ajustes textuais e de adequação às normas da publicação.

POLÍTICA DE RETRATAÇÃO - CROSSMARK/CROSSREF



Os autores e os editores assumem a responsabilidade e o compromisso com os termos da Política de Retratação da Revista REAMEC. Esta política é registrada na Crossref com o DOI: <https://doi.org/10.26571/reamec.retratacao>

PUBLISHER

Universidade Federal de Mato Grosso. Programa de Pós-graduação em Educação em Ciências e Matemática (PPGECM) da Rede Amazônica de Educação em Ciências e Matemática (REAMEC). Publicação no [Portal de Periódicos UFMT](#). As ideias expressadas neste artigo são de responsabilidade de seus autores, não representando, necessariamente, a opinião dos editores ou da referida universidade.

EDITOR



Dailson Evangelista Costa  

EDITORES CONVIDADOS

Andréia Dalcin  

Rafael Montoito  

AVALIADORES

Andréia Dalcin  

Elmha Coelho Martins Moura  

HISTÓRICO

Submetido: 10 de setembro de 2023.

Aprovado: 23 de novembro de 2023.

Publicado: 9 de dezembro de 2023.
