PROFESSOR DE MATEMÁTICA ONLINE
Revista eletrônica da Sociedade Brasileira de Matemática

PMO v.12, n.3, 2024
ISSN: 2319-023X
https://doi.org/10.21711/2319023x2024/pmo1223

Lógica computacional no ensino da Matemática: a matemagia da programação

Lee Jing Xuan Rubia Mara de Oliveira Santos

Resumo

O processo de ensino e aprendizagem da Matemática nas escolas públicas enfrenta desafios crescentes, com um notável aumento de alunos apresentando conhecimentos básicos pouco consolidados e raciocínio lógico subdesenvolvido, resultando em baixo rendimento e desânimo dos estudantes. Em resposta a esse cenário, este artigo apresenta um relatório de aplicação de uma metodologia ativa de ensino com temática inspirada em bruxos e magias de Harry Potter, para introduzir conceitos de lógica computacional e a linguagem de programação Python. Por meio de um conjunto de atividades, foram desenvolvidos pensamentos matemáticos, lógicos e computacionais que são associados aos diversos conteúdos matemáticos da educação básica. Por fim, o trabalho contribuiu para um maior engajamento dos alunos e interações produtivas nas aulas, destacando a relevância da lógica computacional no desenvolvimento de competências e habilidades matemáticas.

Palavras-chave: Ensino-aprendizagem da Matemática; Metodologia Ativa; Lógica Computacional.

Abstract

The process of teaching and learning Mathematics in public schools faces growing challenges, with a notable increase in students exhibiting poorly consolidated basic knowledge and underdeveloped logical reasoning, resulting in low performance and student discouragement. In response to this scenario, this article presents a report on the application of an active teaching methodology inspired by the wizards and magic of Harry Potter to introduce concepts of computational logic and the Python programming language. Through a series of activities, mathematical, logical, and computational thinking was developed and associated with various mathematical contents of basic education. Ultimately, the work contributed to greater student engagement and productive interactions in the classroom, highlighting the relevance of computational logic in the development of mathematical competencies and skills.

Keywords: Teaching-learning Mathematics; Active Methodology; Computational Logic.

1. Introdução

Em 2021, de acordo com os dados apresentados pelo Índice de Desenvolvimento da Educação Básica (Ideb) [4], observa-se um significativo retrocesso na aprendizagem devido à pandemia. Em Matemática, no 2º ano do Ensino Fundamental, 80% dos estudantes não conseguem realizar operações simples de soma e subtração, resultando em uma queda de 9 pontos na proficiência média. No 5º ano, a porcentagem de estudantes incapazes de identificar figuras geométricas atinge 38,9%. Nos anos finais do Ensino Fundamental (9º ano) e no Ensino Médio, a proficiência média caiu 7 pontos. Além disso, os resultados do Sistema de Avaliação da Educação Básica (Saeb) [8] indicam uma regressão profunda no aprendizado da Matemática. Como resultado, os alunos que foram automaticamente aprovados devido ao período pandêmico não consolidaram os conhecimentos básicos, enfrentando agora grandes obstáculos para avançar nos conteúdos programáticos de suas séries.

A Escola Estadual Professora Clarinda Mendes de Aquino (CMA), localizada na avenida Murilo Rolim Júnior, 200 - Jardim Petrópolis, Campo Grande - MS, é uma instituição de educação integral e autoral que fomenta o protagonismo juvenil e valoriza constantemente práticas pedagógicas inovadoras e diversificadas. Diante de um público de alunos com conhecimento básico limitado e diversas habilidades e competências subdesenvolvidas, metodologias alternativas visando à recuperação das aprendizagens, sem, no entanto, retardar o progresso educacional, estavam pendentes. Dessa forma, em uma disciplina da parte diversificada de Matemática para os 9^{os} anos, foi proposta uma metodologia ativa e lúdica, com elementos de bruxos e magias baseados em Harry Potter, para ensinar a linguagem de programação Python. O objetivo é promover um maior protagonismo nas aprendizagens matemáticas, melhorar o desempenho nas avaliações e aprimorar o raciocínio matemático por meio da lógica computacional.

A proposta foi implementada a partir do segundo semestre de 2023, constituindo uma continuação de uma eletiva do primeiro semestre, que explorava diversos conceitos matemáticos como se fossem feitiços e magias, inspirados nos filmes de Harry Potter. A escolha frequente desses filmes como elementos atrativos foi motivada pela grande paixão da maioria dos estudantes pela série, além de uma boa associação dos conteúdos a serem abordados. O destaque da nova disciplina reside no seu enfoque mais centralizado na lógica computacional e na linguagem de programação Python, alinhado com as habilidades previstas na Base Nacional Comum Curricular (BNCC) [9], e na integração da cultura digital como uma transversalidade do tema.

O Python, criado em 1989 pelo programador holandês Guido Van Rossum, foi escolhido para a proposta por ser atualmente considerada a linguagem de programação mais acessível. Isso se deve às suas sintaxes mais simples e à capacidade de aplicar a lógica de programação diretamente no código, dispensando a necessidade de o programador dominar conceitos mais avançados. Além disso, com o contínuo avanço científico e tecnológico, diversas áreas de atuação começaram a exigir um mínimo de conhecimento em lógica computacional. Portanto, a linguagem de programação Python não apenas beneficiará o desenvolvimento intelectual dos alunos, mas também abrirá portas para futuras oportunidades profissionais.

Diante do exposto, foram delineadas as etapas essenciais que fundamentaram o êxito do projeto: a elaboração da ementa e da guia de aprendizagem para orientação; a seleção dos alunos interessados na proposta durante a Feira das Eletivas; conjunto de atividades para ensinar os comandos básicos de Python; desenvolvimento de algoritmos; a criação do "Diário Digital dos Bruxos"como tarefa final; e, por fim, a culminância, que consiste na apresentação, à comunidade escolar, de todas atividades realizadas e resultados alcançados.

2. Fundamentação teórica

Dada a atual complexidade no ensino da Matemática, é crucial adotar abordagens pedagógicas que promovam a participação ativa dos alunos. Nesse contexto, as ideias de Paulo Freire oferecem uma visão transformadora do processo educacional. Freire (1968) [7] critica a abordagem bancária de ensino, na qual os alunos são considerados passivos receptores de conhecimento. Em sua proposta de pedagogia libertadora, os estudantes tornam-se participantes ativos no processo de aprendizagem, dialogando, questionando e refletindo sobre os conteúdos, especialmente no campo desafiador da Matemática.

Em consonância com as concepções de Freire, a interconexão entre a temática de Harry Potter e os conceitos matemáticos destaca a relevância da contextualização no processo de aprendizagem, conforme defendido por Boaler (2016) [2]. Boaler enfatiza a necessidade de tornar a Matemática relevante e acessível, estimulando a curiosidade dos alunos e permitindo que eles experimentem a disciplina de maneira criativa e envolvente. No entanto, mesmo diante dessas ideias progressistas, ainda testemunhamos continuadamente a persistência da educação bancária no ensino da Matemática. Ou seja, os alunos são tratados como meros recipientes vazios a serem preenchidos com as fórmulas e teoremas transmitidos pelo professor, o que limita sua capacidade de pensar criticamente e de se engajar ativamente no processo de aprendizagem.

Ao explorar metodologias ativas, as ideias de Vygotsky (1978) [11] sobre a aprendizagem colaborativa e a zona de desenvolvimento proximal servem de inspiração. Trabalhando em grupo na construção de algoritmos, os alunos podem interagir, discutir ideias e construir conhecimentos e raciocínios lógicos coletivamente, ampliando suas capacidades cognitivas. É uma oportunidade de engajar os alunos de maneira mais eficaz, em oposição aos métodos tradicionais, promovendo a criatividade e o protagonismo dos alunos em seu próprio aprendizado.

Nos últimos anos, a linguagem de programação Python tem recebido crescente reconhecimento, contando sua participação no desenvolvimento de *software* moderno, gerenciamento de infraestrutura, análise de dados, criação de jogos e robótica. Com isso, debates sobre sua importância na educação básica e as práticas de integração no ensino da Matemática tornaram-se assuntos recorrentes na literatura. Por exemplo, a investigação da eficiência da linguagem de programação Python, como recurso didático para formular e resolver problemas matemáticos no contexto da educação básica [10]; a análise das possibilidades do Python na escola pública para Ensino Médio [5]; a linguagem Python como ferramenta no ensino básico [3]; entre outros.

No entanto, percebe-se uma carência de relatos de experiências direcionados ao Ensino Fundamental como público-alvo. E então, pretendemos destacar e investigar as possibilidades de aplicação dessas práticas com alunos que ainda não cursam o Ensino Médio. A ênfase incide sobre a importância da alfabetização e do letramento digital desde as fases iniciais da educação, alinhando-se à competência geral 5 da BNCC:

> *Compreender, utilizar e criar tecnologias digitais de informação e comunicação de forma crítica, significativa, reflexiva e ética nas diversas práticas sociais (incluindo as escolares) para se comunicar, acessar e disseminar informações, produzir conhecimentos, resolver problemas e exercer protagonismo e autoria na vida pessoal e coletiva.* (BNCC, 2018)

Portanto, a proposta de utilizar metodologias ativas centradas na linguagem de programação Python visa não apenas enriquecer o aprendizado dos alunos, mas também preparar os estudantes para desafios e oportunidades do mundo digital e matemático em constante evolução.

3. Metodologia

Os tópicos a seguir oferecem os detalhes de cada etapa desenvolvida. No entanto, devido à ênfase central na programação, as descrições que não possuíam esse foco, foram resumidos ou dispensados. Contudo, foi

criada uma página no Instagram (@cma_hogwartianos_matemagia) para compartilhar abertamente todas as atividades realizadas e iniciativas promovidas.

3.1. Ementa

No documento, constam a justificativa, os objetivos, as habilidades e competências a serem desenvolvidas, o conteúdo programático, a metodologia, os recursos didáticos, a proposta de culminância e as referências bibliográficas. A cada semestre, o material guia o percurso educacional, proporcionando uma compreensão geral do escopo e da direção das disciplinas.

3.2. Guia de aprendizagem

Na criação da guia de aprendizagem, a finalidade foi detalhar para os estudantes e a comunidade os fundamentos, as atividades e as avaliações que serão realizadas durante o período. No documento, constam a justificativa da unidade, atividades prévias, atividades didáticas, conteúdos, fontes e referências, atividades autodidáticas, atividades didático-cooperativas, atividades complementares, temas transversais, valores, objetivos específicos e critérios de avaliação.

3.3. Feira das Eletivas

No começo do semestre, a escola organizou a Feira das Eletivas para os alunos do 9º ano do Ensino Fundamental escolherem as eletivas de seu interesse. Antes de apresentar novas propostas, foram destacados os resultados positivos do semestre anterior e enfatizado o sucesso da tematização da sala-ambiente de Matemática pelos próprios alunos, conforme ilustrado na Figura 1.

Figura 1: Sala-Ambiente da Matemática tematizada para Matemagia

Com os resultados do semestre passado, foi demonstrada a grande produtividade da eletiva anterior e o crescimento significativo do protagonismo dos estudantes.

Em seguida, um algoritmo interativo foi usufruído para cativar os alunos, a fim de garantir que a linguagem de programação Python não perca seu destaque como tema principal. Inspirado no diário de Tom Marvolo Riddle do universo Harry Potter, o algoritmo trocava diálogos com os usuários e tinha um enigma a ser resolvido, o qual era descobrir o nome de um professor através da permutação das letras do nome que aparecia no código. No final, os estudantes manifestaram grande contetamento, curiosidade e interesse para participarem das aulas e atividades sugeridas.

O algoritmo interativo foi implementado no Spyder, um ambiente de desenvolvimento integrado (IDE) específico para a linguagem de programação Python. Os detalhes do algoritmo são apresentados no Código 1.

```
import time
intervalo = 3

while 1+1==2:
    x = input("Coloque o seu nome: \n")
    time.sleep(1)
    print("\nOi,",x,", prazer em te conhecer...")
    time.sleep(1)
    print("\nEu sou Felipe G. X. Junoran \n")
    time.sleep(1)

    y = input("Qual série e turma você é? \n")
    time.sleep(1)
    while y != "9A" and y!="9B" and y!="9C":
        y = input("\n9A, 9B ou 9C?\n")
        time.sleep(1)
    if y=="9A":
        print("\nHUMM...\nA turma considerada o terror dos profs?!")
        time.sleep(2)
        print("Coitado do Prof Lee, padrinho da sala...")
        time.sleep(3)
        print("Deve ser por isso que ele tem cabelo branco aos 22")
        time.sleep(3)
    if y=="9B":
        print("\nHUMM...\nA turma que estão sempre desanimados?!")
        time.sleep(2)
        print("Coitado do Prof Fernandinho, padrinho da sala...")
        time.sleep(3)
        print("Deve ser por isso que ele teve que até fazer canjica na festa julina para ajudar a sala a contribuir")
        time.sleep(3)
    if y=="9C":
        print("\nHUMM...\nA turma que não para de bagunçar?!")
        time.sleep(2)
        print("Ainda bem que o padrinho de vcs é o Prof Torres...")
        time.sleep(3)
        print("Pois, filosoficamente, vocês nascem bons, mas a sociedade que os corromperam")
        time.sleep(3)

    z = input ("\nEu tenho um segredo e uma fofoca, só posso te contar um, qual você quer saber? \n")
    time.sleep(1)
    while z !="segredo" and z!="fofoca":
        z = input("\nsegredo ou fofoca? \n")
        time.sleep(1)
    if z =="fofoca":
        print("\nVocê sabia? Eu tenho um amigo que viveu até 100 anos...")
```

```
        time.sleep(intervalo)
        print("Sabe como ele fez isso? Cuidando muito bem SÓ da vida dele! S2")
        time.sleep(3)
        print("\nEnfim, foi bom te conhecer..., desejo a você muito sucesso e aprendizagem num novo semestre no CMA\n\n")
        time.sleep(5)
    if z=="segredo":
        print("\nFelipe G. X. Junoran não é meu verdadeiro nome... \nNem a forma mais formal que as pessoas me chamam...")
        time.sleep(4)
        print("\nO jeito mais correto e formal de me chamar... \nDeve vir primeiramente um pronome de tratamento abreviado...")
        time.sleep(5)
        print("\nE depois, meu nome completo formado por 3 partes...")
        time.sleep(4)
        print("\nUm fato interessante, é que o jeito correto de me chamar... \nSão formados pelas mesmas letras do meu nome falso Felipe G. X. Junoran")
        time.sleep(4)
        print("\nSe você ACERTAR quem eu sou com um bom ARGUMENTO, você ganha um PRESENTE INESPERADO!!!")
        time.sleep(8)
        print("\nPor fim, fico contente em te conhecer, a eletiva da matemagia sempre estará de portas abertas para você!\n\n")
        time.sleep(10)
```

Código 1: Código do algoritmo interativo parte 1

3.4. Atividades tutoriais de Python

Neste tópico, serão apresentadas, sequencialmente, as atividades tutoriais (que podem ser entendidas como aulas) destinadas a estudo e compreensão dos comandos básicos do Python, bem como dos símbolos matemáticos e operadores booleanos que serão imprescindíveis na construção dos algoritmos subsequentes.

Atividade introdutória

Antes de iniciar as atividades previstas, foi feita a referência ao programa de PROFMAT (Mestrado Profissional em Matemática em Rede Nacional) da UFMS (Universidade Federal de Mato Grosso do Sul) https://ppgprofmat.ufms.br/ que proporcionou ao docente responsável a oportunidade e capacidade de planejamento, implementação e execução do projeto.

Para começar, foi apresentada uma breve história da origem de Python, seguindo-se definições e explicações do que se trata uma linguagem de programação. Como noções primordiais, foram esclarecidas também, as relações lógicas entre o programador, a máquina e os usuários, que são fundamentais para o entendimento dos comandos básicos nas atividades posteriores e na construção dos algoritmos. Basicamente, o programador deve exercer a função de dar instruções e comandos no painel de código para a máquina executar de forma lógica e precisa. Já os usuários, recebem a resposta do algoritmo pelo console (painel de saída), podendo também inserir dados e informações no mesmo espaço.

Como conclusão da primeira atividade, foi apresentado o IDE Spyder do Python na versão 3.10 e seus métodos para edição e teste de códigos, o qual poderia ser utilizado pelos alunos nas atividades seguintes. Entretanto, devido à falta de computadores disponíveis, foi solicitado aos alunos que baixassem o aplicativo "Coding Python", com funções semelhantes a um IDE, disponível na Google Play Store.

Atividade sobre o comando `print`

Essa atividade tem como objetivo explicar o primeiro comando básico `print`, que imprime (mostra) informações no console.

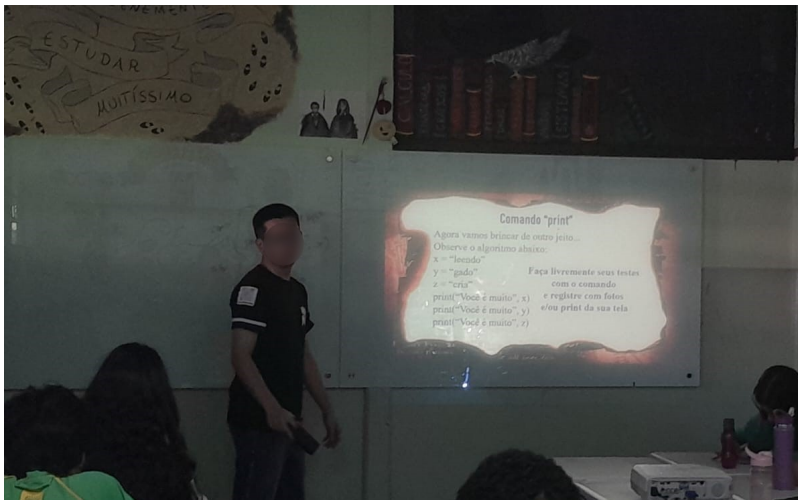

Figura 2: Ensinando Comando `print`

Para iniciar a atividade, os estudantes começaram com experimentos simples, como analisar os resultados de `print("oi")`, `print("Tudo bem?")` e as possibilidades com outros textos entre as aspas. Assim, estabelece uma relação entre a mensagem inserida com a resposta que é apresentada no painel de saída.

Após os pequenos testes anteriores, foi solicitado aos alunos que investigassem o que aconteceria ao dispensar ou acrescentar em excesso aspas, parênteses ou outros símbolos. A conclusão foi que, na programação, seja uma vírgula, espaço, letra maiúscula ou minúscula são detalhes relevantes que podem resultar em `bugs` (erros no código) do algoritmo ou em interpretações distintas pela máquina. Esses detalhes guardam semelhanças com as operações numéricas nas aulas de Matemática, onde um sinal de positivo ou negativo atribuído incorretamente pode levar a um resultado inesperado.

Na sequência, foi desenvolvida uma lógica computacional associada a um pensamento matemático recorrente na disciplina, que consiste na substituição de variáveis por seus valores já definidos. Por exemplo, ao determinar `x = "aluno"`, e executar o código `print("Você é um", x)`, resultará a resposta no console `Você é um aluno`, pois a máquina compreende o `x` como a palavra `aluno`. A partir de um exemplo como o anterior, seguimos com testes para casos com mais variáveis, seja no mesmo ou em vários comandos `print`.

Essa simples lógica computacional é aplicável a diversos conteúdos matemáticos; por exemplo, as expressões algébricas, equações, funções, fórmulas matemáticas, entre outros. Uma parte interessante foi observar que os alunos entenderam em menos tempo e com mais facilidade a lógica apresentada, apesar de que demonstram dificuldades em realizar o mesmo com as letras que aparecem nos conteúdos citados durante as aulas de Matemática.

Depois, foi dada continuidade com as operações básicas no Python. Usando corretamente os símbolos matemáticos, a linguagem de programação pode funcionar de forma idêntica a uma calculadora. Entretanto, foi importante destacar atenção no uso das aspas, pois todo caractere sobre aspas é compreendido como um dado tipo `string` (texto). Assim, para obter resultado de uma conta, o uso das aspas é prescindível dentro

do comando `print`.

Por exemplo, para obter a soma de 2 com 3, o correto é inserir `print(2+3)`, recebendo `5` como resposta no console. Se adicionar as aspas, inserindo `print("2+3")`, aparecerá `2+3` no painel de saída.

Por fim, o restante do tempo disponível foi reservado para que os alunos pratiquem os símbolos ensinados: adição (`+`), subtração (`-`), multiplicação (`*`), divisão (`/`), e potenciação (`**`).

Aas figuras 3 e 4 mostram algumas capturas de telas dos celulares dos alunos enquanto testavam e depuravam códigos no aplicativo "Coding Python".

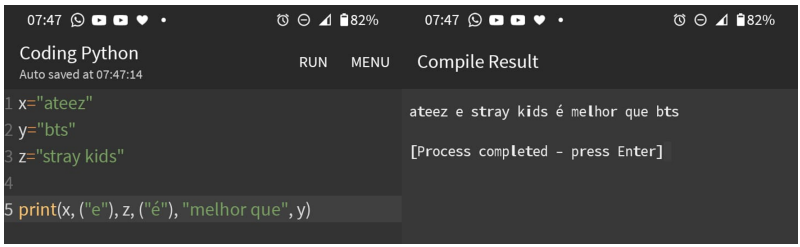

Figura 3: Captura de tela 1

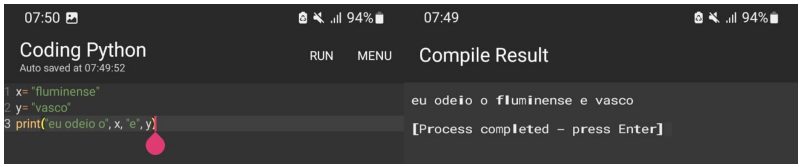

Figura 4: Captura de tela 2

Atividade sobre o comando `input`

Essa atividade tem como objetivo explicar o comando `input` que permitirá aos usuários inserirem dados no painel de saída. Para que os alunos tenham um melhor entendimento, foram revisados os assuntos das aulas anteriores, relembrando principamente as noções primordiais e a lógica de substituir variáveis. Pois, se antes as variáveis são definidas pelo programador no painel de código, agora, as variáveis serão definidas por usuários no painel de saída.

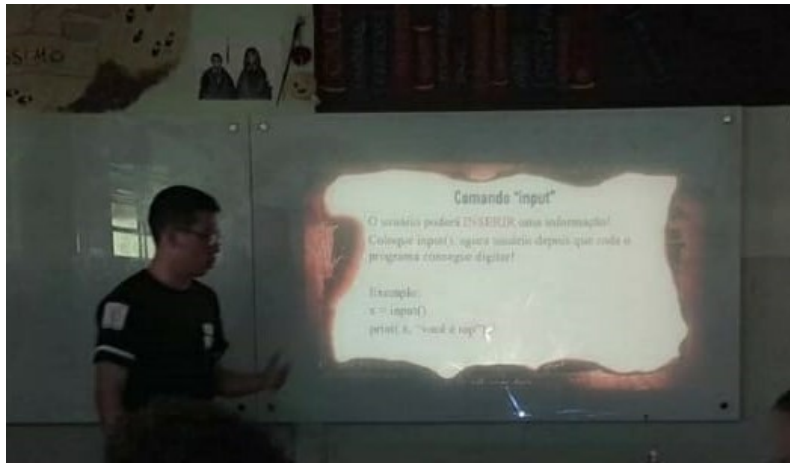

Figura 5: Ensinando comando `input`

Ao escrever e executar `input()`, os usuários terão permissão para digitarem no console. Para que os caracteres inseridos pelo usuário sejam manipulados como variável, a codificação deverá ser do tipo `x = input()`. Agora, para orientar o usuário a inserir uma informação específica, é possível adicionar uma descrição com aspas dentro do parênteses.

Por exemplo, ao inserir na primeira linha `x = input("Coloque o seu nome:")`, e na outra linha `print(x,"é legal")`, quando o usuário digitar seu nome e pressionar `enter`, a próxima linha do código será processada, e a máquina retornará `"nome do usuário" é legal`. Ou seja, quando o usuário informa o nome, a máquina armazena `x = "nome do usuário"`, e assim, na segunda linha, mostrará no painel de saída a junção do `x` e "`é legal`".

Após a realização de mais testes como o exemplo apresentado anteriormente, os estudantes conseguiram entender o novo comando. Agora, para manipular números, serão necessários 2 comandos complementares, o `int()` e `float()`. Em síntese, as informações dadas pelo usuário no console são originalmente dados tipo `string`, e assim, para que um número inserido seja interpretado como um valor numérico, precisa convertê-lo usando os comandos mencionados. Caso contrário, a máquina processará os números como caracteres de texto.

A diferença entre as funções `int()` e `float()` reside no fato de que a primeira converte o valor para um número inteiro, enquanto a segunda converte-o para um número racional, que pode incluir uma parte decimal. Essas distinções podem ser associadas aos conhecimentos matemáticos sobre conjuntos numéricos, revisando as noções de elementos contidos e relações de pertinência. Vale ressaltar que todo número inteiro é, de fato, um número racional, mas o inverso não é necessariamente verdadeiro.

Para finalizar, os estudantes tiveram que construir um algoritmo que perguntasse a idade do usuário, e que retornasse como respostas consecutivas a idade do programador e a diferença das idades, considerando que o usuário seja sempre mais novo que o programador. Nas figuras 6 e 7 abaixo, são algumas capturas de telas dos alunos testando códigos nessa aula.

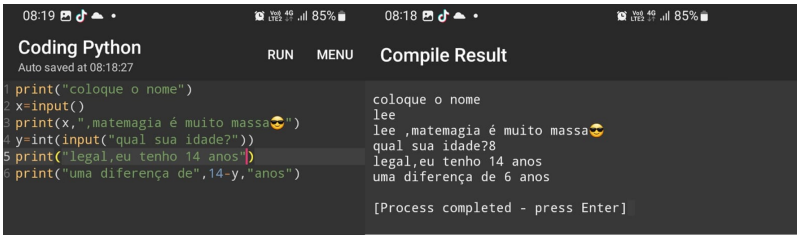

Figura 6: Captura de tela 3

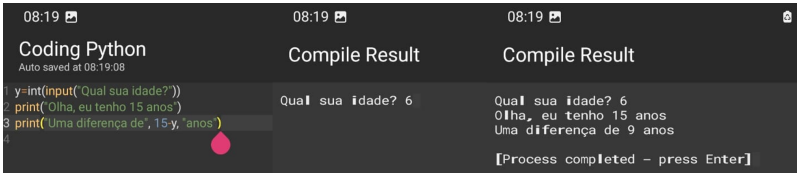

Figura 7: Captura de tela 4

Atividade sobre os operadores relacionais e dados booleanos

Essa atividade tem como objetivo introduzir operadores relacionais e dados booleanos frequentemente utilizados em estruturas condicionais que serão ensinados nas próximas atividades.

Os operadores relacionais desempenham um papel crucial na programação Python ao permitir comparações entre valores. O operador de igualdade (`==`) é utilizado para verificar se dois valores são iguais, enquanto o operador de diferença (`!=`) avalia a condição oposta. Além disso, os operadores de comparação incluem os maior que (`>`), menor que (`<`), maior ou igual a (`>=`), e menor ou igual a (`<=`), que permitem verificar relações de ordem entre valores. Já os dados booleanos representam o estado lógico, determinando se as relações estabelecidas são verdadeiras (`true`) ou falsas (`false`).

Para fixar melhor as ideias, foi elaborado exercícios simples para treinar o uso de todos operadores relacionais, comparando os números 2 e 3. E, depois, foi realizada uma pequena discussão sobre os casos em que os valores comparados são iguais. Por exemplo, se executarmos `2 > 2` e `2 >= 2`, o primeiro retornará `false` e o segundo `true`. Assim, podemos raciocinar de forma semelhante nos casos em que comparamos valores com variáveis. Por exemplo, nas desigualdades `x > 2` e `x >= 2`, devemos entender que, em relação ao número 2, a primeira é exclusiva e a segunda é inclusiva.

As análises anteriores tornam-se ainda mais relevantes quando os operadores booleanos `and` (e), `or` (ou) e `not` (não) foram ensinados. Estes são usados para juntar expressões condicionais. O `and` retorna `true` apenas quando ambas expressões forem verdadeiras. O `or` retorna `true` se pelo menos uma expressão for verídica. E o `not` inverte a resposta do dado booleano.

Nas estruturas condicionais que ainda serão ensinadas, os operadores booleanos também desempenham a função de criterizar variáveis. Por exemplo, ao usar a expressão `x > 2 and x < 3`, um intervalo de valores para `x` foi definido, percorrendo todos os números reais entre 2 e 3.

Essas lógicas, com os operadores relacionais e dados booleanos, podem ser relacionadas principalmente aos estudos sobre intervalos e inequações na Matemática. Esses conceitos fundamentais servem como base para outros temas, como conjunto de soluções, gráficos, probabilidade e estatística. Assim, observamos que, por

meio dessas aulas, os conteúdos matemáticos previamente considerados abstratos e de difícil compreensão foram mais bem consolidados.

Atividade sobre o comando `if`, `else` e `elif`

Essa atividade tem como objetivo abordar as estruturas condicionais.

Primeiramente, o comando `if` permite o programa executar blocos de código apenas se uma condição especificada for verdadeira.

Para exemplificar melhor, o exercício de calcular a diferença das idades da aula sobre comando `input` foi retomado. A diferença é que, agora, dois blocos de códigos distintos serão exigidos. Uma será executada apenas quando o usuário inserir uma idade acima de 23, e outra somente quando for menor ou igual a 23.

Após um tempo para os alunos realizarem suas tentativas, depuramos os códigos que não estavam de acordo com o solicitado. O principal erro observado foi a ausência do `int()` para converter a idade inserida pelo usuário em um valor numérico manipulável. E as outras falhas provinham de erros de pontuação e falhas de indentação. E assim, foi disponibilizado um gabarito alternativo conforme ilustra no Código 2.

```
x = int(input("Qual a sua idade:"))
if x > 23:
    print("Você nasceu antes de 2000")
if x <= 23:
    print("Você nasceu em 2000 ou depois")
```

Código 2: Gabarito alternativo do exercício sobre idade

Dando sequência, foi introduzido o comando `else` que também faz parte das estruturas condicionais no Python. Sua funçao é executar um novo bloco de códigos quando a condição fornecida ao `if` não for atendida. Logo, podemos simplificar o código do exercício sobre idade observe no Código 3.

```
x = int(input("Qual a sua idade:"))
if x > 23:
    print("Você nasceu antes de 2000")
else:
    print("Você nasceu em 2000 ou depois")
```

Código 3: Gabarito alternativo do exercício sobre idade 2

Por último, foi ensinado sobre o comando `elif` que aparece usualmente entre os comandos `if` e `else`, desempenhando o papel de adicionar mais uma regra, caso a anterior não seja satisfeita.

Utilizando novamente o exercício sobre idade para exemplificar, suponha que agora haja mais uma resposta diferente para os usuários entre 18 e 23 anos. Assim, o código pode ser modificado conforme mostra no Código 4.

```
x = int(input("Qual a sua idade:"))
if x > 23:
    print("você nasceu antes de 2000")
elif x>=18:
    print("Você já é maior de idade")
else:
    print("Você nasceu em 2000 ou depois")
```

Código 4: Exercício sobre idade modificado

Nesse último exemplo, houve algumas discussões interessantes, pois os alunos questionaram o que mudaria se fosse usado mais um comando `if` no lugar do `elif`. Segundo eles, não afetaria o resultado. Entretanto,

realizando testes com qualquer número acima de 23, é notável as diferenças no console, como apresentam as figuras 8 e 9.

```
x = int(input("Qual a sua idade:"))
if x > 23:
    print("Você nasceu antes de 2000")
elif x>=18:
    print("Você já é maior de idade")
else:
    print("Você nasceu em 2000 ou depois")
```

```
In [1]: runfile('C:/Users/ACER/Desktop/ELETIVA 9º ANO
2º SEM/untitled1.py', wdir='C:/Users/ACER/Desktop/
ELETIVA 9º ANO 2º SEM')
Qual a sua idade:28
Você nasceu antes de 2000
```

Figura 8: Teste do exercício sobre idade usando comando `elif`

```
x = int(input("Qual a sua idade:"))
if x > 23:
    print("Você nasceu antes de 2000")
if x>=18:
    print("Você já é maior de idade")
else:
    print("Você nasceu em 2000 ou depois")
```

```
In [2]: runfile('C:/Users/ACER/Desktop/ELETIVA 9º ANO
2º SEM/untitled1.py', wdir='C:/Users/ACER/Desktop/
ELETIVA 9º ANO 2º SEM')
Qual a sua idade:28
Você nasceu antes de 2000
Você já é maior de idade
```

Figura 9: Teste do exercício sobre idade usando comando `if`

Essas discussões e investigações contribuem de maneira significativa para racionínio matemático e operações envolvendo conjuntos, além do espírito científico e conhecimento empírico.

3.5. Desenvolvimento de algoritmos

Na construção dos algoritmos abaixo, foram incorporados todos os conhecimentos e conceitos trabalhados anteriormente e os conteúdos matemáticos relacionados e revisados. No total de 6 atividades, foram desenvolvidos seis algoritmos, apresentados a seguir:

1. Cálculo da idade a partir do ano de nascimento

   O primeiro algoritmo é facilmente construído partindo do mesmo raciocínio do exercício sobre diferença das idades quando foi ensinado o comando `input`; observe o código escrito por uma aluna na Figura 10. Se, antes, a forma de conjecturar era `Idade do Programador - Idade do Usuário` para calcular diferença das idades, agora, será usado `Ano atual - Ano de Nascimento do Usuário` para determinar a idade do usuário. Vale salientar, que no caso do exercício primordial, foram validados apenas os valores inferiores à idade do programador. No entanto, com habilidades e lógicas mais refinadas para programação, os alunos tornaram-se capazes de aperfeiçoar os códigos elaborados anteriormente, evitando entrada de valores que não forneceriam resposta coerente no painel de saída.

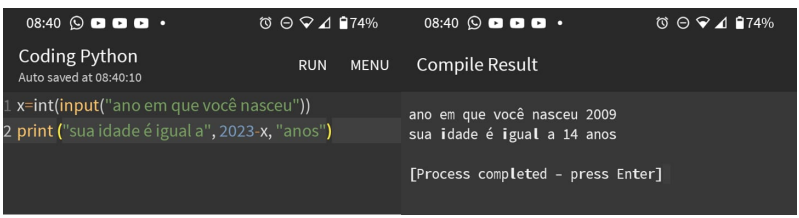

Figura 10: Captura de tela 5

A elaboração desse algoritmo permitiu que os alunos fortalecessem ainda mais sua compreensão em relação a expressões numéricas e algébricas.

2. Determinando padrinhos das turmas

Neste algoritmo, a ideia central é exercitar as habilidades dos alunos de construírem blocos de códigos na estrutura condicional. E também, pensar logica e minuciosamente para atender todos os tipos de casos possíveis. Além das três turmas do 9º ano, deve-se levar em conta quando um dado inserido não corresponde a nenhuma expressão condicional. Esse caso ocorre quando não foi bem especificada a resposta esperada, e, também, quando o dado inserido pelo usuário, de fato, está fora do escopo previsto pelas condições estabelecidas. No Código 5, é apresentado um modelo correto para esse algoritmo.

```
print("Responda com o número da série e a turma em letra maiúscula")
x = input("Qual a sua turma?")
if x == "9A":
    print("Seu padrinho é o Prof Lee")
elif x == "9B":
    print("Seu padrinho é o Prof Fernando")
elif x == "90":
    print("Seu padrinho é o Prof Torres")
else:
    print("Não aceitamos esse tipo de resposta")
```

Código 5: Modelo correto do algoritmo para determinar padrinhos das turmas

A partir desse algoritmo, os estudantes começaram a compreender que semelhante ao conteúdo de conjuntos, contagem e probabilidade, devem ter cautela ao considerar todos os casos, ou seja, devem garantir que a união de todos os subconjuntos forme o conjunto universo.

3. Cronograma das aulas da segunda-feira

Na construção desse algoritmo, a finalidade é exemplificar de forma simples como a programação pode auxiliar e otimizar nossa rotina. A atividade consiste em elaborar um cronograma das aulas da segunda-feira. Além dos comandos básicos, o operador booleano `or` pode ser útil, especificamente, quando temos dois tempos consecutivos da mesma matéria. No Código 6, é apresentado um modelo correto do algoritmo para a turma do 9º ano A.

```
x = int(input("Em qual tempo você se encontra? "))
if x == 1 or x==2:
    print("Eletiva de Matemática")
elif x==3:
    print("Aula de Ciências")
elif x==4 or x==5:
    print("Projeto de Vida")
elif x==6 or x==7:
    print("Aula de Artes")
elif x==8 or x==9:
    print("Aula de Língua Portuguesa")
else:
    print("Apenas aceitamos respostas de 1 a 9")
```

Código 6: Modelo correto do algoritmo cronograma das aulas da segunda-feira

No contexto desse algoritmo, é possível revisitar os conceitos associados aos conectivos lógicos de conjunção e disjunção em um conjunto solução. Adicionalmente, ao explorar estudos probabilísticos, é relevante mencionar os princípios Multiplicativo e Aditivo.

4. Cálculo da média bimestral

Esse algoritmo demonstra como a programação pode ter grandes utilidades no âmbito escolar. Não somente para os alunos, mas o professor também pode utilizar os códigos para facilitar seus trabalhos.

Foi considerado o caso comum em que se tem duas notas parciais no bimestre e a média seja calculada de forma aritmética.

No decorrer da construção, foi importante discutir sobre o armazenamento de dados na programação. Pois, um erro frequente observado é que, na criação de dois espaços para inserção de dados usando comando `input()`, os alunos costumam associar a duas variáveis `x`. Dessa forma, mesmo que não resulte em falhas no processamento do código, o Python irá tomar apenas o último resultado como valor do `x`.

A partir desta discussão, podemos fazer conexões com os conteúdos de expressões algébricas, equações e funções, explorando a compreensão de incógnitas e variáveis. Podemos citar em sistemas de equações as incógnitas que representam o mesmo valor. E, em funções, as relações entre variáveis dependentes e independentes, assuntos abordados nas aulas de Matemática.

No Código 7 a seguir, é mostrado um modelo correto do algoritmo desejado.

```
x = float(input("Digite a primeira nota: "))
y = float(input("Digite a segunda nota: "))
print("A média bimestral é", (x+y)/2)
```

Código 7: Modelo correto do algoritmo para calcular média bimestral

Uma forma alternativa e interessante de codificar seria construir mais uma variável z, e relacioná-la ao resultado da expressão $\frac{x+y}{2}$.

5. Determinando paridade de um número

Para a construção desse algoritmo, foi necessário introduzir uma operação especial do Python. Ao escrever % entre dois números, o resto da divisão do primeiro pelo segundo número é retornado como resultado.

Por outro lado, houve uma pequena discussão sobre os métodos de como determinar a paridade de um número. Praticamente todos os alunos afirmaram sobre o método de analisar o último algarismo: se terminasse com 0, 2, 4, 6 ou 8, o número seria par, e caso contrário, seria ímpar, terminando com 1, 3, 5, 7 ou 9.

No entanto, foi apresentada uma lógica mais conveniente para facilitar na escrita do código. Bascicamente, precisa observar que todo número par deixa resto 0 ao dividir por 2, enquanto todo número ímpar deixa resto 1. Dessa forma, um modelo correto para o algoritmo é como ilustrado no Código 8.

```
x = int(input("Digite um número inteiro: "))
if x%2 == 0:
    print("Esse número é par")
else:
    print("Esse número é impar")
```

Código 8: Modelo do algoritmo para determinar paridade de um número

6. Cálculo do discriminante Delta (Δ)

No último algoritmo, foi desenvolvido um código sobre cálculo do discriminante Delta que pertence à fórmula de Bhaskara, e que é tratado no conteúdo de equações polinomiais do 2º grau para o 9º ano.

Para o sucesso da construção, os estudantes precisavam criar 3 espaços para o usuário inserir os coeficientes da equação do 2º grau e reproduzir a fórmula de Delta no Python. De todos algoritmos, foi notado um contentamento maior dos alunos ao construir este, uma vez que a máquina resolve com facilidade algo que é um grande desafio para eles. Na Figura 11, é apresentada a ilustração deste código escrito corretamente por uma aluna.

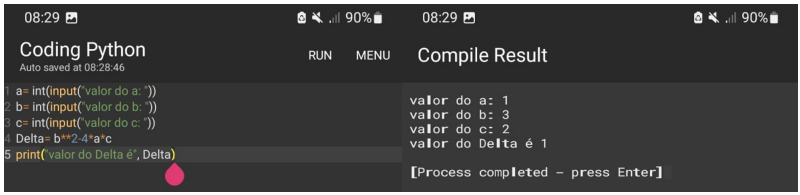

Figura 11: Captura de tela 6

Na maioria das aulas Matemáticas, as fórmulas são apresentadas de forma imediata, sem uma explicação ou investigação sobre o motivo de serem expressas da maneira apresentada. Como resultado, as reações dos alunos frequentemente envolvem surpresa, confusão e frustração, pois percebem essas fórmulas como um conjunto de letras com sinais matemáticos sem um sentido real coerente. No entanto, por meio deste algoritmo, os alunos conseguiram associar as letras aos coeficientes da equação e compreender as operações relacionadas.

3.6. Diário digital dos bruxos

Para encerrar as atividades, foi designada uma tarefa final que consistia na produção de um algoritmo interativo semelhante àquele apresentado na feira das eletivas. Um diário que simula novamente o diário de Tom Marvolo Riddle do Universo Harry Potter, e que será usado para interagir com a comunidade escolar no evento da culminância. Os alunos irão trabalhar em grupo para desenvolver partes do código, e, no fim, os melhores serão unificados em um algoritmo final.

A fase inicial do código, que cada grupo irá elaborar, deverá exibir uma memória marcante de cada membro ao longo do semestre. Além disso, é preciso criar um nome cativante para o grupo e listar os nomes individuais de cada integrante. No entanto, a identidade de cada memória não deve ser revelada, sendo responsabilidade do usuário adivinhar e enumerar corretamente. E no desfecho do código, deverá constar uma parte que efetua operações matemáticas.

Por fim, foram escolhidos os códigos mais bem escritos pelos alunos para integrar no algoritmo final. Para melhorar o fluxo na interação, foi utilizado o comando `time.sleep()` importado da biblioteca `time`, cuja função é introduzir pausas temporárias. E, também, uma estrutura de repetição, o comando `while`, foi usufruída para repetir o bloco de códigos caso tenha erros nos dados inseridos pelo usuário. No Código 9, são os códigos parciais feitos por um grupo de alunos que será composto no algoritmo final.

```
import time

x = input("Coloque seu nome: ")
time.sleep(1)

print("Prazer,",x)
time.sleep(1)

print("Nós somos tecnobruxos")
time.sleep(1)

print("Dominamos matemagia e tecnologias da sociedade contemporânea")
time.sleep(3)

print("Deixamos um poder especial nesse diário digital!")
time.sleep(2)
```

```
print("Podemos te dar acesso a esse poder, mas em troca, você precisa nos ajudar com nossas memórias...")
time.sleep(4)

print("Para criar esse diário digital mágico, embutimos nossas almas e uma das memórias mais importante de cada um de nós")
time.sleep(4)
print("Porém, muito tempo juntos, esquecemos quais memórias pertencem a qual de nós...")
time.sleep(4)
print("Ajude nos a identificar")
time.sleep(2)

print("Memória 1: meu gato morreu dia 14 de julho")
print("Memória 2: Fui na bahia e peguei chuva dia 20 de julho")
print("Memória 3: Aprendi a andar de bicicleta em dia 26 de janeiro")
print("Memória 4: Adoro livros, principalmente de Raphael Montes")
print("Memória 5: Eu desenhei as estrelas na sala quatro dia 12 de junho")
print("Memória 6: Eu li mo dao zu shi dia  6 de junho")
time.sleep(6)

print("Somos os  mais famosos da ala psiquiátrica, formados por Ana Julia Neves, Michelly, Cassiany, Bruna, Ana Eliza e Yasmin")
time.sleep(3)

z = input("Enumere na ordem dos nomes a memória de cada um:")
if z != "365124":
    time.sleep(2)
    print("Não sintonizamos com as memórias que você identificou...")
    print("Deve estar errado...")
    print("Você não terá acesso ao poder...")

else:
    time.sleep(2)
    print("ISSO MESMO! ESSAS SÃO AS MEMÓRIAS CORRETAS DE CADA UM DE NÓS")
    time.sleep(2)
    print("Deixarei você acessar o poder desse diário!")
    time.sleep(2)
    print("O poder desse diário é calcular POTENCIAÇÃO ")
    x=int(input("base:"))
    y=int(input("potência:"))
    print(x**y)
```

Código 9: Códigos parciais para algoritmo final

3.7. Culminância

Na Culminância, os estudantes apresentaram, por meio de cartazes, todos os conhecimentos adquiridos durante a disciplina. Para garantir atratividade, diversas atividades dinâmicas também foram contempladas. O destaque, no entanto, foi o algoritmo final, reconhecido por sua inovação tecnológica. A resposta positiva da comunidade escolar foi marcada por elogios, reações de surpresa e contentamento. Inclusive, vários alunos de outras turmas e séries questionaram o motivo de não terem tido a mesma oportunidade dessas aprendizagens. As figuras 12 e 13 mostram algumas fotos da culminância.

Figura 12: Culminância - foto 1

Figura 13: Culminância - foto 2

4. Resultados

A nossa proposta de abordar a Matemática de maneira ativa e lúdica resultou na criação de salas temáticas, incorporadas ao projeto "Estudantes no Controle" da escola, uma iniciativa da CGE-MS em parceria com a SED. A escola CMA foi premiada em primeiro lugar no final do projeto [1].

Ao analisar o comportamento e as produções dos estudantes durante o projeto, observamos que o raciocínio lógico é naturalmente estimulado na programação. A criação, execução e correção de códigos tornaram-se um processo significativo e prazeroso para a maioria dos alunos, estimulando o desenvolvimento de lógica e a revisão de conceitos matemáticos de maneira motivadora e relevante. Essa abordagem contrasta com aulas tradicionais de resolução de problemas matemáticos, pois, embora essas exijam tarefas cognitivas, frequentemente carecem de contextualização significativa para os alunos. Além disso, notamos que a porcentagem de alunas que dominaram o Python em menos tempo superou a dos alunos, respaldando a importância da diversidade de gênero na área de programação [6].

Em síntese, os resultados destacam que o domínio de conhecimentos computacionais contribui de maneira efetiva para a consolidação dos conteúdos matemáticos. A lógica de programação torna-se uma linguagem complementar que facilita a compreensão de abstrações e conjecturas matemáticas. Essas habilidades não apenas fortalecem a base conceitual, mas também preparam os estudantes para enfrentar os desafios tecnológicos contemporâneos, tornando o processo de ensino-aprendizagem da Matemática mais eficiente e dinâmico.

5. Conclusões

A partir do retorno e comentários de alunos, professores e demais membros da comunidade escolar, constatamos que nosso trabalho foi bem-sucedido e recebeu reconhecimento. O respaldo e estímulo que recebemos impulsionam-nos a prosseguir e expandir o projeto, com o objetivo de proporcionar, nos próximos anos, a oportunidade de aprendizado e aprimoramento em conhecimentos computacionais a todos os estudantes. No entanto, é essencial compreender que tanto a lógica computacional quanto a lógica matemática são habilidades e competências que se desenvolvem ao longo do tempo. Nesse sentido, não se deve esperar uma melhora significativa imediata no desempenho nas avaliações. Contudo, a relevância dessas habilidades no ensino da Matemática é indiscutível. Ao integrar as lógicas de programação no entendimento dos conteúdos matemáticos, os assuntos tornam-se mais coerentes e acessíveis

Para nossas próximas iniciativas, planejamos avançar na exploração de modelagens matemáticas. À medida que os conhecimentos fundamentais em linguagem de programação Python forem consolidados, almejamos aprofundar nossos estudos em heurísticas e modelos matemáticos para resolver e otimizar uma variedade de problemas, como planejamento financeiro, dimensionamento de lotes, transporte, *design* de rede, entre outros desafios. Além disso, pretendemos formar grupos de pesquisa com alunos motivados, interessados e com bom desempenho escolar, incentivando sua participação em programas como o Pictec (Programa de Iniciação Científica e Tecnológica de Mato Grosso do Sul) e eventos como a FetecMS (Feira de Tecnologias, Engenharias e Ciências de Mato Grosso do Sul).

Agradecimentos

Agradecemos à Fundect pelo incentivo e apoio financeiro, por fim, à UFMS pela oportunidade de trilhar o caminho da pesquisa científica.

Referências

[1] Agência de Notícias do Governo do Estado de Mato Grosso do Sul, *Cidadania é Celebrada por Estudantes em Dia de Premiação do Projeto da CGE de MS*, Disponível em: <https://agenciadenoticias.ms.gov.br/cidadania-e-celebrada-por-estudantes-em-dia-de-premiacao-do-projeto-da-cge-de-ms/>. Acesso em: 25 de dezembro de 2023.

[2] BOALER, Jo.l Mindsets: Unleashing Students' Potential through *Mathematica Creative Math, Inspiring Messages and Innovative Teaching*. John Wiley & Sons, 2016.

[3] Borges, Brunno de Oliveira. *A Linguagem Python como Ferramenta no Ensino Básico*. Uberaba-MG: Universidade Federal do Triângulo Mineiro, 2021. Mestrado Profissional em Matemática.

[4] CENPEC. *Ideb 2021: Cai o aprendizado, ressaltam-se as desigualdades*. Disponível em: <https://www.cenpec.org.br/noticias/ideb-2021-cai-o-aprendizado-ressaltam-se-as-desigualdades>. Acesso em: 14 de dezembro de 2023.

[5] Costa, A. C. M., Graça, R. J. S., Mota, C. V. A., Franco, A. A., Muniz, V. H. J., Maia, L. L., & Liese, T. M. (2017). Python: Será que é possível numa Escola Pública de Ensino Médio? *In Anais* do XXIII Workshop de Informática na Escola (WIE 2017) e VI Congresso Brasileiro de Informática na Educação (CBIE 2017). DOI: 10.5753/cbie.wie.2017.255.

[6] Dantas, C., Gouveia, A., & Fernández Venero, M. L. *Programação em Python: Inserindo mulheres na tecnologia*. Anais dos Workshops do VI Congresso Brasileiro de Informática na Educação (WCBIE 2017). DOI: 10.5753/cbie.wcbie.2017.1041

[7] FREIRE, Paulo *Pedagogia do Oprimido.* Rio de Janeiro: Paz e Terra, 1968. ISBN: 978-8577533709.

[8] INEP. *MEC e Inep divulgam resultados do Saeb e do Ideb 2021*. Disponível em: <https://www.gov.br/inep/pt-br/assuntos/noticias/saeb/mec-e-inep-divulgam-resultados-do-saeb-e-do-ideb-2021>. Acesso em: 14 de dezembro de 2023.

[9] Ministério da Educação. *Base Nacional Comum Curricular*. Disponível em: <http://basenacionalcomum.mec.gov.br/>. Acesso em: 19 de dezembro de 2023.

[10] Morais, D. C. (2021). *A Linguagem de Programação Python para o Ensino da Matemática* (Trabalho de Conclusão de Curso). Florianópolis: Universidade do Sul de Santa Catarina. Disponível em: <file:///C:/Users/ACER/Downloads/TCC_DIEGO_DE_MORAIS-2021%20(1).pdf-Ass%20(2)%20(5).pdf>. Acesso em: 13 de agosto de 2024.

[11] VYGOTSKY, Lev Semenovich. *Mind in Society: The Development of Higher Psychological Processes.* Harvard University Press, 1978.

Lee Jing Xuan
Universidade Federal de Mato Grosso do Sul
<leejingxuan2000@gmail.com>

Rubia Mara de Oliveira Santos

Universidade Federal de Mato Grosso do Sul
<rubia.oliveira@ufms.br>

Recebido: 18/01/2024
Publicado: 13/11/2024