Weberson Sarjob Ferreira Cruz

UMA PROPOSTA SOBRE O USO DO PYTHON PARA RESOLUÇÃO DE EXERCÍCIOS NO ENSINO FUNDAMENTAL ANOS FINAIS

Rondonópolis

2025

Weberson Sarjob Ferreira Cruz

UMA PROPOSTA SOBRE O USO DO PYTHON PARA RESOLUÇÃO DE EXERCÍCIOS NO ENSINO FUNDAMENTAL ANOS FINAIS

Dissertação de mestrado apresentada ao PROFMAT como parte dos requisitos exigidos para a obtenção do título de Mestre em Matemática

UNIVERSIDADE FEDERAL DE RONDONÓPOLIS

MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL

Orientadora: Prof. Dra Joelma Ananias Oliveira

Rondonópolis

2025

Dados Internacionais de Catalogação na Fonte
Ficha Catalográfica elaborada de forma automática com os dados fornecidos pelo(a) autor(a).

C955p Cruz, Weberson Sarjob Ferreira.
UMA PROPOSTA SOBRE O USO DO PYTHON PARA RESOLUÇÃO DE EXERCÍCIOS NO ENSINO FUNDAMENTAL ANOS FINAIS [recurso eletrônico] / Weberson Sarjob Ferreira Cruz. – Dados eletrônicos (1 arquivo : 105 f., il. color., pdf). – 2025.

Orientador(a): Joelma Ananias de Oliveira.
Dissertação (mestrado) – Universidade Federal de Rondonópolis, Instituto de Ciências Exatas e Naturais, Programa de Pós-Graduação em Matemática em Rede Nacional, Rondonópolis, 2025.
Inclui bibliografia.

1. PROGRAMAÇÃO EM PYTHON. 2. ENSINO DE MATEMÁTICA. 3. TECNOLOGIAS EDUCACIONAIS. I. Oliveira, Joelma Ananias de, *orientador*. II. Título.

MINISTÉRIO DA EDUCAÇÃO

UNIVERSIDADE FEDERAL DE RONDONÓPOLIS - UFR

PRÓ-REITORIA DE ENSINO DE PÓS-GRADUAÇÃO E PESQUISA - PROPGP/UFR

PROGRAMA DE PÓS-GRADUAÇÃO EM MATEMÁTICA - MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL - PROFMAT/UFR.

FOLHA DE APROVAÇÃO

TÍTULO: UMA PROPOSTA SOBRE O USO DO PYTHON PARA RESOLUÇÃO DE EXERCÍCIOS NO ENSINO FUNDAMENTAL ANOS FINAIS**.**

AUTOR : MESTRANDO WEBERSON SARJOB FERREIRA CRUZ.

Dissertação submetida ao programa de pós-graduação do Mestrado Profissional em Matemática em Rede Nacional-PROFMAT, da Universidade Federal de Rondonópolis-UFR, vinculado ao curso de Matemática da UFR, como requisito parcial para obtenção do grau de Mestre em Matemática.

Dissertação defendida e aprovada em **20/05/2025.**

OBS.: Somente os membros titulares da banca examinadora assinam este documento.

COMPOSIÇÃO DA BANCA EXAMINADORA

1. Profa. Dra. Joelma Ananias de Oliveira (Presidente da Banca /Orientadora);

2. Prof. Dr. Emerson Dionísio Belançon (Membro interno titular/UFR);

3. Profa. Dra. Luciana Lee (Membro Externo Titular/UFES);

4. Prof. Dr. Aroldo José de Oliveira (Membro Interno Suplente/UFR);

5. Prof. Dr. Marcus Vinícius de Andrade Neves (Membo Interno Suplente/UFR);

6. Profa. Dra. Lia Corrêa da Costa (Membro Externo Suplente/IFMT).

Rondonópolis-MT, **20/05/2025.**

Documento assinado eletronicamente por **Emerson Dionisio Belançon**, **Docente - UFR**, em 21/05/2025, às 15:20, conforme horário oficial de Brasília, com fundamento no art. 6º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Joelma Ananias de Oliveira**, **Docente - UFR**, em 21/05/2025, às 15:21, conforme horário oficial de Brasília, com fundamento no art. 6º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Luciana Lee**, **Usuário Externo**, em 21/05/2025, às 15:45, conforme horário oficial de Brasília, com fundamento no art. 6º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://sei.ufr.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0516981** e o código CRC **7A19A954**.

Referência: Processo nº 23853.005682/2025-78 SEI nº 0516981

Agradecimentos

Agradecer é um gesto de reconhecimento e carinho por aqueles que, de alguma forma, contribuíram para uma trajetória. Nesta caminhada intensa e desafiadora ao longo do curso, vivi inúmeros encontros e desencontros, aprendizados e superações. Por isso, o mínimo que posso fazer é expressar minha mais sincera gratidão.

Em primeiro lugar, agradeço a Deus, pela força, saúde e sabedoria concedidas em cada etapa dessa jornada. A Ele, todo o meu louvor por nunca ter me deixado desistir, mesmo nos momentos mais difíceis.

Estendo meus agradecimentos aos colegas de curso, com quem compartilhei vivências marcantes e aprendizados mútuos. Em especial, às queridas Simone e Grazi, que tornaram essa caminhada mais leve, alegre e inesquecível. Aos professores, minha mais profunda admiração e respeito. Foram eles que, com dedicação e compromisso, nos guiaram, acreditaram em nosso potencial e nos proporcionaram o conhecimento necessário para seguirmos com autonomia e confiança. De forma especial, agradeço à Professora Dra. Joelma, pelo empenho, incentivo e atenção que marcaram profundamente minha formação.

Não poderia deixar de prestar minha homenagem mais emocionante àqueles que são minha base e minha razão: meus pais, Antônio José e Edilenir Ferreira. A vocês, devo tudo o que sou — a vida, os valores, o caráter e o amor incondicional. Foram dias difíceis, de muito esforço, cansaço e desânimo, mas em nenhum momento deixei de sentir o apoio silencioso e constante de vocês. Sei que essa conquista é também de vocês, e tenho plena certeza de que estarão sempre na primeira fila, aplaudindo cada nova vitória com entusiasmo e amor.

Por tudo isso, meu muito obrigado. Amo vocês!

“ Eu acho que todos neste país deveriam aprender a programar um computador. Deveriam aprender uma linguagem de programação, pois te ensina como pensar ”

(Steve Jobs)

Resumo

Este estudo aborda a aplicação da linguagem de programação Python como ferramenta educacional no ensino de conceitos matemáticos, com ênfase na resolução de exercícios propostos no material estruturado disponibilizado pelo governo do Estado de Mato Grosso. A dissertação propõe uma abordagem detalhada e didaticamente orientada para o uso de Python, esclarecendo conceitos fundamentais, como o emprego de variáveis, operações básicas e estruturas de controle, com o propósito de capacitar não apenas os estudantes, mas também os professores, para a utilização pedagógica dessa linguagem de programação. Paralelamente, o estudo examina a compatibilidade dessa metodologia com os princípios e competências estabelecidos pela Base Nacional Comum Curricular (BNCC), destacando a relevância da inclusão de conteúdos ligados à programação no currículo escolar, como estratégia para desenvolver o raciocínio lógico, a capacidade de resolução de problemas e a autonomia intelectual dos estudantes. Embora se configure como uma proposta didática aplicada, a pesquisa delineia uma proposta teórico-metodológica robusta, discutindo possibilidades, benefícios e desafios. Essa sugestão indica que o uso da linguagem Python em sala de aula pode não apenas favorecer a compreensão de conteúdos matemáticos, sobretudo aqueles de caráter geométrico e algébrico, mas também contribuir para a formação de competências em programação, reconhecidamente essenciais para a inserção crítica e criativa no mundo digital contemporâneo, especialmente para os estudantes que almejam seguir por esse campo. Por fim, a pesquisa reforça e propõe a utilização dessa linguagem de programação como condição indispensável para a consolidação de práticas pedagógicas inovadoras e para o atendimento às novas demandas formativas da educação básica.

Palavras-chave: Programação em Python; Ensino de Matemática; Tecnologias Educacionais.

Abstract

This study addresses the application of the Python programming language as an educational tool in the teaching of mathematical concepts, with an emphasis on solving exercises proposed in the structured material made available by the government of the State of Mato Grosso. The dissertation proposes a detailed and didactically oriented approach to the use of Python, clarifying fundamental concepts such as the use of variables, basic operations, and control structures, with the aim of enabling not only students but also teachers to use this programming language in pedagogical contexts. In parallel, the study examines the compatibility of this methodology with the principles and competencies established by the Base Nacional Comum Curricular (BNCC), highlighting the relevance of including programming-related content in the school curriculum as a strategy to foster logical reasoning, problem-solving skills, and students' intellectual autonomy. Although characterized as an applied didactic proposal, the research outlines a robust theoretical and methodological framework, discussing possibilities, benefits, and challenges. This proposition indicates that the use of the Python language in the classroom can not only enhance the understanding of mathematical content — especially those of a geometric and algebraic nature — but also contribute to the development of programming skills, which are widely recognized as essential for critical and creative engagement in the contemporary digital world, particularly for students who intend to pursue this field. Finally, the research reinforces and proposes the use of this programming language as an indispensable condition for consolidating innovative pedagogical practices and for meeting the new formative demands of basic education.

Keywords: Python Programming; Mathematics Education; Educational Technologies.

Lista de ilustrações

Sumário

1 Introdução

A educação, por natureza, é continuamente desafiada a se transformar e se adaptar às mudanças tecnológicas e culturais que moldam o mundo contemporâneo. Com o avanço das tecnologias, especialmente desde as últimas décadas do século XX, várias áreas do conhecimento, incluindo o ensino da matemática, foram profundamente impactadas. A globalização e a rápida disseminação das Tecnologias da Informação e Comunicação (TICs) têm exigido que as instituições de ensino incorporem essas ferramentas ao currículo escolar, oferecendo aos estudantes não apenas uma educação matemática tradicional, mas também competências tecnológicas que são cada vez mais necessárias.

A linguagem Python se destaca por ser versátil, de fácil compreensão e aplicável em diversas áreas, incluindo a solução de problemas matemáticos. Sua sintaxe clara facilita o aprendizado, sendo ideal para iniciantes, inclusive alunos do ensino fundamental, que podem, assim, desenvolver habilidades matemáticas junto a competências como raciocínio lógico e pensamento crítico.

O contexto educacional brasileiro apresenta desafios específicos para a implementação de tecnologias no ensino, devido à diversidade regional e às disparidades socioeconômicas. Muitos estudantes ainda enfrentam dificuldades de acesso a tecnologias modernas, e as escolas nem sempre dispõem de infraestrutura adequada. Por isso, é fundamental adotar soluções criativas, como o uso de plataformas de programação *online*, como a Programiz, que permitem o aprendizado prático do Python sem a necessidade de softwares instalados, tornando o conhecimento acessível a um maior número de alunos.

A Base Nacional Comum Curricular (BNCC), que norteia a educação brasileira, enfatiza a importância das competências digitais no processo de ensino-aprendizagem. Ao reconhecer o papel essencial das TICs na educação atual, a BNCC orienta escolas e educadores a adaptarem suas práticas para incluir elementos essenciais do século XXI, fundamentais para o desenvolvimento integral dos estudantes.

Este trabalho propõe a integração da linguagem de programação Python ao ensino de Matemática, com o intuito de potencializar as práticas docentes por meio de uma abordagem computacional. Analisa-se a implementação da linguagem Python com o compilador *online* Programiz, explorando conceitos por meio da programação. Este estudo procura mostrar não apenas as melhorias no desempenho escolar dos alunos, mas também como a introdução do Python pode aumentar sua motivação e interesse pelo aprendizado.

O segundo capítulo desta dissertação apresenta uma análise abrangente do contexto da BNCC e do uso de tecnologias no ensino da matemática. Ao longo das últimas décadas, o ensino da matemática passou por uma série de transformações, enfrentando desafios e

registrando avanços em práticas pedagógicas. Será destacada a necessidade de estratégias inovadoras que promovam uma aprendizagem significativa, com ênfase em metodologias ativas e no uso de tecnologias digitais, fundamentais para motivar e engajar os estudantes.

O terceiro capítulo abordará a linguagem de programação Python como um recurso didático no ensino de Matemática. A princípio, será apresentada uma visão geral sobre o Python, destacando sua origem, evolução e consolidação como uma das linguagens de programação mais utilizadas no mundo. Sua estrutura simples e, ao mesmo tempo, robusta favorece a compreensão de conceitos matemáticos, tornando-o uma ferramenta estratégica no ensino de disciplinas da área de exatas. Em um segundo momento, será discutido o potencial do Python enquanto ferramenta educacional, ressaltando as características que o tornam especialmente apropriado para aplicações matemáticas, bem como sua relevância em distintas áreas do conhecimento. Por fim, será explorado o uso do ambiente *online* Programiz, um compilador que oferece praticidade, interatividade e recursos essenciais ao desenvolvimento de atividades pedagógicas envolvendo programação — especialmente voltado aos educandos do estado de Mato Grosso, cuja realidade tecnológica, marcada pelo uso de *Chromebooks*, não permite a instalação de *softwares*.

O quarto capítulo foca na aplicação prática do Python como recurso pedagógico no ensino da Matemática. Inicialmente, será apresentada a preparação do ambiente de programação utilizando o Programiz, uma plataforma *online* que permite aos alunos escrever e testar códigos sem a necessidade de instalar *softwares*, o que é especialmente útil para estudantes que utilizam *Chromebooks*. Em seguida, será feito o detalhamento dos códigos desenvolvidos, incluindo o uso de bibliotecas, funções e os principais tipos de dados, como inteiros, números de ponto flutuante e *strings*. Também serão abordadas as operações básicas, como adição, subtração, multiplicação, divisão, exponenciação e ordem de resolução. Além disso, serão discutidas as estruturas condicionais, ajudando os estudantes a entenderem como ensinar o computador a "pensar". Por fim, será apresentada uma explicação clara sobre variáveis e o funcionamento dessas operações dentro da linguagem computacional.

O quinto capítulo estabelece uma conexão direta entre teoria e prática, por meio da resolução de problemas matemáticos com o uso da linguagem Python. A proposta é apresentar exemplos concretos, que vão desde situações mais simples até desafios matemáticos de maior complexidade, todos resolvidos no ambiente *online* Programiz. Os exercícios utilizados pertencem ao material estruturado da rede pública do estado de Mato Grosso, e a resolução com Python tem como objetivo facilitar a visualização de conceitos, automatizar cálculos e tornar o aprendizado mais dinâmico. Ao longo do capítulo, os conteúdos abordam temas como potências e raízes, equações do segundo grau, semelhança de triângulos, teorema de Pitágoras, trigonometria, relações métricas na circunferência e volume de cilindros, cada seção traz atividades interativas, promovendo um aprendi-

zado interdisciplinar e colaborativo, que estimula o raciocínio lógico, a criatividade e a autonomia dos estudantes.

Ao final desta dissertação, busca-se apresentar uma proposta inovadora de integração da linguagem Python ao ensino da Matemática, voltada especialmente para estudantes que demonstram interesse por programação e tecnologias computacionais. A ideia central é mostrar como essa abordagem pode tornar o processo de aprendizagem mais atrativo, interativo e conectado com os saberes digitais que permeiam a vida cotidiana dos alunos. Ao utilizar o Python como ferramenta pedagógica, o ensino da Matemática ganha novas dimensões, permitindo que conceitos abstratos sejam explorados de maneira prática, visual e lógica. Essa proposta revela-se particularmente interessante por promover um ensino mais engajado e desafiador, que estimula o pensamento computacional, a criatividade e a resolução de problemas. Além disso, vale ressaltar como as tecnologias podem impulsionar uma forma de aprender mais interativa, participativa e diferente da tradicional. Elas contribuem para o desenvolvimento de habilidades importantes no mundo atual, cada vez mais marcado pela presença da tecnologia.

2 A BNCC e o uso de tecnologias no ensino da matemática

Ao longo dos séculos, o ensino de Matemática tem passado por uma expressiva evolução, ajustando-se às mudanças sociais, tecnológicas e educacionais. O modelo tradicional, caracterizado por aulas centradas no professor, com ênfase na memorização e no domínio de técnicas algorítmicas, tem gradualmente cedido espaço a metodologias mais dinâmicas. Estas, por sua vez, priorizam a compreensão conceitual e o aprimoramento da capacidade de resolução de problemas. Tais transformações evidenciam a crescente demanda por uma formação que permita aos alunos não apenas entender os fundamentos matemáticos, mas também aplicá-los em contextos variados, promovendo uma aprendizagem alinhada às exigências do mundo contemporâneo.

A Base Nacional Comum Curricular (BNCC) destaca a Matemática como um componente essencial na formação integral dos estudantes, atribuindo a ela um papel significativo no desenvolvimento de habilidades como o raciocínio lógico e crítico, além da capacidade de solucionar problemas. Esse entendimento reflete a importância de uma abordagem que valorize não apenas a competência técnica, mas também a formação de sujeitos capazes de atuar de forma reflexiva e autônoma na sociedade (BRASIL, 2017).

Ao reconhecer a Matemática como elemento central na formação integral dos estudantes, a BNCC reafirma seu papel histórico, não apenas como um campo técnico, mas como uma ferramenta indispensável para o desenvolvimento de competências fundamentais para a vida em sociedade.

Além disso, a ênfase em formação de sujeitos reflexivos e autônomos reforça a perspectiva de que a Matemática vai além de fórmulas e cálculos. Trata-se de um campo que conecta o pensamento estruturado com a ação prática, capacitando os estudantes a interpretar e transformar a realidade em que vivem. Alinhada às diretrizes da BNCC, ela propõe um ensino que supera a mera reprodução de técnicas, valorizando a construção de significados e a aplicação contextualizada do conhecimento.

O domínio da Matemática é destacado pela BNCC como indispensável para todos os estudantes da Educação Básica, não apenas por sua relevância prática no mundo contemporâneo, mas também pelo impacto significativo na formação de cidadãos críticos e conscientes de suas responsabilidades sociais (BRASIL, 2017).

Essa perspectiva reconhece a matemática como uma competência essencial, cuja importância vai além dos cálculos ou resoluções de problemas técnicos, contribuindo para o desenvolvimento integral dos estudantes e para a formação cidadã. A BNCC enfatiza

que o ensino da matemática deve promover a autonomia dos alunos, capacitando-os a compreender e enfrentar situações reais por meio de raciocínio lógico e crítico.

Nesse contexto, a Matemática assume um papel central como instrumento de inclusão social, ao garantir uma educação que prepare os estudantes para atuar de maneira ativa e consciente na sociedade. Além disso, ao evidenciar a Matemática como um meio para enfrentar desafios concretos, a BNCC reafirma que suas aplicações transcendem os limites escolares. Habilidades como o raciocínio lógico não se restringem ao âmbito técnico, mas também capacitam os alunos a tomar decisões fundamentadas e propor soluções em diversos contextos, ampliando suas oportunidades de participação social e cidadania ativa.

O desenvolvimento das habilidades matemáticas pode ser potencializado por diferentes formas de organização do aprendizado, envolvendo a análise de situações do cotidiano, de outras áreas do saber e da própria Matemática. Nesse contexto, processos como a resolução de problemas, a investigação, o desenvolvimento de projetos e a modelagem destacam-se como metodologias fundamentais. Além de promoverem a aprendizagem, essas abordagens são instrumentos para o desenvolvimento de competências essenciais, como o raciocínio lógico, a comunicação matemática, a argumentação e a representação. Adicionalmente, contribuem para o fortalecimento do pensamento computacional, relevante no cenário educacional contemporâneo (BRASIL, 2017).

A BNCC ressalta que o desenvolvimento das habilidades matemáticas se beneficia de práticas pedagógicas diversificadas, capazes de conectar a Matemática ao cotidiano, a outras áreas do conhecimento e ao universo matemático. Essa perspectiva valoriza um aprendizado que transcende a memorização ou repetição de procedimentos, promovendo o engajamento ativo dos estudantes por meio de atividades significativas.

Nos últimos anos, a integração de tecnologias no campo educacional tem transformado profundamente as formas de aquisição e compartilhamento do conhecimento. Na educação matemática, essa evolução se destaca significativamente, contribuindo para enfrentar desafios, como a desmotivação dos estudantes e a dificuldade em demonstrar a aplicabilidade prática dos conceitos ensinados, potencializando o envolvimento dos alunos, mas também ampliando as possibilidades de contextualização e dinamização do ensino tornando-o mais acessível e alinhado às demandas contemporâneas.

De acordo com as competências gerais da BNCC, é essencial que os estudantes desenvolvam a capacidade de utilizar ferramentas e processos matemáticos, incluindo tecnologias digitais, para resolver problemas cotidianos, sociais e de outras áreas do conhecimento. Essa abordagem promove a aplicação prática da matemática, integrando-a ao contexto real e permitindo a validação de estratégias e resultados. Assim, o ensino de matemática se torna mais significativo, estimulando o pensamento crítico e a resolução de problemas de forma colaborativa e contextualizada (BRASIL, 2017).

No entanto, é importante observar que essa integração mais profunda da Matemática com o uso de tecnologias e a resolução de problemas aplicados é mais evidenciada nas diretrizes do ensino médio, enquanto nos anos finais do ensino fundamental a abordagem ainda é predominantemente voltada à consolidação de conceitos básicos e ao desenvolvimento inicial do raciocínio matemático. Apesar disso, o ensino fundamental pode e deve incluir práticas que introduzam os estudantes a essas competências, de forma gradual, preparando-os para uma progressão mais rica no ensino médio.

No Brasil, a incorporação de tecnologias na educação tem enfrentado desafios significativos, como as desigualdades no acesso, a resistência cultural e a necessidade de formação adequada dos docentes. Apesar disso, iniciativas de políticas públicas, como a implementação da BNCC, têm incentivado a integração das Tecnologias da Informação e Comunicação (TICs) no sistema educacional, com o objetivo de assegurar que todos os estudantes estejam preparados para as demandas do mundo digital.

A evolução do ensino da matemática, portanto, reflete a adaptação constante às necessidades de uma sociedade em mudança, percebe-se isso pós pandemia em 2019 a crescente evolução. Essa evolução também reflete a incorporação de recursos tecnológicos e metodologias inovadoras no ensino, como o uso de ferramentas digitais, a modelagem matemática e a investigação científica. Esses elementos enriquecem a prática pedagógica, tornando a aprendizagem mais envolvente e significativa, além de preparar os estudantes para os desafios de um mundo cada vez mais tecnológico e interconectado.

A capacitação de professores em tecnologias educacionais é essencial para que inovações no ensino sejam bem-sucedidas. O uso eficaz de ferramentas tecnológicas em sala de aula depende diretamente das habilidades e da vontade dos educadores. Com a crescente necessidade de competências digitais no ensino, a formação contínua dos professores tem sido uma prioridade. Esse esforço busca não só acompanhar as mudanças tecnológicas, mas também incentivar práticas de ensino mais dinâmicas e focadas no aluno.

Segundo Souto (2014):“a velocidade exponencial com que as tecnologias digitais imprimem mudanças em nossas vidas nos impõe novos desafios”.

Além disso, o uso de tecnologias no ensino da Matemática oferece oportunidades para tornar a aprendizagem mais dinâmica e significativa. Recursos como softwares matemáticos, simuladores, aplicativos e plataformas digitais permitem uma abordagem mais visual, interativa e aplicada, facilitando a compreensão de conceitos abstratos e conectando-os a situações práticas e reais. Contudo, para que essas ferramentas sejam integradas de maneira eficaz, é necessário um esforço de formação docente e uma visão pedagógica que favoreça o uso crítico e reflexivo das tecnologias.

É notável que o cenário educacional passou por uma transformação significativa para aqueles que estavam em sala de aula antes e depois da pandemia de COVID-19. Durante

esse período, foi necessário adaptar-se a novas realidades, como o uso de aulas *online*, ferramentas digitais e planilhas eletrônicas. Como professores, enfrenta-se o desafio de reinventar-se, incorporando tecnologias e estratégias pedagógicas inovadoras para assegurar a continuidade da aprendizagem em meio às adversidades. Essas mudanças marcaram não apenas uma evolução nas práticas educacionais, mas também demandaram uma resiliência e uma adaptação constante ao novo contexto.

De acordo com Lima e Santos (2020), a pandemia de COVID-19 transformou radicalmente os cenários educacionais, exigindo adaptações estruturais significativas para garantir o acesso à aprendizagem. Apesar dos esforços realizados pelos professores, o processo de ensino enfrentou inúmeros desafios devido às limitações impostas pelo contexto pandêmico.

Os professores, no centro desse processo, precisaram rapidamente reinventar suas práticas pedagógicas. Isso envolveu não apenas a adoção de novas ferramentas tecnológicas, mas também a ressignificação de metodologias para atender à diversidade de alunos, muitos dos quais enfrentaram limitações significativas, como falta de acesso à internet, equipamentos tecnológicos adequados ou um ambiente propício para o estudo em casa.

Além disso, o distanciamento físico implicou uma perda de interação social entre alunos e professores, elemento essencial para o desenvolvimento integral do estudante. O ensino remoto, por mais que tenha sido uma solução emergencial viável, evidenciou a dificuldade de criar um ambiente de aprendizagem efetivo e engajador sem a presença física, o que afetou a motivação e o desempenho de muitos estudantes.

A citação também ressalta os esforços dos professores, que se destacaram por sua resiliência e criatividade. Mesmo diante de desafios como sobrecarga de trabalho, adaptação a plataformas digitais e, em muitos casos, falta de formação específica, os educadores demonstraram compromisso com a educação, buscando alternativas para minimizar os impactos negativos da crise.

Portanto, o contexto descrito por Lima e Santos (2020), não só ilustra as dificuldades enfrentadas durante a pandemia, mas também aponta para a necessidade de políticas públicas educacionais mais robustas. É imperativo investir em infraestrutura tecnológica, formação continuada de professores e ações que combatam as desigualdades no acesso à educação, garantindo que crises futuras possam ser enfrentadas de maneira mais equitativa e eficaz.

Essa análise demonstra como momentos de crise revelam vulnerabilidades do sistema educacional, como também podem se tornar catalisadores para mudanças estruturais que promovam uma educação mais inclusiva e resiliente.

Conforme BRASIL (2017) a BNCC destacou a necessidade de investir na formação docente para a integração de tecnologias no ensino, em conformidade com as novas

diretrizes curriculares que enfatizam o desenvolvimento das competências relacionadas às Tecnologias da Informação e Comunicação (TICs) pelos estudantes . Nesse contexto, iniciativas governamentais, como o Programa Nacional de Tecnologia Educacional (ProInfo) têm desempenhado um papel essencial ao oferecer capacitação em tecnologias digitais e práticas pedagógicas, visando assegurar uma inclusão tecnológica uniforme e equitativa no ambiente escolar público.

Segundo o Ministério da Educação (BRASIL, 2024): "o ProInfo visa integrar tecnologias ao ensino público e tem como objetivo qualificar os professores para a utilização pedagógica da informática na rede pública de educação básica". Fazendo com que os estudantes tenham acesso à informática em escolas públicas.

2.1 Benefícios de aprender programação no ensino fundamental

A inclusão do ensino de programação no currículo do Ensino Fundamental tem sido amplamente respaldada por educadores, especialmente nas áreas de ciências exatas, como a Matemática, devido à sua capacidade de fomentar o desenvolvimento de habilidades cognitivas, pensamento lógico e estratégias de resolução de problemas. Valente, Freire e Arantes (2018) exploram o papel histórico e contemporâneo das tecnologias na educação, destacando tendências futuras e reflexões sobre os desafios e as possibilidades do uso da informática no processo educacional. A introdução da programação nesse nível de ensino contribui para a construção de competências essenciais no contexto do século XXI, como a criatividade, o raciocínio lógico e a aptidão para enfrentar desafios complexos. Além disso, essa formação inicial tem o potencial de melhorar significativamente o desempenho dos estudantes tanto em Matemática quanto em outras disciplinas.

A BNCC estabelece diretrizes para a educação básica no Brasil, que, de maneira indireta incluem a integração de habilidades de programação, como o uso da linguagem Python, no currículo escolar. Em 2022, o Conselho Nacional de Educação (CNE) aprovou normas específicas sobre Computação na Educação Básica, complementando a BNCC. Essas normas foram formalizadas na Resolução CNE/CEB nº 1, de 4 de outubro de 2022, que orienta a implementação de conteúdos de computação, incluindo programação, em todas as etapas da educação básica.

Para o Conselho Nacional de Educação:

> No ensino fundamental, há de se destacar o atendimento à diretriz de compreender a computação como uma área de conhecimento que contribui para explicar o mundo atual e ser um agente ativo e consciente de transformação capaz de analisar criticamente seus impactos sociais, ambientais, culturais, econômicos, científicos, tecnológicos, legais e éticos (BRASIL, 2022a).

Para Brasil (2022a), no ensino fundamental, é fundamental compreender a computação como uma área de conhecimento que não apenas explica o mundo atual, mas também capacita os estudantes a serem agentes conscientes e críticos, aptos a analisar seus impactos sob diversas perspectivas, incluindo sociais, ambientais, culturais, econômicas, científicas, tecnológicas, legais e éticas.

> A presente Resolução define normas sobre Computação na Educação Básica, em complemento à Base Nacional Comum Curricular (BNCC) na seguinte conformidade: § 1º Processos e aprendizagens referentes à Computação na Educação Básica devem ser implementados considerando a BNCC, o disposto na legislação, nas normas educacionais e no aqui disposto. § 2º O desenvolvimento e formulação dos currículos deve considerar as tabelas de competências e habilidades anexas. § 3º A formação inicial e continuada de professores deve considerar o aqui disposto (BRASIL, 2022b).

Ao destacar que os *feedbacks* das tecnologias podem "organizar ou reorganizar o pensamento humano", Souto (2014) propõe uma reflexão profunda sobre a interatividade que essas ferramentas promovem. Diferentemente de materiais tradicionais, as tecnologias digitais oferecem respostas dinâmicas e adaptáveis, que podem tanto corrigir erros como propor caminhos alternativos para a resolução de problemas. Isso favorece o desenvolvimento de habilidades metacognitivas, permitindo que o estudante compreenda seus próprios processos de aprendizagem e os refine com base nos retornos que recebe.

> Aprender a programar exige que os estudantes decomponham problemas complexos em partes menores, abordando cada uma de forma lógica. Essa habilidade é particularmente valiosa em Matemática, onde a resolução de problemas é uma competência central. Valente (2018) destaca que "a habilidade de decompor problemas complexos é fundamental para a matemática e é desenvolvida naturalmente no aprendizado de programação" (VALENTE; FREIRE; ARANTES, 2018, p.54)

No contexto do ensino de Matemática, essa habilidade transcende o simples cálculo, envolvendo processos como identificar padrões, formular estratégias e verificar soluções. A decomposição lógica de problemas permite aos estudantes uma abordagem sistemática para resolver questões complexas, ajudando-os a estruturar seus pensamentos e, ao mesmo tempo, compreender as interconexões entre os diferentes elementos do problema.

O aprendizado de programação também agrega um componente prático e interdisciplinar ao ensino da Matemática. Por exemplo, ao implementar um algoritmo para calcular a soma de uma série numérica, os estudantes não apenas resolvem o problema matemático, mas também experimentam na prática conceitos como variáveis. Isso os aproxima de um entendimento mais profundo, no qual a Matemática e a Computação se complementam para resolver problemas reais.

Além disso, Valente (2018) ressalta um ponto essencial para o desenvolvimento de competências do século XXI. A habilidade de decompor problemas complexos é uma competência que transcende a sala de aula, sendo valiosa em áreas como engenharia, economia e ciência de dados. Dessa forma, ensinar programação no contexto matemático não apenas aprimora habilidades específicas, mas também contribui para a formação de indivíduos preparados para enfrentar desafios em diversas áreas.

Portanto, a integração do aprendizado de programação ao ensino de Matemática oferece uma oportunidade ímpar para desenvolver o raciocínio lógico, a criatividade e a autonomia dos estudantes, preparando-os para uma sociedade cada vez mais mediada pela tecnologia e pelo pensamento computacional.

3 O Python, uma linguagem mundial

3.1 O Python

Python é uma linguagem de programação de alto nível, criada com o objetivo de ser simples, fácil de aprender e eficiente.

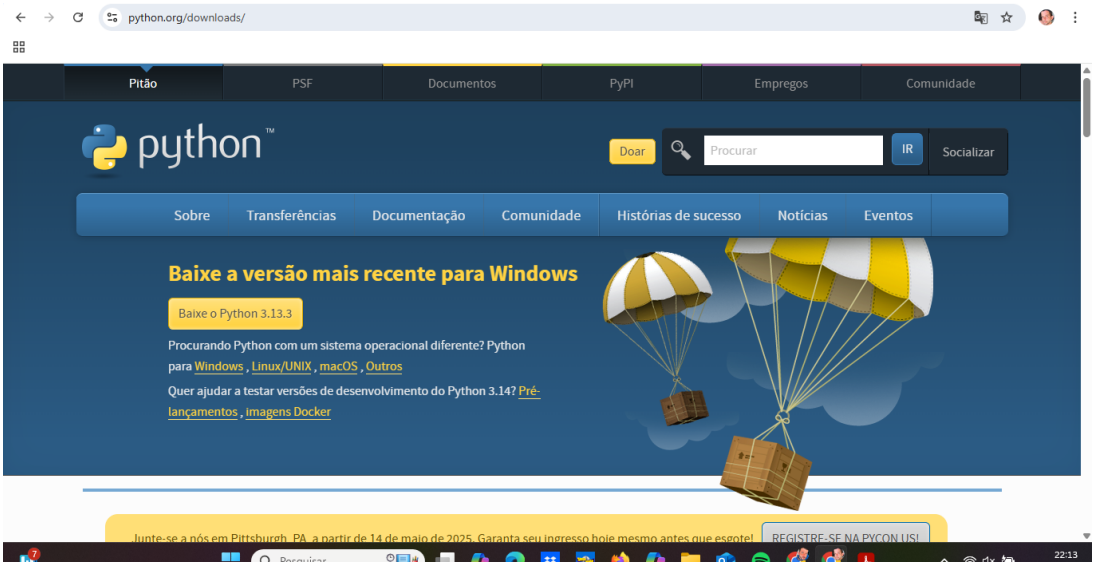
Segundo Costa (2023) Python, criada em 1991 pelo programador holandês Guido van Rossum, é atualmente uma das linguagens de programação mais populares do mundo, destacando-se por sua facilidade de aprendizado e uso intuitivo.

Ainda conforme Costa (2023), o nome Python tem origem no programa de televisão *Monty Python's Flying Circus*, uma escolha de Guido van Rossum, que buscava um título curto, único e divertido para a linguagem. Após o lançamento, Python rapidamente ganhou popularidade, especialmente entre profissionais que precisavam de uma ferramenta prática para automação e criação de *scripts*. Hoje, a linguagem está consolidada como uma das mais utilizadas em diversas áreas tecnológicas, incluindo desenvolvimento *web*, ciência de dados, inteligência artificial e aprendizado de máquina.

Os sistemas operacionais Linux e MacOS já possuem instalado o interpretador Python, que faz a interlocução entre o programador e o computador. No *Windows* é necessário realizar a instalação através do endereço eletrônico https://www.python.org.

A Figura 1 apresenta a página oficial de *download* do Python, onde os usuários podem obter a versão mais recente compatível com seus sistemas operacionais.

Figura 1 – Página de Download



Fonte: (FOUNDATION, 2024).

Após a instalação, é possível utilizá-lo por meio de uma IDE (*Integrated Development Environment*) para facilitar o desenvolvimento e a execução de programas.

Uma IDE (*Integrated Development Environment* ou Ambiente de Desenvolvimento Integrado) é um *software* que reúne diversas ferramentas úteis para programadores, com o objetivo de facilitar a criação, edição, execução e depuração de códigos.

Conforme a Python Software Foundation (2024):

> O Python é uma linguagem de programação interpretada, interativa e orientada a objetos. Ele incorpora módulos, exceções, tipagem dinâmica, tipos de dados dinâmicos de alto nível e classes. Há suporte a vários paradigmas de programação além da programação orientada a objetos, como a programação procedural e funcional. O Python oferece ao desenvolvedor um poder notável aliado a uma sintaxe simples e clara. Possui interfaces para muitas chamadas e bibliotecas do sistema, bem como para vários sistemas de janelas, e é extensível por meio de linguagens como C ou C++. Também é utilizado como linguagem de extensão para aplicativos que precisam de uma interface programável. Finalmente, o Python é portátil: pode ser executado em várias variantes do Unix, incluindo Linux e Mac, além do Windows.

Uma das melhores características do Python é sua flexibilidade. Sua linguagem suporta muitos conceitos de programação segundo sua plataforma, incluindo programação orientada a objetos, condicional e funcional. Essa compatibilidade permite que os programadores escolham o estilo de programação mais apropriado para uma determinada tarefa. Além disso, Python possui um poderoso sistema de tipo e gerenciamento automático de memória. Python também é conhecido por sua rica biblioteca, que fornece uma variedade de ferramentas para uma variedade de tarefas, muito conhecido como "bateria compartilhada" do Python destinada a fornecer uma base completa para desenvolvimento de projetos, eliminando a necessidade de procurar bibliotecas adicionais para tarefas comuns.

A popularidade do Python aumentou intensamente ao longo dos anos, e ele está regularmente classificado entre os melhores do mercado no ramo da programação, reconhecido por sua simplicidade, versatilidade e grande comunidade de desenvolvedores. Python é usado em vários projetos e em diferentes setores, desde desenvolvimento *web*, automação e inteligência artificial.

Segundo Conforme: Python Software Foundation (2024, apud Vezjak):

> Python é a principal linguagem de programação que usamos em nossa escola para ensinar aos alunos conceitos básicos de programação e algoritmos. Nossos alunos adoram Python - com Python eles podem criar seus próprios aplicativos, *sites*, questionários, resolver problemas diferentes e muito mais. Os alunos usam diferentes módulos como Pygame, Matplotlib, Numpy, Tkinter etc. para transformar suas ideias em realidade e desenvolver habilidades como pensamento computacional, criatividade e resolução de problemas.

A acessibilidade e simplicidade da sintaxe do Python, juntamente com sua grande biblioteca padrão e forte comunidade de desenvolvedores, tornam-no uma linguagem de programação ideal para iniciantes. A linguagem fornece uma base sólida para explorar o mundo da programação de maneira incrível e até mesmo divertida se usada em sala de aula. É por esta razão que escolheu-se estas linguagens de programação na nossa proposta de ensino.

Além da instalação tradicional do Python em computadores pessoais, diversas plataformas *online* têm se destacado por oferecer ambientes interativos para a prática de programação. Essa ferramenta permite que estudantes e professores escrevam, executem e testem códigos em Python de forma rápida e prática, sem a necessidade de instalar qualquer *software* localmente.

Embora ofereça funcionalidades semelhantes às de um ambiente de desenvolvimento, tais compiladores não são considerados uma IDE completa, pois não possuem recursos avançados como depuração integrada, gerenciamento de múltiplos arquivos ou suporte a projetos extensos. No entanto, por sua praticidade, acessibilidade e foco em atividades introdutórias.

3.2 Python como ferramenta educacional

A implementação da linguagem Python no ambiente educacional alinha-se à tendência global de incorporar a programação aos currículos escolares, como propõem diversas diretrizes educacionais internacionais e nacionais. No Brasil, a Base Nacional Comum Curricular (BNCC) enfatiza a relevância das Tecnologias da Informação e Comunicação (TICs) na educação básica, apontando a programação como uma competência fundamental para o desenvolvimento do raciocínio lógico, da criatividade e da capacidade de resolver problemas complexos.

Nesse cenário, Python se destaca como uma ferramenta poderosa devido ao seu ecossistema robusto, à sua sintaxe acessível e à sua versatilidade, o que a torna ideal para contextos educativos, especialmente na iniciação científica e na aplicação de conhecimentos matemáticos em situações práticas. Assim, Python cumpre um papel pedagógico estratégico ao permitir que conceitos abstratos sejam explorados de forma interativa, promovendo uma aprendizagem significativa e contextualizada.

Dentro desse mesmo escopo, o currículo do estado de Pernambuco representa um exemplo concreto da adoção de Python no contexto da educação básica. De acordo com a Secretaria de Educação de Pernambuco (2021), o documento curricular em questão foi elaborado por meio de uma colaboração entre o Estado e a União dos Dirigentes Municipais da Educação (UNDIME), contando com um amplo debate envolvendo professores das redes estadual e municipal de ensino médio, além da contribuição de docentes de instituições

públicas de ensino superior do Estado, do Conselho Estadual de Educação, do Sindicato dos Profissionais da Educação de Pernambuco, entre outras entidades representativas.

> Processos Criativos (EMIFMAT06PE): Propor e testar soluções éticas, estéticas, criativas e inovadoras para problemas reais, considerando a lógica de programação Python aplicada a partir dos conhecimentos matemáticos associados ao domínio de operações e relações matemáticas simbólicas e formais.
>
> Mediação e Intervenção Sociocultural (EMIFMAT09PE): Propor e testar estratégias de mediação e intervenção para resolver problemas de natureza sociocultural e de natureza ambiental relacionados à lógica de programação Python aplicada a partir de conhecimentos matemáticos. (SECRETARIA DE EDUCAÇÃO DE PERNAMBUCO, 2021, p.31).

A linguagem Python, cada vez mais utilizada em ambientes de aprendizagem, oferece formas pelas quais os alunos podem aplicar conceitos matemáticos em situações reais e significativas. Essa linguagem possui a capacidade de traduzir expressões abstratas de matemática para representações visuais, o que ajuda a compreender teorias complexas através de manipulações interativas e experimentos em primeira mão.

Ao integrar a programação no ensino de matemática, os alunos são encorajados a desenvolver habilidades essenciais para o século XXI, tais como o raciocínio computacional, a solução de problemas algorítmicos e o pensamento crítico. A capacidade de codificar algoritmos para resolver problemas matemáticos não só ajuda a aprofundar o entendimento dos conceitos fundamentais. Esta abordagem é um reflexo das recomendações da BNCC, que enfatiza a importância de integrar novas tecnologias e estratégias de ensino que promovam o aprendizado ativo e contextualizado (BRASIL, 2017).

3.3 Programiz: Utilizando o compilador *online*

A plataforma Programiz configura-se como um ambiente educacional digital voltado ao ensino de múltiplas linguagens de programação, destacando-se pela oferta de recursos interativos que atendem tanto a iniciantes quanto a usuários mais experientes. Com especial ênfase nas linguagens Python, C, C++, Java e Kotlin, a ferramenta foi concebida com a proposta de tornar o aprendizado da programação acessível a todos, independentemente de sua infraestrutura tecnológica.

Entre seus principais diferenciais está o compilador *online*, que permite ao usuário escrever, compilar e executar códigos diretamente em seu navegador, eliminando a necessidade de instalação de ambientes locais ou configurações técnicas mais complexas. Tal característica torna o Programiz especialmente atrativo para contextos educacionais que enfrentam restrições de acesso a recursos computacionais mais avançados, uma vez que é possível utilizar a plataforma em qualquer dispositivo conectado à internet.

O uso desse tipo de recurso digital, sobretudo no campo do ensino da matemática, representa uma oportunidade de inserir os discentes em experiências de aprendizagem mais interativas, motivadoras e alinhadas às competências tecnológicas exigidas no século XXI. Além disso, permite que se estabeleça uma ponte entre a abstração matemática e sua aplicação prática, promovendo uma aprendizagem ativa e significativa mesmo em contextos educacionais marcados por limitações estruturais.

A Figura 2 ilustra a interface do ambiente *online* Programiz, destacando seu design intuitivo e a funcionalidade do compilador diretamente acessível no navegador.

Figura 2 – Interface do compilador *online* Programiz.

Fonte: Elaborado pelo autor

O compilador *online* disponibilizado pela plataforma Programiz revela-se particularmente eficaz no processo de iniciação ao ensino de programação. Seu ambiente interativo e de fácil manuseio oferece aos aprendizes a possibilidade de desenvolver, testar e refinar códigos em um espaço virtual seguro, sem as barreiras técnicas geralmente impostas pela configuração de ambientes locais de desenvolvimento. Essa característica torna a ferramenta amplamente utilizada por estudantes, professores e entusiastas da programação que buscam métodos acessíveis, ágeis e didáticos para aprender e ensinar lógica computacional.

A execução dos códigos se dá por meio de um comando direto, representado pela tecla "*Run*", a qual aciona imediatamente a execução do script elaborado, permitindo a visualização instantânea dos resultados na área de saída. Essa funcionalidade estimula uma aprendizagem ativa e exploratória, permitindo que os usuários experimentem, testem hipóteses e corrijam erros de forma dinâmica e contínua.

Dentre as principais características do compilador *online* do Programiz, destacam-se:

Interface Intuitiva: O ambiente apresenta um design simplificado e de navegação fluida, facilitando o acesso de iniciantes ao universo da programação sem exigir

conhecimentos prévios em ambientes computacionais complexos.

Suporte Multilíngue: A plataforma é compatível com diversas linguagens de programação amplamente utilizadas no cenário atual, como Python, Java, C, C++ e Kotlin, permitindo ao usuário experimentar diferentes estruturas sintáticas e estilos de codificação.

Execução em Tempo Real: Os códigos podem ser executados instantaneamente, com os respectivos resultados exibidos imediatamente em uma janela de saída acoplada, o que contribui para uma compreensão mais clara do funcionamento dos algoritmos.

Compatibilidade Ampla com Dispositivos: Por ser acessado diretamente via navegador, o Programiz independe de instalações locais, podendo ser utilizado em qualquer dispositivo conectado à internet, seja ele computador, *chromebook*, *tablet*, *smartphone*.

Mensagens de *Feedback* para Correções: Em casos de erros de sintaxe ou lógica, o compilador fornece mensagens explicativas que auxiliam o aprendiz na identificação e correção de falhas, promovendo o desenvolvimento do pensamento lógico e do raciocínio computacional.

É importante salientar, no entanto, que, apesar de suas múltiplas vantagens, o compilador *online* do Programiz apresenta limitações em termos de robustez, sendo inadequado para projetos extensos ou que exijam múltiplos arquivos e recursos avançados de depuração. Ainda assim, sua funcionalidade é plenamente satisfatória para o ensino de fundamentos da programação, resolução de exercícios elementares e desenvolvimento de pequenos projetos voltados ao aprendizado.

4 Embarcando na Jornada de Aprendizado Python

4.1 Preparação do ambiente de programação em linguagem Python usando o Programiz

A preparação do ambiente de programação é uma etapa crucial para a implementação deste projeto educacional, que integra o ensino de matemática com a prática de programação em Python. Considerando a acessibilidade e a simplicidade necessárias para atingir o público-alvo do ensino fundamental, avalia-se inicialmente o uso do Google Collaboratory (Google Colab), uma ferramenta gratuita que permite a execução de códigos Python diretamente no navegador, utilizando o processamento em nuvem. No entanto, os e-mails institucionais dos alunos das escolas estaduais de Mato Grosso não possuem permissão de acesso à plataforma, o que inviabilizou sua adoção.

4.1.1 Bibliotecas

As bibliotecas em Python são coleções de módulos que reúnem funções, classes e objetos previamente definidos, com o objetivo de oferecer funcionalidades específicas e otimizar as tarefas dos desenvolvedores. Elas funcionam como ferramentas prontas para uso, que eliminam a necessidade de os programadores criarem código do zero para tarefas comuns ou complexas, resultando em maior eficiência, produtividade e organização no desenvolvimento de aplicações.

Essas bibliotecas podem oferecer funcionalidades diversificadas, desde operações matemáticas e manipulação de dados até geração de gráficos, interação com redes e bancos de dados, ou atividades especializadas, como aprendizado de máquina, processamento de imagens e desenvolvimento de interfaces gráficas. Isso faz do Python uma linguagem extremamente versátil e popular, permitindo que desenvolvedores de diferentes níveis de experiência acessem soluções poderosas com poucas linhas de código.

O Python conta com uma biblioteca padrão robusta, que é instalada junto com a linguagem e fornece um amplo conjunto de ferramentas para tarefas básicas e avançadas. Além disso, a comunidade de desenvolvedores criou inúmeras bibliotecas de terceiros que podem ser instaladas e integradas aos projetos, expandindo significativamente o alcance das aplicações que podem ser desenvolvidas com Python.

Em síntese, as bibliotecas em Python promovem a reutilização de código, melhoram a legibilidade e simplificam a criação de projetos complexos, oferecendo soluções específicas

para praticamente qualquer aplicação imaginável.

1. **math**
   Contém a implementação de funções utilizadas em cálculos matemáticos mais avançados, como raiz quadrada, potência, seno, cosseno, entre outros.

2. **random**
   Fornece funções responsáveis pela geração de números aleatórios, sendo muito útil em jogos, simulações e sorteios.

3. **statistics**
   Fornece funções que calculam média, mediana, moda e outros conceitos estatísticos básicos.

4. **numpy**
   Fornece funções que realizam operações matemáticas com vetores e matrizes. Aplicada em áreas como física, matemática e engenharia.

5. **matplotlib**
   Disponibiliza funções que permitem a criação de gráficos e visualização dados de forma prática, ideal para complementar análises numéricas.

6. **pandas**
   Muito usada em análise de dados, manipulação de tabelas e leitura de planilhas em formatos como .csv e .xlsx.

7. **fractions**
   Disponibiliza funções que permitem trabalhar com números racionais, convertendo valores decimais em frações exatas.
   Exemplo: Fraction(0.5) retorna `1/2`.

Todas essas bibliotecas fazem parte da **Biblioteca Padrão do Python**. A documentação oficial pode ser acessada em: https://docs.python.org/pt-br/3/library/.

4.2 Entrada e saída de dados

Nesta seção, será apresentado o detalhamento dos códigos desenvolvidos utilizando a linguagem de programação Python. O objetivo é descrever de forma clara e didática a lógica aplicada, os comandos utilizados e a função de cada trecho de código no contexto da proposta deste trabalho.

4.2.1 Tipos de Dados Padrão no Python

No contexto da linguagem Python, diversos são os tipos de dados nativos que constituem a base para o desenvolvimento de estruturas computacionais e algoritmos. Entre eles, destacam-se:

- **Inteiro** (*int*): Trata-se de um tipo de dado composto por números inteiros, ou seja, sem casas decimais. Esse tipo é amplamente utilizado para representar quantidades exatas, como contadores, índices e resultados de operações inteiras.
- **Ponto flutuante ou decimal** (*float*): Refere-se a números reais com casas decimais. Esse tipo é fundamental para expressar medidas contínuas e valores aproximados em cálculos matemáticos, físicos ou financeiros.
- **Complexo** (*complex*): Tipo de dado que representa números complexos, formados por uma parte real e uma parte imaginária. É especialmente relevante em aplicações de engenharia e matemática, como aquelas estudadas no ensino médio, envolvendo a unidade imaginária i.
- **Cadeia de caracteres** (*str*): Denomina-se *string* a sequência de caracteres organizada em uma ordem específica. Utiliza-se esse tipo para representar palavras, expressões linguísticas, textos e outros dados simbólicos.
- **Booleano** (*bool*): Tipo de dado lógico, caracterizado por assumir apenas dois valores possíveis: *True* (verdadeiro) ou *False* (falso). Ele é essencial para expressar condições e decisões dentro de estruturas de controle como *if*, *while* e *for*.
- **Lista** (*list*): Estrutura de dados bastante versátil, utilizada para armazenar coleções ordenadas de elementos. As *lists* são mutáveis, permitindo adição, remoção e alteração de itens após sua criação.
- **Tupla** (*tuple*): Semelhante à *list*, a *tuple* também armazena uma sequência de elementos. A principal distinção está na sua imutabilidade — uma vez definida, seus valores não podem ser modificados. Define-se por meio de parênteses e separação por vírgulas.
- **Dicionário** (*dict*): Estrutura de dados baseada em pares de chave e valor. Cada chave associa-se diretamente a um valor, permitindo acesso rápido e eficiente às informações. Sua notação em Python utiliza chaves ({ }) com os pares chave:valor separados por dois pontos.

4.2.2 Funções: Entradas e saídas

As funções são blocos de código que realizam tarefas específicas e podem ser reutilizadas em diferentes partes de um programa, o que contribui para tornar o código mais organizado, legível e eficiente.

A linguagem Python conta com diversas funções embutidas (*built-in*), que são implementadas nativamente na linguagem e estão prontas para uso. Essas funções facilitam tanto operações básicas quanto avançadas, abrangendo desde a manipulação de dados até cálculos matemáticos, conversões de tipo e manipulação de textos (*strings*).

Nesta seção, aborda-se algumas das funções *built-in* mais utilizadas em atividades introdutórias:

***print*():** Exibe valores na tela.

***input*():** Lê dados digitados pelo usuário.

***len*():** Retorna o número de caracteres em uma *string* (que será explicado detalhadamente logo a seguir).

4.2.2.1 Aprendendo a executar *print*()

Este é geralmente o primeiro comando aprendido ao iniciar na programação, pois permite uma interação simples e direta entre o código e o usuário.

A Figura 3 mostra o uso do comando *print*() em Python, utilizado para exibir mensagens na tela. No exemplo, a frase " O Python é legal " é impressa no na tela de *output*.

Figura 3 – Aprendendo a imprimir (*print*)

Fonte: Elaborado pelo autor

Na imagem da Figura 3, a parte precedida por # representa um comentário, utilizado

para anotações do programador e que não será processado pelo computador. Em seguida, utiliza-se a função *print()* e, entre aspas, passa-se uma frase. Quando esse programa é executado clicando na tecla *Run*, ele exibe a frase fornecida como argumento para a função. À direita, observa-se a saída (*output*), que corresponde à frase entre aspas: **"O Python é legal"**.

Pode-se agora modificar essa frase para explorar a ferramenta, utilizando, por exemplo, aspas simples em vez de aspas duplas, entre outras variações possíveis.

A Figura 4 demonstra o uso da função *print()* com aspas simples para exibir uma nova mensagem:

'Estudar é bom, mas estudar matemática é ainda melhor.'

Essa variação demonstra que o Python aceita tanto aspas simples quanto duplas para definir *strings*, oferecendo maior flexibilidade na escrita de textos no código.

Figura 4 – Alterando a frase e a aspas simples

Fonte: Elaborado pelo autor

Utilizar aspas duplas ou simples em Python resulta no mesmo efeito: ambas imprimem o conteúdo da *string* da mesma forma. No exemplo, altera-se a frase para **"Estudar é bom, mas estudar matemática é ainda melhor"**, e ao clicar em ***"Run"***, o resultado é exibido no painel à direita da tela.

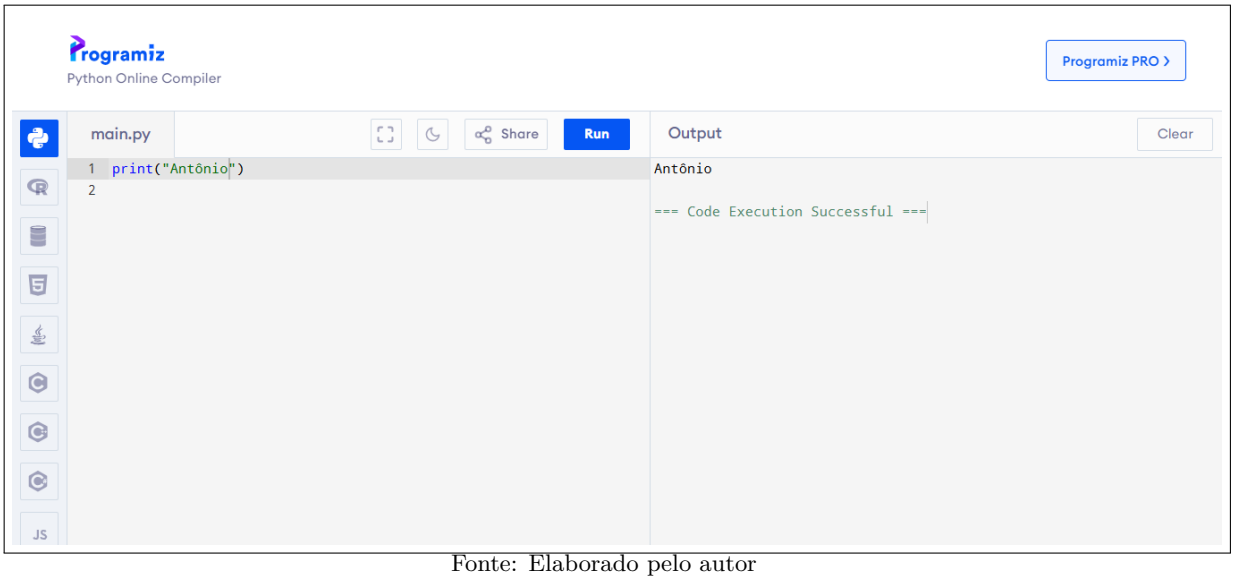
Em seguida, salva-se o nome **"Antônio"** em uma variável e o execute. Por se tratar de um texto (ou *string*), o valor deve ser colocado entre aspas, sejam elas simples ou duplas. Já no caso de valores numéricos, as aspas não são necessárias. Essa distinção representa uma das diferenças fundamentais entre os tipos de dados (ou classes de dados) em Python.

A Figura 5 apresenta como armazenar um valor em uma variável no Python. No exemplo, a variável `nome` recebe o valor **"Antônio"** e, em seguida, é impressa com o

comando *print*(nome).

Esse procedimento é essencial na programação, pois permite armazenar e reutilizar informações ao longo do código de forma eficiente e organizada.

Figura 5 – Imprimindo nome de pessoas



Fonte: Elaborado pelo autor

4.2.2.2 Criando uma variável

Ao iniciar seus estudos em programação na linguagem Python, o estudante depara-se com um dos conceitos mais fundamentais da ciência da computação: o conceito de variável. Nesse contexto, o professor explica que uma variável, em Python, é uma espécie de "caixa de armazenamento" capaz de guardar um valor que poderá ser usado, consultado ou alterado ao longo do programa.

De forma análoga ao que se observa na matemática, onde uma letra como x pode representar um número, em Python uma variável também representa um valor — porém, com a importante vantagem de poder armazenar diversos tipos de dados, como números inteiros, números decimais, textos dentre outros.

A criação de uma variável em Python é simples e direta. Basta escolher um nome significativo e atribuir um valor utilizando o sinal de igualdade. As variáveis tornam-se, portanto, elementos centrais na construção de programas que interagem com o usuário, realizam cálculos, armazenam informações e organizam processos lógicos.

Na Figura 6, a variável **idade** está sendo utilizada para armazenar o valor **15**, que representa a idade de um estudante.

Figura 6 – Criando uma variável

Fonte: Elaborado pelo autor

4.2.2.3 Aprendendo *input*()

Ao chegar neste ponto do estudo da linguagem Python, o estudante é convidado a dar um passo essencial na construção do raciocínio computacional: permitir que o usuário interaja com o programa. Para isso, o estudante deverá aprender a utilizar corretamente a função *input()* — um dos primeiros e mais importantes mecanismos de entrada de dados em Python.

O professor orienta que, ao utilizar o *input()*, o estudante compreenda não apenas a forma de escrevê-lo no código, mas, sobretudo, o seu propósito: **solicitar ao usuário que digite uma informação**, que será então capturada pelo programa e armazenada em uma variável.

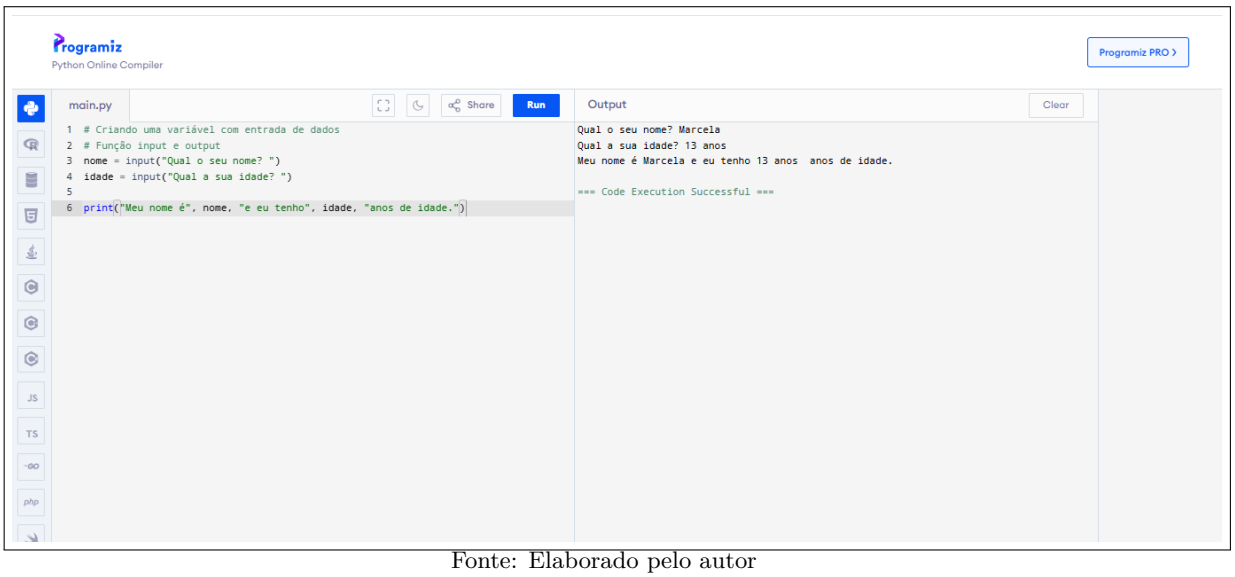
Nesta etapa da proposta didática, conforme ilustrado na Figura 7, propõe-se que os estudantes, de forma orientada e interativa, explorem os comandos fundamentais da linguagem Python por meio da plataforma Programiz. Espera-se que, nesse momento, cada aluno seja capaz de inserir uma variável e utilizar o comando *input()* para realizar a leitura de uma informação fornecida pelo usuário, preferencialmente por meio de uma pergunta simples e contextualizada. Essa tipo de atividade tem por objetivo familiarizar o estudante com a estrutura básica da linguagem, ao mesmo tempo em que estimula a compreensão do funcionamento de entradas de dados em um ambiente computacional.

Figura 7 – Aprendendo *input*()

Fonte: Elaborado pelo autor

Na Figura 8 cria-se duas variáveis que recebem valores por meio da função *input()*. Esses valores são inseridos pelo usuário no momento da execução do programa. Em seguida, executa os dados armazenados — no caso, o nome **"Marcela"** e a idade **13**.

Figura 8 – Aprendendo o input e output

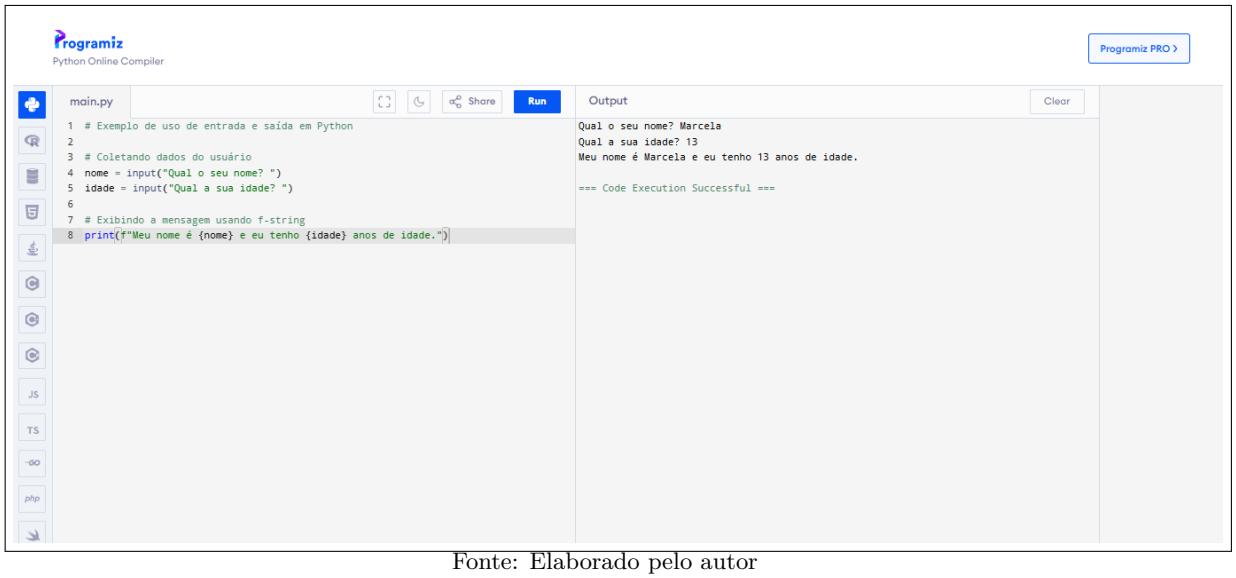


Fonte: Elaborado pelo autor

Basta iniciar a *string* com um f antes das aspas e colocar o nome das variáveis entre chaves `{}`, como mostrado na Figura 9. Isso garante uma melhor legibilidade do código e facilita a formatação da saída.

Para garantir compatibilidade entre variáveis do tipo numérico e *string* ao combinar os dados em uma única frase dentro da função *print()*, utiliza-se as chamadas *f-strings*. Esse recurso permite inserir variáveis diretamente dentro de uma *string*, com clareza e simplicidade.

Figura 9 – Chamadas *f-strings*



Fonte: Elaborado pelo autor

Neste momento da proposta didática, sugere-se a realização de uma atividade de fixação prática, na qual os estudantes serão desafiados a criar, individualmente ou em duplas, um programa simples utilizando a plataforma Programiz. O programa deverá solicitar ao usuário a entrada de quatro informações: nome, idade, esporte preferido e disciplina favorita. Em seguida, o código deverá exibir uma frase de apresentação na primeira pessoa do singular, combinando esses dados em uma mensagem personalizada.

A Figura 10 ilustra um exemplo de exercício que utiliza comandos de entrada e saída, consolidando as noções iniciais de Python. No exemplo, o programa coleta informações básicas e organiza os dados em variáveis, exibindo-os por meio de uma frase formatada e personalizada, proporcionando ao estudante uma experiência prática e contextualizada na utilização de comandos fundamentais da linguagem.

Figura 10 – Atividade de fixação.

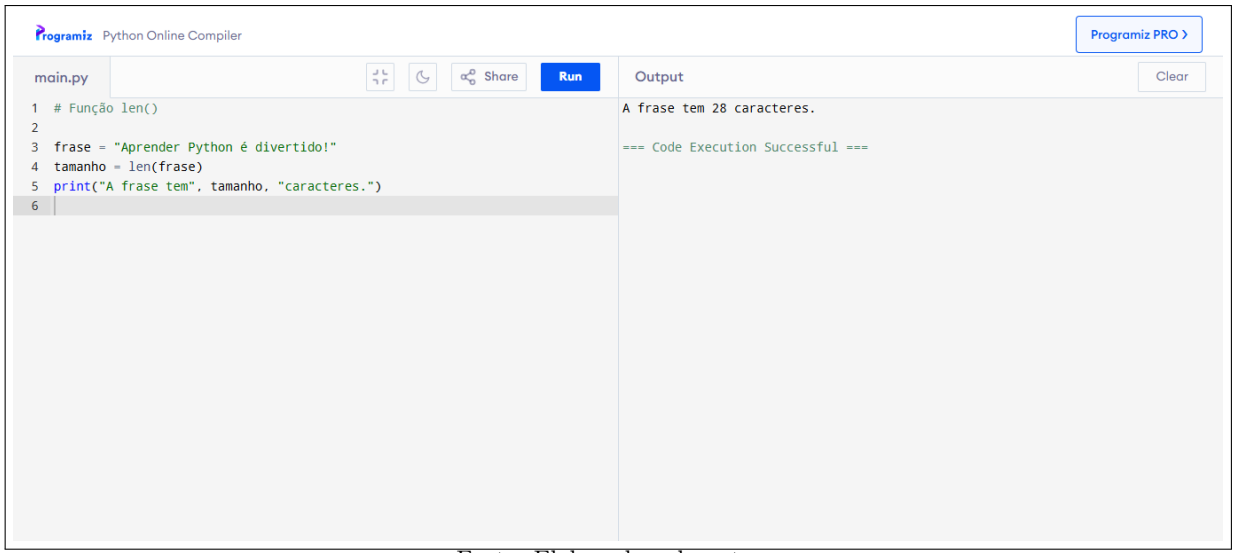
Fonte: Elaborado pelo autor

4.2.2.4 Aprendendo *len()*

A função *len()* em Python é uma função nativa da linguagem, utilizada para determinar o tamanho de uma sequência de dados, como uma *string*, uma lista, uma tupla ou outros tipos iteráveis. De modo geral, ela retorna um número inteiro que representa quantos elementos estão presentes naquela estrutura.

Na Figura 11, ao aplicar *len()* a uma *string*, ela retorna o número de caracteres, incluindo letras, espaços e sinais de pontuação. Já ao aplicá-la a uma lista, retorna quantos itens essa lista contém.

Figura 11 – Aprendendo *len*()

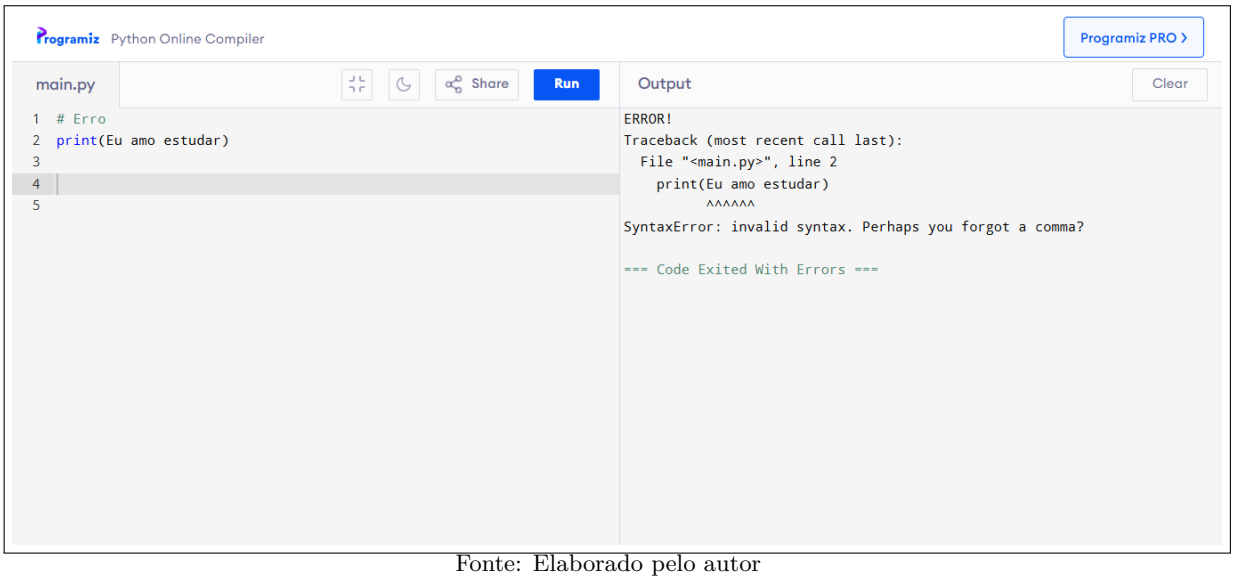


Fonte: Elaborado pelo autor

4.3 Erro nos comando em Python

A Figura 12 apresenta uma execução mal sucedida no ambiente Programiz, decorrente de um erro de sintaxe na linguagem Python. O código tem como objetivo realizar a leitura. No entanto, o erro ocorre na linha 2, devido ao uso incorreto da vírgula, o que resulta em um *SyntaxError*.

Figura 12 – Erro na codificação



Fonte: Elaborado pelo autor

Caso o estudante tenha errado algum comando ele terá essa mensagem:

```
ERROR!
Traceback (most recent call last):
  File "<main.py>", line 2
    print(Eu amo estudar)
          ^^^^^^
SyntaxError: invalid syntax. Perhaps you forgot a comma?

=== Code Exited With Errors ===
```

Essa situação é extremamente rica do ponto de vista didático, pois evidencia a importância da precisão na escrita de código e promove a aprendizagem por meio da análise de erros. Além disso, reforça a ideia de que a matemática e a programação compartilham o rigor na linguagem e a necessidade de atenção aos detalhes na construção lógica das soluções.

4.4 Tipos de Operadores

Em Python, cada valor pertence a um tipo de dado específico. Existem diversos tipos de dados, porém proposta dispõe-se os mais comuns:

4.4.1 Operadores aritméticas básicos

Pode-se considerar que em Python, uma expressão é uma combinação de valores e operadores que, ao serem avaliados, resultam em um único valor. Os operadores matemáticos, como (+, -, *, /, //, %, **, 1/x) permitem realizar operações aritméticas, seguindo a ordem de precedência usual da matemática. Exemplo: 2 + 3 * 6 é avaliado como 20, pois a multiplicação tem precedência sobre a adição.

A Tabela 1 apresentada a seguir, tem por objetivo elencar e descrever os principais símbolos matemáticos utilizados na linguagem de programação Python para a realização de operações aritméticas básicas e avançadas.

Tabela 1 – Operadores Aritméticos em Python

Operação	**Símbolo**
Soma	$+$
Subtração	$-$
Multiplicação	$*$
Divisão	$/$
Divisão Inteira	$//$
Resto da divisão (mod)	$\%$
Potenciação	$**$
Radiciação	$**(1/x)$
Arredondamentos	$round$

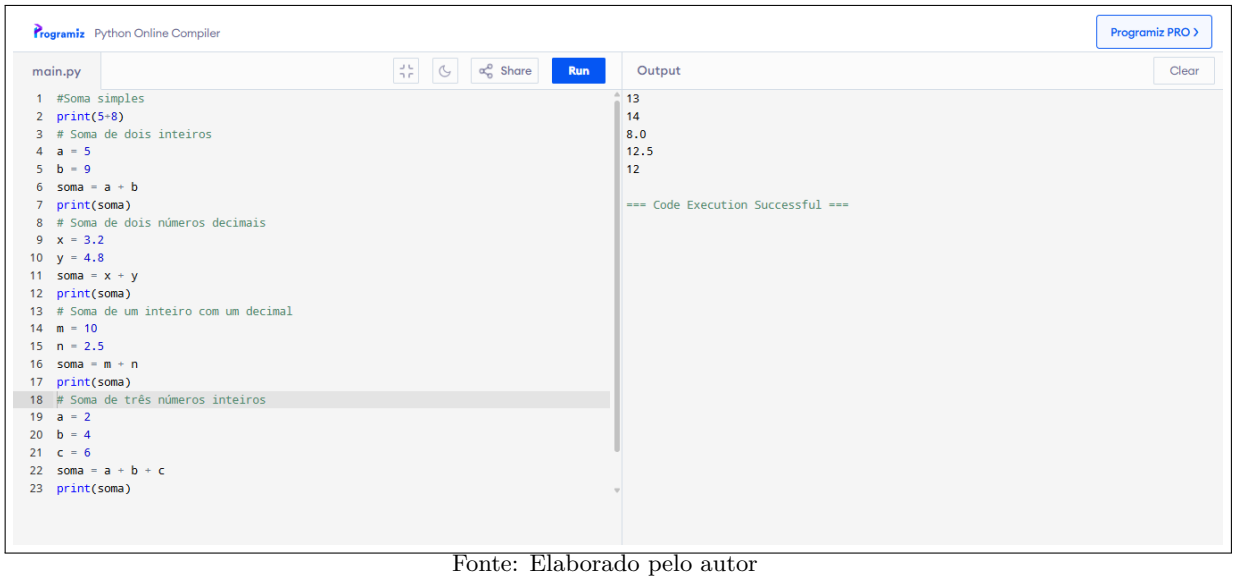
Fonte: Elaborado pelo autor

4.4.1.1 **Adição (+):**

A adição em Python é feita com o operador +, que soma dois números e retorna o resultado.

A Figura 13 demonstra diferentes formas de realizar somas em Python.

Figura 13 – Adição



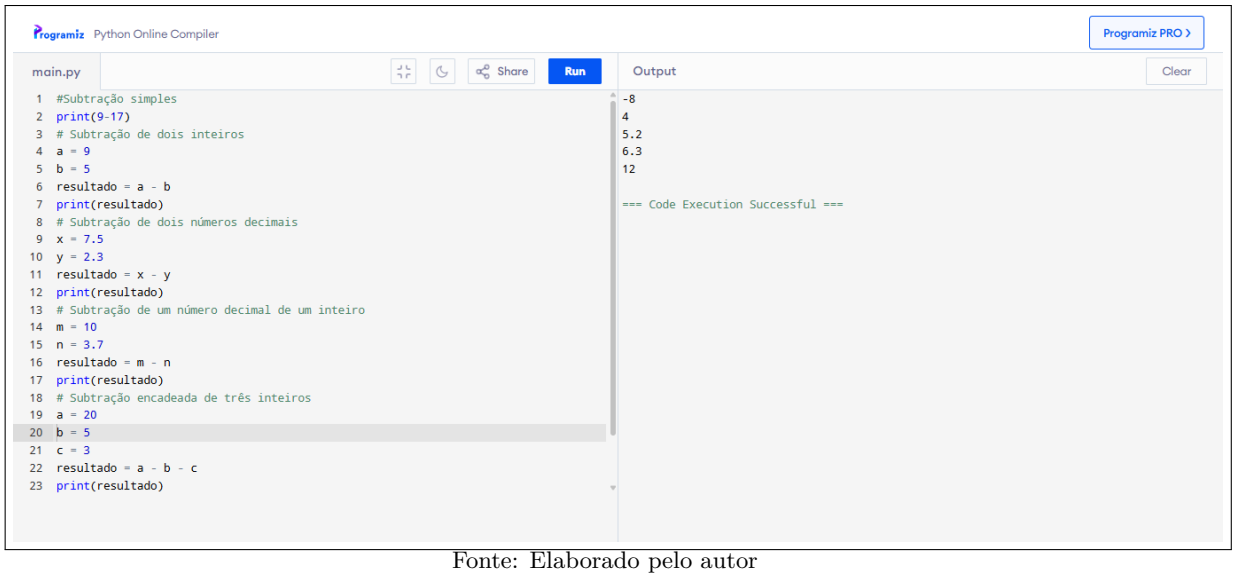
Fonte: Elaborado pelo autor

4.4.1.2 **Subtração (-):**

A subtração em Python usa o operador -, que subtrai um número de outro.

A Figura 14 apresenta exemplos práticos de como realizar subtrações utilizando a linguagem Python. O primeiro exemplo mostra a operação direta com dois números, enquanto o segundo utiliza variáveis para armazenar valores e realizar a subtração entre elas.

Figura 14 – Subtração



Fonte: Elaborado pelo autor

4.4.1.3 **Multiplicação (*):**

A multiplicação é feita com o operador *, que multiplica dois números.

A Figura 15 ilustra como utilizar o operador de multiplicação em Python por meio de exemplos simples e didáticos. No primeiro exemplo, observa-se a multiplicação direta entre dois números. Já no segundo, a operação é realizada entre valores previamente armazenados em variáveis.

Figura 15 – Multiplicação

Fonte: Elaborado pelo autor

4.4.1.4 Divisão simple *float* (/), Divisão Inteira (//) e Módulo (%):

A divisão é feita com o operador /, que divide um número por outro. O resultado sempre será um número decimal. A divisão inteira retorna a parte inteira da divisão e descarta o restante. O operador de módulo retorna o resto de uma divisão.

A Figura 16 apresenta três operações fundamentais no ensino de programação com Python: a divisão simples com ponto flutuante (/), a divisão inteira (//) e o operador módulo(%).

Figura 16 – Divisão, Divisão inteira, Resto da divisão

Fonte: Elaborado pelo autor

Em Python, as operações de divisão, divisão inteira e resto de divisão desempenham um papel fundamental nas operações matemáticas e de manipulação de números. Cada uma dessas operações tem características específicas que definem como os números são processados e os resultados são retornados.

A operação de divisão em Python é representada pelo operador de barra `/` e resulta sempre em um número do tipo *float*, mesmo que os números envolvidos sejam inteiros. Isso significa que a divisão de dois inteiros retorna um valor com casas decimais, fornecendo o resultado exato da operação. Esse comportamento distingue Python de outras linguagens em que a divisão de inteiros pode resultar em outro número inteiro, truncando qualquer parte decimal.

A divisão inteira em Python é realizada com o operador `//`. Diferente da divisão comum, ela retorna a parte inteira do resultado, descartando qualquer valor fracionário ou decimal. Essa operação é útil quando deseja-se dividir dois números e obter apenas o quociente inteiro, sem considerar a parte decimal. É amplamente empregada em situações que requerem cálculos discretos, como contagem de itens em lotes ou ao determinar o número de vezes que algo pode ser completamente dividido por outro número.

O resto da divisão é encontrado utilizando o operador de porcentagem `%` e é conhecido como operação de módulo. Ele retorna o valor restante de uma divisão inteira entre dois números.

4.4.1.5 **Exponenciação (**):**

A exponenciação em Python refere-se ao processo de elevar um número (a base) a um determinado expoente (ou potência). Esse recurso é amplamente utilizado em cálculos matemáticos e científicos que requerem o uso de potências e pode ser aplicado em diversos

contextos, como álgebra, computação científica, processamento de dados e geração de crescimento exponencial em algoritmos.

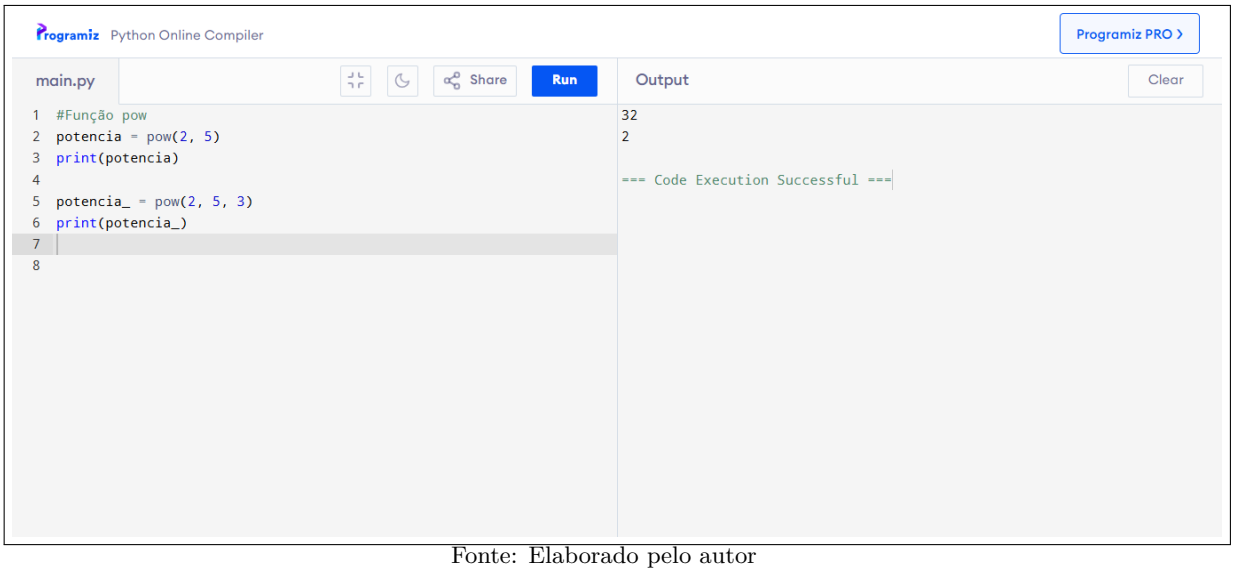
Exponenciação é a operação de elevar um número a uma potência. Na Figura 17 são demonstrados exemplos práticos utilizando essa operação, com a execução do código no Programiz, indicando operações com expoentes pelo operador **, permite elevar um número (base) a uma determinada potência (expoente).

Figura 17 – Exponenciação

Fonte: Elaborado pelo autor

Na Figura 18 apresenta-se a **função *pow*()** em Python que também é utilizada para calcular potências. Ela recebe dois ou três argumentos. Com dois argumentos, eleva o primeiro número (a base) ao segundo (o expoente). Com três argumentos, calcula a potência da base elevada ao expoente e depois aplica o módulo (resto da divisão) sobre o resultado. Essa função é versátil, aceita números inteiros e decimais, e também é útil em operações matemáticas mais avançadas, como as que envolvem aritmética modular.

Figura 18 – Exponenciação com função *pow*



Fonte: Elaborado pelo autor

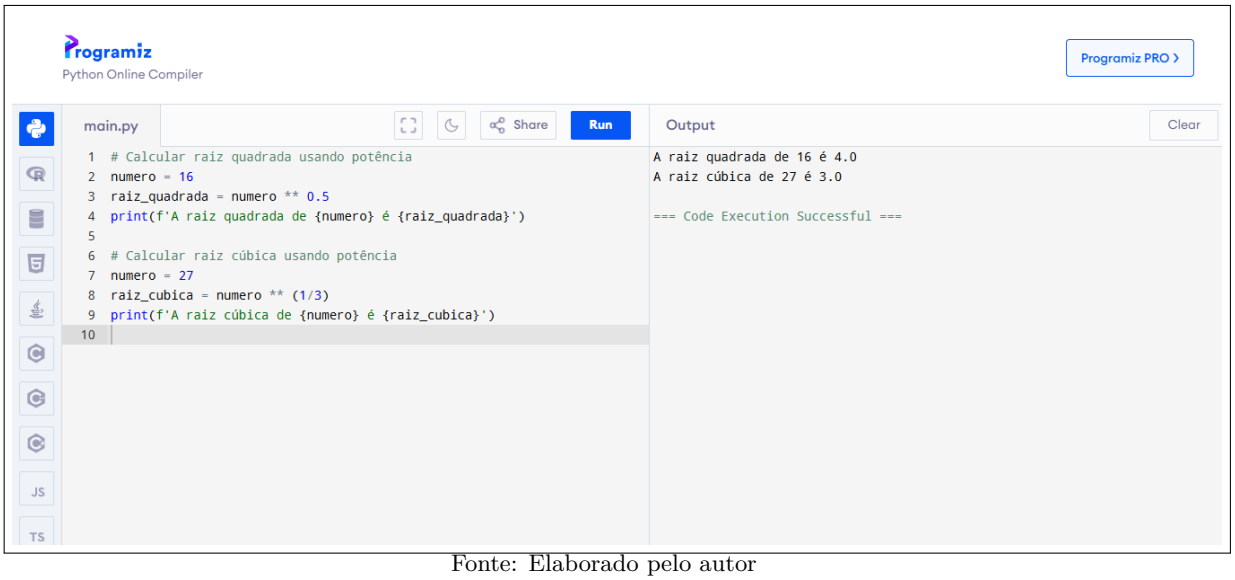
Quando usada com dois argumentos, retorna o valor da base elevada ao expoente, como *pow(2, 5)*, que resulta em 32. Com três argumentos, ela realiza a operação de potência modular, ou seja, calcula a potência e depois aplica o módulo, como em *pow(2, 5, 3)*, que equivale a (2^5) mod 3 e retorna 2.

4.4.1.6 Raízes(**1/x)

As raízes são exemplos de exponenciação com expoente fracionário.

A Figura 19 aborda o conceito de raízes em Python, demonstrando como calcular a raiz quadrada e a raiz cúbica de números utilizando o operador de exponenciação. O exemplo apresentado no código mostra que, para encontrar a raiz quadrada de um número, pode-se elevar a base à potência de 0.5. Da mesma forma, a raiz cúbica é obtida ao elevar a base à potência 1/3.

Figura 19 – Raízes

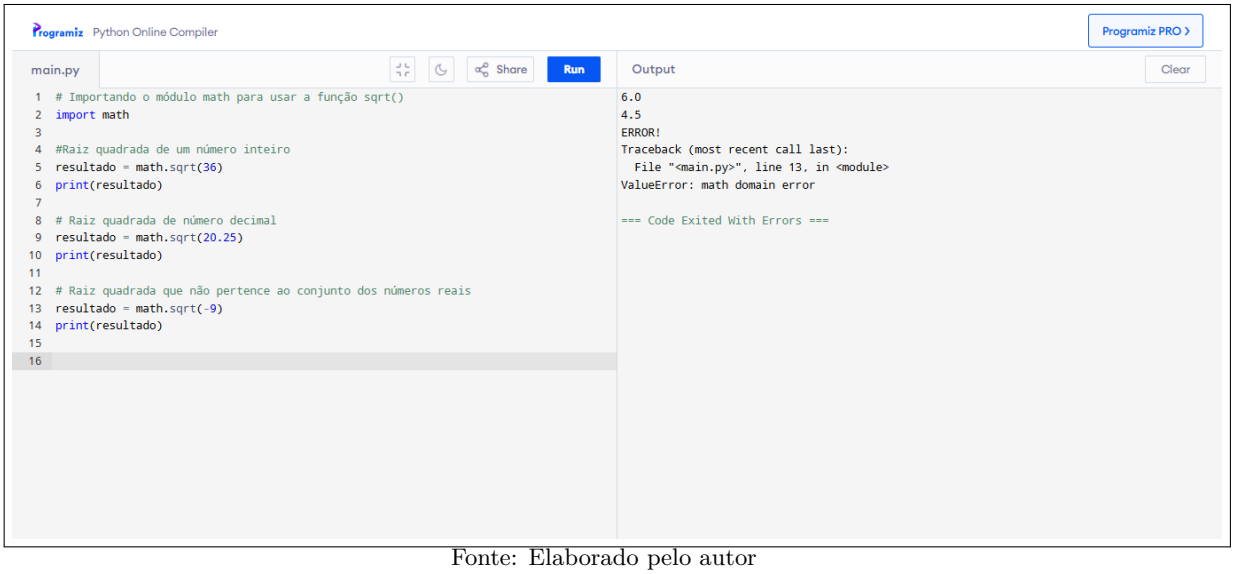


Fonte: Elaborado pelo autor

Esse comportamento permite resolver operações de radiciação de maneira direta, sem a necessidade de funções específicas para raízes quadradas ou cúbicas, simplificando expressões complexas de forma prática.

Na Figura 20, apresenta-se a função ***math.sqrt()*** é utilizada em Python para calcular a raiz quadrada de um número. Ela faz parte do biblioteca math, que precisa ser importado antes do uso. Essa função recebe como argumento um número (inteiro ou decimal) e retorna sua raiz quadrada com precisão decimal.

Figura 20 – Raízes usando *math.sqrt*



Fonte: Elaborado pelo autor

4.4.1.7 Ordem de Resolução

Uma **função** é um bloco de código que realiza uma tarefa específica, geralmente com base em valores de entrada (chamados **argumentos**) e que pode retornar um resultado. As funções ajudam a organizar, reutilizar e estruturar melhor o código.

Descrição da Função `operacao_matematica`:

- `def` é uma palavra-chave em Python usada para definir uma função.
- `operacao_matematica` é o nome da função. Você pode dar qualquer nome, mas deve ser significativo para indicar o que a função faz.
- Os parênteses `()` são usados para indicar que se trata de uma função. Dentro deles, você pode colocar argumentos que a função recebe. Neste caso, como não há nenhum argumento, os parênteses estão vazios.
- O sinal de dois pontos `:` indica o início do bloco de código pertencente à função. O que vier a seguir, com indentação (espaço recuado), será parte do "corpo" da função.

A Figura 21 explora o conceito de ordem de operação em Python, uma parte fundamental da matemática que determina como as expressões são avaliadas. O código apresentado define uma função `operacao_matematica` que realiza uma série de operações, incluindo potenciação e operações básicas como adição e subtração.

Figura 21 – Ordem de operação

Fonte: Elaborado pelo autor

Python segue uma ordem de resolução nas operações, semelhante à matemática. O exemplo a seguir ilustra como definir e usar a função `operacao_matematica` em Python:

- **Potenciação**: 4^2 é calculado como `4**2` em Python, resultando em 16.
- **Operações dentro dos parênteses**: Primeiramente, $16 - 6 = 10$.
- **Multiplicação**: $3 \times 10 = 30$.
- **Divisão**: $12/3 = 4$.
- **Soma e subtração**: $2 + 30 - 4 + 5 = 33$.

Na execução, o resultado final da expressão é calculado e exibido, com a saída indicando que o resultado é 33.0. Este exemplo não apenas ilustra como a ordem das operações afeta o resultado, mas também mostra a importância de usar parênteses para garantir que as operações sejam realizadas na sequência correta.

A Figura 22 explora a execução de expressões sem o uso de parênteses, chaves ou colchetes. O código demonstra como o Python avalia a expressão definida na função `operacao_matematica`, que realiza diversas operações aritméticas.

Figura 22 – Ordem de operação

Fonte: Elaborado pelo autor

Nesse caso = 2 + 3 * 4**2 - 6 - 12 / 3 + 5

= 2 + 3 * 16 - 6 - 4 + 5

= 2 + 48 - 6 - 4 + 5

= 45

As Potencias são resolvidas primeiro, seguido de Multiplicação e divisão que têm precedência sobre adição e subtração.

4.4.2 Operadores de comparação:

A Tabela 2 visa sistematizar os principais símbolos utilizados na linguagem Python para a realização de operações relacionais, isto é, aquelas que comparam dois valores e retornam um resultado booleano: *True* (verdadeiro) ou *False* (falso).

Tabela 2 – Operadores de Comparação em Python

Operação	**Símbolo**
Maior que	$>$
Menor que	$<$
Maior ou igual	$>=$
Menor ou igual	$<=$
Diferente	$!=$
Igualdade	$==$

Fonte: Elaborado pelo autor

A Figura 23 demonstra o uso de operadores comparativos, como ==, !=, >, <, >= e <=. Cada linha realiza uma comparação entre as variáveis x = 5 e y = 3, e o resultado é impresso no console, mostrando se a expressão é *True* ou *False*. À direita, observa-se o *output* correspondente a cada comparação, que confirma o funcionamento lógico dessas operações.Útil para ilustrar conceitos básicos de comparação e lógica *booleana* em Python.

Figura 23 – Operadores de comparação

Fonte: Elaborado pelo autor

4.4.3 Operadores de Atribuição

Operadores de atribuição são símbolos usados na programação para atribuir valores a variáveis e, em muitos casos, também realizar operações matemáticas ao mesmo tempo. Em Python, o operador de atribuição mais básico é o sinal de igual (=), que simplesmente atribui um valor a uma variável, como em x = 20.

A Figura 24 ilustra exemplos práticos do uso dos operadores de atribuição em Python, executados em um compilador *online*. O código demonstra diferentes formas de atribuir valores a uma variável utilizando operadores como =, +=, -=, *=, /=, **= e //=. À esquerda, o código realiza operações sucessivas sobre a variável x, e à direita, o *output* exibe o resultado após cada atribuição. A imagem serve como exemplo didático para mostrar como esses operadores modificam e atualizam o valor armazenado na variável, sendo um recurso fundamental na programação com Python.

Figura 24 – Operadores de Atribuição

Fonte: Elaborado pelo autor

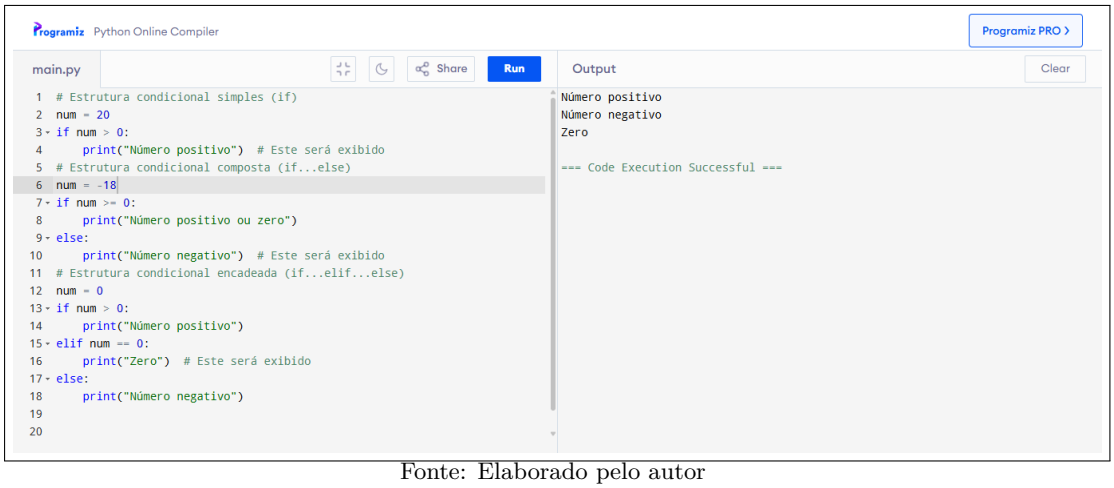
4.4.4 Estruturas Condicionais

As estruturas condicionais permitem que seus programas tomem decisões inteligentes, executando diferentes ações com base em condições específicas.

A estrutura condicional mais básica em Python é o *IF-THEN-ELSE*, que permite executar um bloco de código somente se uma condição for verdadeira. A condição é uma expressão que pode ser avaliada como *True ou False*. O *else* é usado para executar um bloco de código alternativo caso a condição do *if* seja *False*. Para lidar com múltiplas condições, você pode usar a estrutura *elif* (abreviação de "*else if*"). O *elif* permite testar uma nova condição se a condição do *if* anterior for *False*.

Segue um breve exemplo de sua utilização na Figura 25.

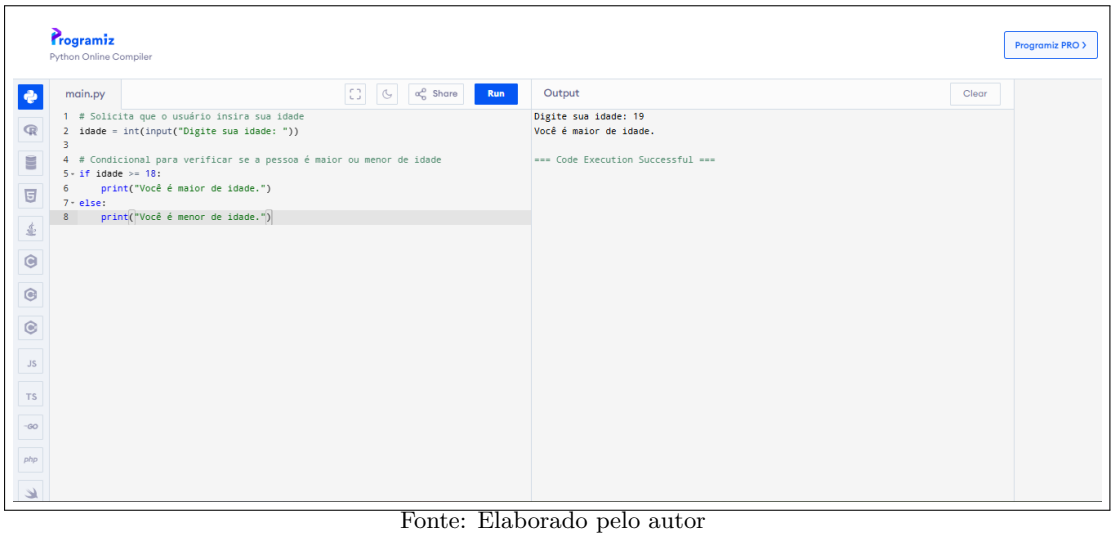
Figura 25 – Condicional *if,else ou elif*



Fonte: Elaborado pelo autor

A Figura 26 apresenta um exemplo prático do uso de estruturas condicionais na linguagem de programação Python, especificamente o comando *if*, que permite ao programa tomar decisões com base em uma condição lógica. Trata-se de um dos pilares da lógica computacional, cujo funcionamento se fundamenta em avaliar expressões *booleanas* e, a partir disso, executar diferentes blocos de código.

Figura 26 – Condicional if

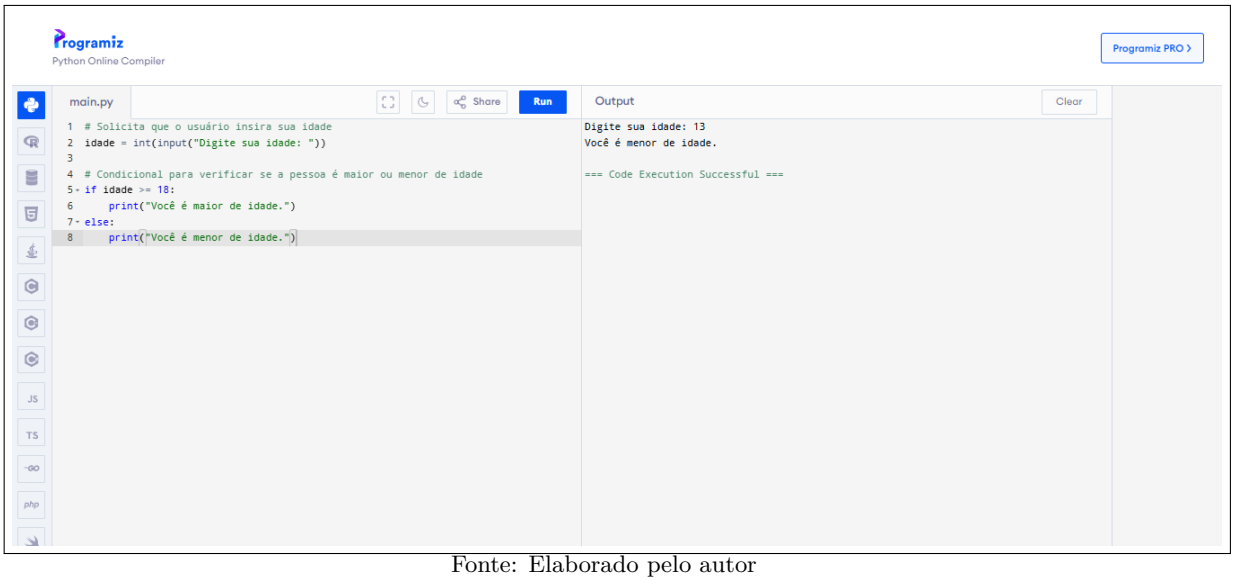


Fonte: Elaborado pelo autor

A Figura 27 ilustra o uso do comando else em Python, que é executado quando a condição anterior (*if*) é falsa. No exemplo, ao digitar a idade 13, como essa não satisfaz a condição idade >= 18, o programa segue para o bloco *else* e imprime: "Você é menor de idade.".

Essa estrutura é essencial para criar alternativas lógicas nos programas e tornar a tomada de decisão automatizada mais eficiente e organizada.

Figura 27 – Condicional else



Fonte: Elaborado pelo autor

Após essa jornada de conhecimento sobre os principais conceitos de Python, como tipos de dados, operadores e classes, os alunos já possuem uma base sólida para começar a resolver exercícios e praticar o que aprenderam. Esse é o momento ideal para aplicar a teoria, experimentar a criação de pequenos programas e desenvolver a lógica de programação, consolidando ainda mais os conteúdos estudados.

5 Resolvendo exercícios com auxilio da linguagem Python

Neste capítulo, serão apresentados exemplos de resolução de exercícios matemáticos voltados aos anos finais do Ensino Fundamental, com o auxílio da linguagem Python e com base no material didático fornecido pelo Estado de Mato Grosso. O objetivo é demonstrar como a programação pode ser uma ferramenta prática e eficaz para automatizar cálculos e explorar conceitos matemáticos de forma interativa. A proposta busca integrar o uso da tecnologia ao processo de aprendizagem, tornando a matemática mais acessível, dinâmica e alinhada às competências contemporâneas.

Antes de dar início a resolução de exercícios com o apoio da linguagem Python, é essencial apresentar o material didático que servirá de base para essa proposta.

O Material Estruturado de Ensino, implementado pelo Governo do Estado de Mato Grosso segundo Rosa (2022a),consiste em um conjunto de recursos pedagógicos destinados a aprimorar a qualidade do ensino nas escolas estaduais. Este sistema é composto por apostilas impressas, plataformas digitais, aplicativos educacionais, avaliações semestrais, exercícios complementares, banco de questões e programas de formação continuada para os professores, totalizando 120 horas anuais de capacitação.

As apostilas são elaboradas conforme Rosa (2022a), de forma regionalizada, alinhando-se à Base Nacional Comum Curricular (BNCC) e ao Documento de Referência Curricular de Mato Grosso (DRC-MT), garantindo que os conteúdos atendam às especificidades locais. Cada aluno recebe materiais correspondentes aos componentes curriculares de sua série, podendo utilizá-los tanto em sala de aula quanto em casa, sem a necessidade de devolução. Além disso, a plataforma virtual oferece acesso a conteúdos didáticos de diversas áreas do conhecimento, promovendo a pesquisa e o aprimoramento do aprendizado, e está disponível para toda a comunidade escolar, incluindo estudantes, familiares, diretores, coordenadores e professores.

De acordo Rosa (2022b) sua implementação tem como objetivo atender alunos e professores do ensino fundamental, médio e das modalidades de Educação de Jovens e Adultos (EJA) e Quilombola.

Em algumas disciplinas, o material sugere o uso de tecnologias educacionais, como plataformas *online*, vídeos, jogos e atividades interativas. Esses recursos são essenciais para o desenvolvimento de habilidades digitais e tornam o aprendizado mais atraente e dinâmico.

Nosso objetivo é oferecer uma experiência de aprendizagem mais dinâmica e enga-

jadora, combinando o conteúdo do material estruturado com a linguagem de programação Python. Em vez de apenas resolver exercícios no caderno, os alunos serão incentivados a traduzir os problemas e soluções para o Python, utilizando os conceitos aprendidos em sala de aula.

Para alinhar o uso da programação em Python como ferramenta pedagógica às propostas curriculares previstas para o 9º ano do Ensino Fundamental, foram selecionados conteúdos matemáticos que apresentam grande potencial para serem enriquecidos a partir de atividades de programação. A seleção contempla tópicos amplamente abordados nesse nível de ensino uma vez que foram retirados do material estruturado do governo de Mato Grosso, proporcionando oportunidades para a articulação entre a linguagem algorítmica e os conceitos matemáticos tradicionais, favorecendo a interdisciplinaridade e a construção do pensamento computacional dos estudantes.

Entre os temas escolhidos, destacam-se os problemas envolvendo potências e raízes, que possibilitam a exploração de propriedades numéricas. As equações do segundo grau, dado seu potencial para atividades que desenvolvam programas capazes de determinar raízes e interpretar resultados, promovendo a compreensão de conceitos algébricos de maneira dinâmica. As propriedades de triângulos e o Teorema de Pitágoras figuram como conteúdos que permitem programar cálculos automáticos de medidas e validar relações métricas fundamentais em figuras planas. Além disso, a trigonometria oferece oportunidade para que os educandos realizem cálculos de razões trigonométricas e resolvam problemas envolvendo ângulos e lados de triângulos retângulos. Relações métricas na circunferência e o cálculo de volumes de cilindros ampliam o repertório. Essa proposta didático-pedagógica visa, portanto, articular a resolução de problemas matemáticos com o uso da tecnologia e da programação, a ideia central é permitir que os alunos codifiquem as soluções para esses problemas usando Python, facilitando uma compreensão mais profunda das implicações teóricas e práticas dos conceitos matemáticos.

5.1 Potências e raízes

Potências e raízes são conceitos fundamentais na matemática que lidam com a multiplicação e a extração de fatores, respectivamente. A potência de um número é o resultado da multiplicação desse número por ele mesmo um determinado número de vezes, expressa na forma a^n, onde a é a base e n é o expoente. Por exemplo, 2^3 equivale a $2 \times 2 \times 2 = 8$.

As raízes, por outro lado, são usadas para determinar qual número, quando elevado a uma determinada potência, resulta em um valor específico. A raiz quadrada de um número x é representada como $\sqrt{x}$ e corresponde ao número que, quando elevado ao quadrado, produz x. Por exemplo, $\sqrt{16} = 4$, pois $4^2 = 16$.

Compreender potências e raízes é essencial não apenas para avançar em conceitos matemáticos mais complexos, mas também para aplicações práticas em diversas áreas, como física, engenharia e programação. Esses conceitos servem como ferramentas valiosas na resolução de uma ampla gama de problemas matemáticos.

5.1.1 Potências

A potência de um número representa uma multiplicação sucessiva. Como ilustra a Figura 19 a expressão a^n, onde a é a base e n é o expoente. O valor de a^n é o resultado de multiplicar a por si mesmo n vezes. Por exemplo, $2^3 = 2 \times 2 \times 2 = 8$.

Figura 28 – Potências

Potenciação de números reais

Se tomarmos um número real ***a*** e um natural ***n*** maior ou igual a 2, ***a*** elevado a ***n*** é escrito e definido por:

$$a^n = \underbrace{a \cdot a \cdot a \cdot \ldots \cdot a \cdot a}_{n \text{ fatores}}$$

Da expressão anterior, dizemos que ***a*** está elevado à enésima potência, sendo ***a*** a base e ***n*** o expoente da potência.

Fonte: Apostila 1 (ZATTONI; CARVALHO, 2023, p.5)

As potências possuem propriedades úteis que são amplamente utilizadas em diversos contextos matemáticos, como:

- Produto de potências de mesma base: $a^m \times a^n = a^{m+n}$.
- Divisão de potências de mesma base:

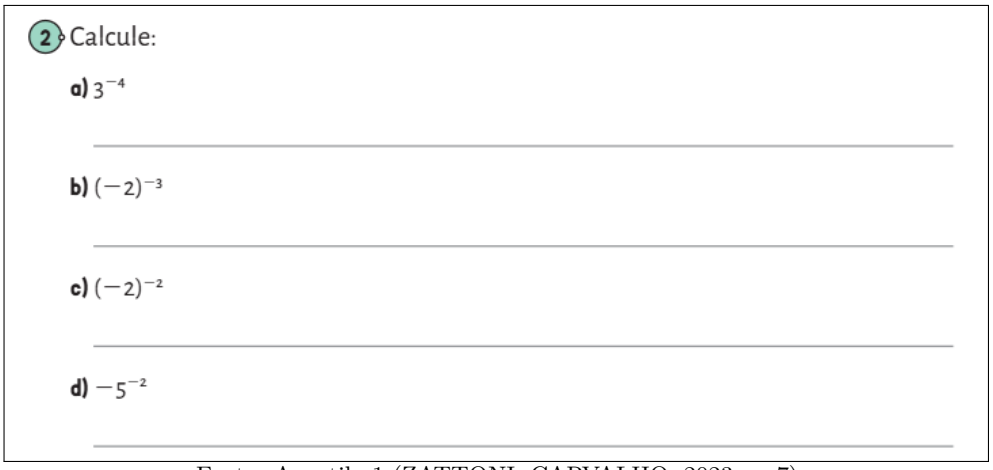
$$\frac{a^m}{a^n} = a^{m-n}, \quad \text{com } m > n \text{ e } a \neq 0.$$

- Potência de uma potência: $(a^m)^n = a^{m \times n}$.

5.1.2 Exercício 1

A Figura 29 apresenta um exercício introdutório sobre o conceito de potências com expoente negativo, conteúdo essencial no desenvolvimento da compreensão algébrica dos estudantes do ensino fundamental. O exercício propõe quatro itens nos quais os alunos devem calcular expressões que envolvem bases positivas e negativas elevadas a expoentes negativos, o que exige atenção à aplicação correta das propriedades das potências, especialmente no que se refere à inversão da base e ao uso adequado dos parênteses.

Figura 29 – Exercício 1



Fonte: Apostila 1 (ZATTONI; CARVALHO, 2023, p.7)

Passo a passo para resolver de maneira tradicional:

a) 3^{-4}

Explicação: Quando o expoente é negativo, inverte a base e torna-se o expoente positivo:

$$3^{-4} = \frac{1}{3^4} = \frac{1}{81}$$

b) $(-2)^{-3}$

Explicação: O expoente negativo inverte a base, e resolve a potência:

$$(-2)^{-3} = \frac{1}{(-2)^3} = \frac{1}{-8} = -\frac{1}{8}$$

É importante recordar que, ao elevar uma base negativa a um expoente par, o resultado será sempre positivo, uma vez que a multiplicação de um número negativo por ele mesmo, em quantidade par de vezes, elimina o sinal negativo. Por outro lado, quando a base negativa é elevada a um expoente ímpar, o resultado será sempre negativo, pois a quantidade ímpar de fatores negativos preserva o sinal.

c) $(-2)^{-2}$

Explicação: Invertendo a base e tornando o expoente positivo, tem-se:

$$(-2)^{-2} = \frac{1}{(-2)^2} = \frac{1}{4}$$

d) -5^{-2}

Explicação: O sinal negativo antes da base é mantido:

$$-5^{-2} = -\frac{1}{5^2} = -\frac{1}{25}$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

No Python, usa-se o operador `**` para calcular potências, inclusive com expoentes negativos. As regras básicas são:

$$a^{-n} = \frac{1}{a^n}.$$

Se o número for negativo, coloca-se entre parênteses: `(-2) ** (-3)`.

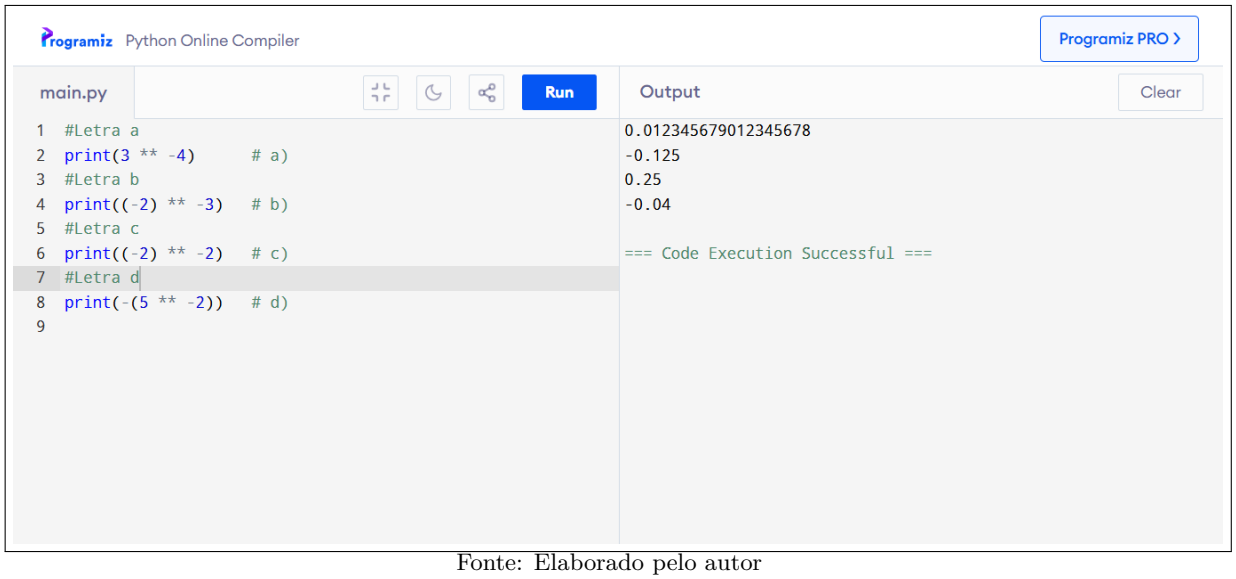
O sinal negativo antes de um número, como em -5^{-2}, é tratado fora da potência.

```
    # Resolução das potências com expoentes negativos

#Letra a
print(3 ** -4)      # a)
#Letra b
print((-2) ** -3)   # b)
#Letra c
print((-2) ** -2)   # c)
#Letra d
print(-(5 ** -2))   # d)
```

Ao final desses comandos o estudante clicará no botão *Run* (Executar).

A Figura 30 ilustra a aplicação da linguagem de programação Python, por meio da plataforma Programiz, na resolução de expressões envolvendo potências com expoentes negativos. Essa abordagem computacional oferece ao estudante a oportunidade de verificar, por meio de um ambiente de codificação, os resultados matemáticos obtidos anteriormente de forma manual.

Figura 30 – Resolução com o Python do exercício 1



Fonte: Elaborado pelo autor

5.1.2.1 Importando *fraction*

Esse código utiliza o módulo *fractions* do Python para calcular potências com expoentes negativos e apresentar os resultados como frações exatas. Isso evita arredondamentos e torna a resposta mais precisa, especialmente em contextos matemáticos.

Na Figura 31 importando a função Fraction() converte os números em frações, e o operador ** realiza a potenciação normalmente, seguindo as regras matemáticas de expoente negativo: inverter a base e aplicar a potência.

Figura 31 – Resolução com o Python importando *fraction*

Fonte: Elaborado pelo autor

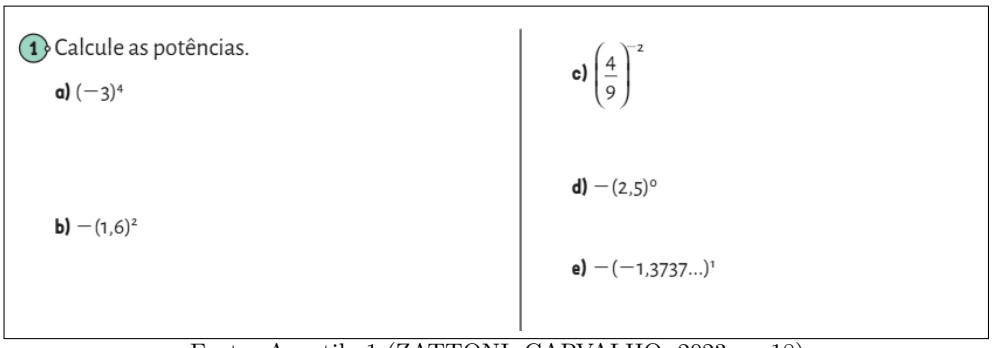
A utilização do Python permite reforçar conceitos como a inversão da base, a leitura correta dos parênteses e a operação algébrica com números racionais. Além disso, promove

a interdisciplinaridade entre Matemática e Computação, favorecendo uma aprendizagem mais dinâmica, investigativa e alinhada às competências exigidas pela BNCC no tocante ao uso de tecnologias digitais no processo de ensino-aprendizagem.

5.1.3 Exercício 2

A Figura 32 apresenta um conjunto de expressões numéricas destinadas ao cálculo de potências com expoentes inteiros, positivos e nulos, utilizando diferentes tipos de bases: negativas, fracionárias, decimais e irracionais.

Figura 32 – Exercício 2



Fonte: Apostila 1 (ZATTONI; CARVALHO, 2023, p.18)

Passo a passo para resolver de maneira tradicional:

a) $(-3)^4$

$$(-3)^4 = (-3)\cdot(-3)\cdot(-3)\cdot(-3) = 81$$

b) $-(1,6)^2$

$$(1,6)^2 = 2,56 \quad \text{e} \quad -(1,6)^2 = -2,56$$

c) $\left(\frac{4}{9}\right)^2$

$$\left(\frac{4}{9}\right)^2 = \frac{4^2}{9^2} = \frac{16}{81}$$

d) $-(2,5)^0$

$$(2,5)^0 = 1 \quad \text{e} \quad -(2,5)^0 = -1$$

e) $-(-1,3737...)^1$

$$(-1,3737...)^1 = -1,3737... \quad \text{e} \quad -(-1,3737...) = 1,3737...$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
    # Letra a
print((-3) ** 4)
```

```
# Letra b
print((1.6) ** 3)

# Letra c
from fractions import Fraction
print((Fraction(4, 9)) ** -2)

# Letra d
print((-2.5) ** 0)

# Letra e
print((-1.3737) ** 1)
```

Ao final desses comandos o estudante clicará no botão *Run* (Executar).

A Figura 33 ilustra a resolução computacional, utilizando a linguagem Python no ambiente Programiz, dos cálculos propostos. A atividade destaca o uso da biblioteca fractions, que permite a manipulação de números racionais de forma exata, especialmente em expressões envolvendo frações elevadas a expoentes inteiros. O código também contempla a manipulação de números decimais e irracionais representados com aproximações, reforçando a importância de tratar com rigor diferentes tipos de números no contexto das potências.

Figura 33 – Resolução em Python do exercício 2

Fonte: Elaborado pelo autor

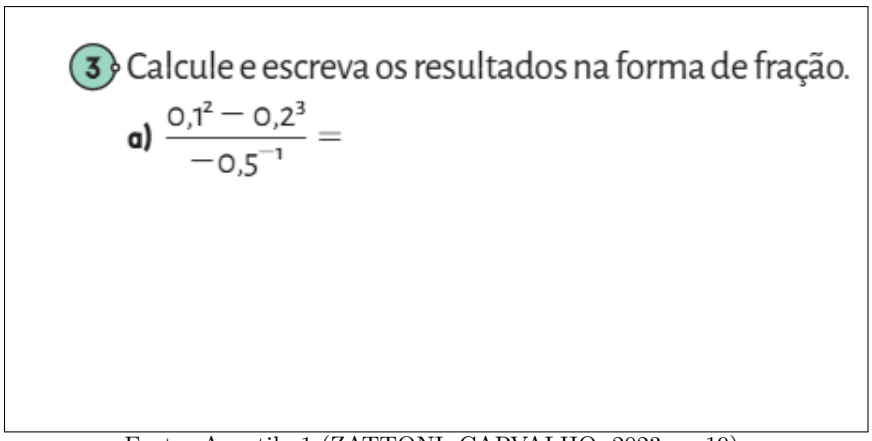
O exercício propõe uma diversidade de situações que exigem do estudante a compreensão aprofundada das propriedades das potências, como a paridade do expoente, o efeito do zero como expoente e a potência de números negativos. Além disso, a

inclusão de frações e números decimais aproxima o conteúdo das situações-problema mais complexas, contribuindo para o desenvolvimento da flexibilidade cognitiva e da competência matemática dos discentes ao lidarem com representações diversas de números reais.

5.1.4 Exercício 3

A Figura 34 propõe uma atividade mais complexa envolvendo potências com números decimais, solicitando que o resultado final seja expresso na forma de fração irredutível. O exercício demanda do estudante não apenas o domínio das propriedades das potências e das operações com números decimais, mas também a habilidade de converter esses valores para frações equivalentes.

Figura 34 – Exercício 3



Fonte: Apostila 1 (ZATTONI; CARVALHO, 2023, p.19)

Passo a passo para resolver de maneira tradicional:

Calcule e escreva os resultados na forma de fração.

a) $\frac{0,1^2-0,2^3}{-0,5^{-1}}$

Passo a passo:

1. Primeiro, calcula-se cada potência:

$$0,1^2 = 0,01 \quad \text{e} \quad 0,2^3 = 0,008$$

$$0,5^{-1} = \frac{1}{0,5} = 2$$

2. Substituí-se na expressão:

$$\frac{0,1^2 - 0,2^3}{-0,5^{-1}} = \frac{0,01 - 0,008}{-2}$$

3. Simplifica-se o numerador:

$$0,01 - 0,008 = 0,002$$

4. Substituí-se novamente:

$$\frac{0,002}{-2}$$

5. Realiza-se a divisão:

$$\frac{0,002}{-2} = -0,001$$

6. Escrevendo $-0,001$ como fração:

$$-0,001 = -\frac{1}{1000}$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

Para que os estudantes consigam resolver o exercício em Python mostrado na Figura 34, deve-se seguir os seguintes passos de forma organizada. Identificação das operações matemáticas, começando a identificar cada operação matemática que será realizada no exercício, tanto no numerador quanto no denominador.

Definição das variáveis: Declare as variáveis necessárias para armazenar o numerador e o denominador separadamente. Isso permitirá realizar operações de maneira mais organizada.

Cálculo das expressões: Realize os cálculos necessários para obter o valor do numerador e do denominador de forma separada, utilizando as operações básicas como potenciação.

Divisão para obter o resultado: Dividindo o resultado do numerador pelo denominador para obter o resultado final.

Conversão para fração (opcional);

Caso seja necessário apresentar o resultado em forma de fração, importe o módulo fractions do Python e utilize a função Fraction() para converter o valor decimal.

```
# Passo 1: Definir os valores da expressão
num1 = 0.1**2      # Calcula 0,1²
num2 = 0.2**3      # Calcula 0,2³
denominator = -(0.5**-1)

# Passo 2: Subtrair os valores no numerador
numerator = num1 - num2  # 0,1² - 0,2³

# Passo 3: Dividir o numerador pelo denominador
```

```
result = numerator / denominator

# Passo 4: Exibir o resultado em decimal e fração
from fractions import Fraction

# Converte o resultado para fração
fraction_result = Fraction(result).limit_denominator()

# Exibe os resultados
print("Resultado decimal:", result)
print("Resultado em fração:", fraction_result)
```

Exibição dos resultados: Utilize a função de impressão *(print())* para mostrar o resultado em formato decimal e, se aplicável, em formato de fração. Isso garantirá que os alunos compreendam as diferentes representações do mesmo valor. Esse passo a passo incentiva a decomposição do problema em partes menores, promovendo uma melhor compreensão dos conceitos matemáticos envolvidos e do uso do Python como ferramenta para a resolução.

Ao clicar na tecla *"Run"* para executar o código que escreveu, na interface, é processado (ou compilado, dependendo se houver erro) e executado. Isso significa que a plataforma irá interpretar ou compilar o código, executar os comandos que você escreveu e exibir o resultado ou saída correspondente na tela. É uma maneira de testar o funcionamento do seu código e verificar se ele está produzindo os resultados esperados.

A Figura 35 evidencia a aplicação da linguagem Python na resolução algébrica de uma expressão envolvendo potências com números decimais. Por meio da plataforma Programiz, o código realiza o cálculo passo a passo: primeiro eleva os decimais às potências solicitadas, depois efetua as operações indicadas no numerador e denominador da expressão e, por fim, converte o resultado decimal para uma fração irredutível com o auxílio da biblioteca fractions. Essa prática valoriza a precisão matemática e a verificação computacional de resultados, favorecendo a interdisciplinaridade entre Matemática e Computação.

Figura 35 – Resolução em Python do exercício 3

Fonte: Elaborado pelo autor

5.1.5 Raízes

Na matemática, as raízes e os radicais são conceitos fundamentais que envolvem a extração de raízes de números. A raiz de um número x é um valor que, quando elevado a uma determinada potência n, resulta em x. O radical é a operação inversa da exponenciação.

A notação para a raiz quadrada de um número x é denotada por $\sqrt{x}$. Por exemplo, $\sqrt{9} = 3$ porque $3^2 = 9$. As raízes também podem ser expressas em termos de potências: a raiz n-ésima de x pode ser escrita como $x^{1/n}$.

Os radicais podem ser simplificados se o radicando — o número sob o radical — puder ser fatorado de uma maneira que envolva quadrados perfeitos (no caso da raiz quadrada) ou potências perfeitas gerais.

Um exemplo de simplificação é:

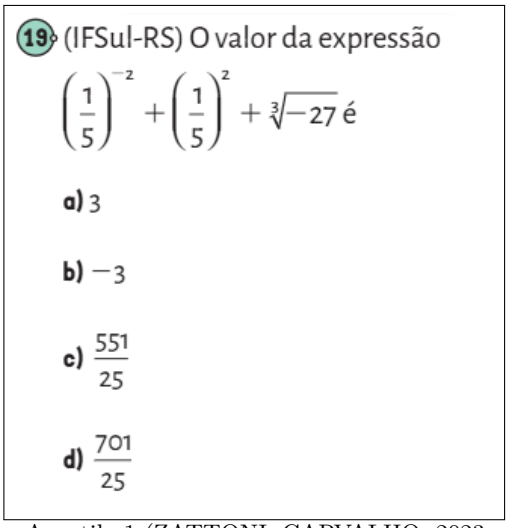
$$\sqrt{18} = \sqrt{9 \cdot 2} = \sqrt{9} \cdot \sqrt{2} = 3\sqrt{2}$$

Os conceitos de raízes e radicais são amplamente utilizados em diversas áreas da matemática e têm aplicações em física, engenharia e outras ciências.

5.1.6 Exercício 4

A Figura 35 apresenta o enunciado de uma expressão matemática que envolve potências negativas, potências positivas e raízes cúbicas. Ilustra um tipo de questão recorrente em avaliações escolares e exames, destacando a importância da interpretação correta das propriedades das potências e das raízes.

Figura 36 – Exercício 4



Fonte: Apostila 1 (ZATTONI; CARVALHO, 2023, p.24)

Passo a passo para resolver de maneira tradicional:

Calcular o valor da expressão:

$$\left(\frac{1}{5}\right)^{-2}+\left(\frac{1}{5}\right)^{2}+\sqrt[3]{-27}$$

Resolução

Primeiramente, resolve-se cada termo da expressão:

- O primeiro termo é:
$$\left(\frac{1}{5}\right)^{-2}=\left(\frac{5}{1}\right)^{2}=5^2=25.$$
- O segundo termo é:
$$\left(\frac{1}{5}\right)^{2}=\frac{1^2}{5^2}=\frac{1}{25}.$$
- O terceiro termo é a raiz cúbica de -27:
$$\sqrt[3]{-27}=-3.$$

Agora, os três valores obtidos são somados:

$$25+\frac{1}{25}+(-3).$$

Para somar, coloca-se todos os termos no mesmo denominador:

$$25=\frac{625}{25},\quad -3=\frac{-75}{25}.$$

Portanto:

$$\frac{625}{25}+\frac{1}{25}-\frac{75}{25}=\frac{625+1-75}{25}.$$

Somando-se o numerador:

$$625 + 1 - 75 = 551.$$

Assim, a soma total é:

$$\frac{551}{25}.$$

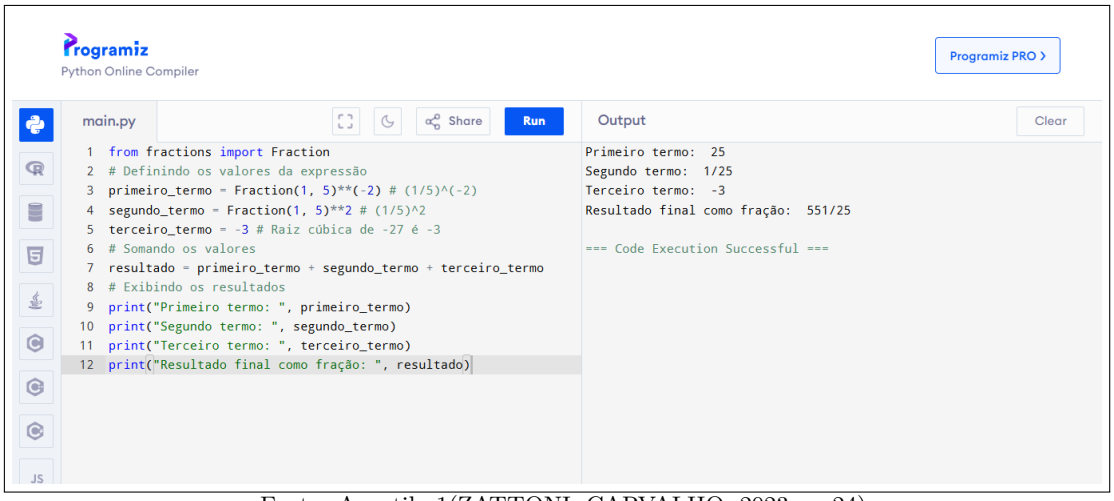
A sugestão para resolução em Python pode ser feita da seguinte forma:

```
    from fractions import Fraction

# Definindo os valores da expressão
primeiro_termo = Fraction(1, 5)**(-2)  # (1/5)^(-2)
segundo_termo = Fraction(1, 5)**2      # (1/5)^2
terceiro_termo = -3                   # Raiz cúbica de -27 é -3

# Somando os valores
resultado = primeiro_termo + segundo_termo + terceiro_termo

# Exibindo os resultados
print("Primeiro termo: ", primeiro_termo)
print("Segundo termo: ", segundo_termo)
print("Terceiro termo: ", terceiro_termo)
print("Resultado final como fração: ", resultado)
```

Ao final desses comandos, o estudante clicará no botão *Run* (Executar).

A Figura 37 apresenta a resolução computacional de uma expressão numérica que envolve potências com expoentes negativos e positivos, além de uma raiz cúbica. Por meio do uso da linguagem Python, a figura evidencia como é possível utilizar recursos da linguagem de programação para automatizar cálculos e obter resultados precisos, inclusive com representação fracionária. Reforçando a importância de integrar ferramentas digitais ao ensino de Matemática, tornando os procedimentos algébricos mais acessíveis e visualmente compreensíveis para os estudantes.

Figura 37 – Resolução em Python do exercício 4



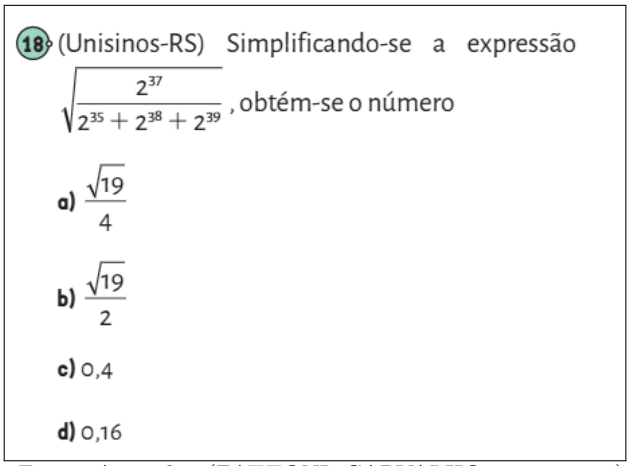
Fonte: Apostila 1(ZATTONI; CARVALHO, 2023, p.24)

A atividade serve como base para análises tanto no âmbito da resolução manual quanto na aplicação computacional, onde erros de sintaxe ou prioridade de operações podem comprometer a obtenção do resultado correto.

5.1.7 Exercício 5

A Figura 38 apresenta um problema clássico envolvendo a simplificação de uma expressão algébrica com potências de base dois e radiciação. A proposta desafia o aluno a reconhecer padrões e aplicar propriedades das potências para reduzir a complexidade da raiz. Trata-se de uma excelente oportunidade para reforçar a importância do fator comum e da manipulação algébrica na resolução de expressões numéricas mais elaboradas.

Figura 38 – Exercício 5



Fonte: Apostila 1(ZATTONI; CARVALHO, 2023, p.24)

Passo a passo para resolver de maneira tradicional:

1.Primeiramente, analisa-se o denominador da expressão:

$$2^{35} + 2^{38} + 2^{39}.$$

2.Fatora-se o menor expoente, 2^{35}, em cada termo:

$$2^{35} + 2^{38} + 2^{39} = 2^{35}(1 + 2^3 + 2^4).$$

3.Simplifica-se os termos dentro do parêntese:

$$1 + 2^3 + 2^4 = 1 + 8 + 16 = 25.$$

4.Portanto, o denominador fica:

$$2^{35}(1 + 2^3 + 2^4) = 2^{35} \cdot 25.$$

5.Agora substituí-se na expressão original:

$$\sqrt{\frac{2^{37}}{2^{35} \cdot 25}}.$$

6.Simplifica-se o numerador e o denominador:

$$\frac{2^{37}}{2^{35} \cdot 25} = \frac{2^{37}}{2^{35} \cdot 25} = \frac{2^{37-35}}{25} = \frac{2^2}{25}.$$

7.Sabe-se que $2^2 = 4$, então tem-se:

$$\frac{2^2}{25} = \frac{4}{25}.$$

8.Aplica-se a raiz quadrada:

$$\sqrt{\frac{4}{25}} = \frac{\sqrt{4}}{\sqrt{25}} = \frac{2}{5}.$$

9.Finalmente, convertendo para decimal:

$$\frac{2}{5} = 0,4.$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

Passos: Para resolver o exercício apresentado na imagem usando Python, os alunos devem seguir os seguintes passos:

Importação do módulo matemático (opcional): Se houver necessidade de realizar operações avançadas, como raízes quadradas, utilize a biblioteca math. Neste caso, importa-se o módulo com import math.

Definição do numerador: Declare uma variável para armazenar o valor do numerador usando operações de potenciação, como 2 ** 37.

Definição do denominador: Calcule e armazene o valor do denominador, que pode ser uma soma de diferentes expressões envolvendo operações de potenciação.

Cálculo da expressão: Realize a divisão do numerador pelo denominador e, se necessário, aplique funções matemáticas como a raiz quadrada, utilizando math.sqrt(), para obter o resultado desejado.

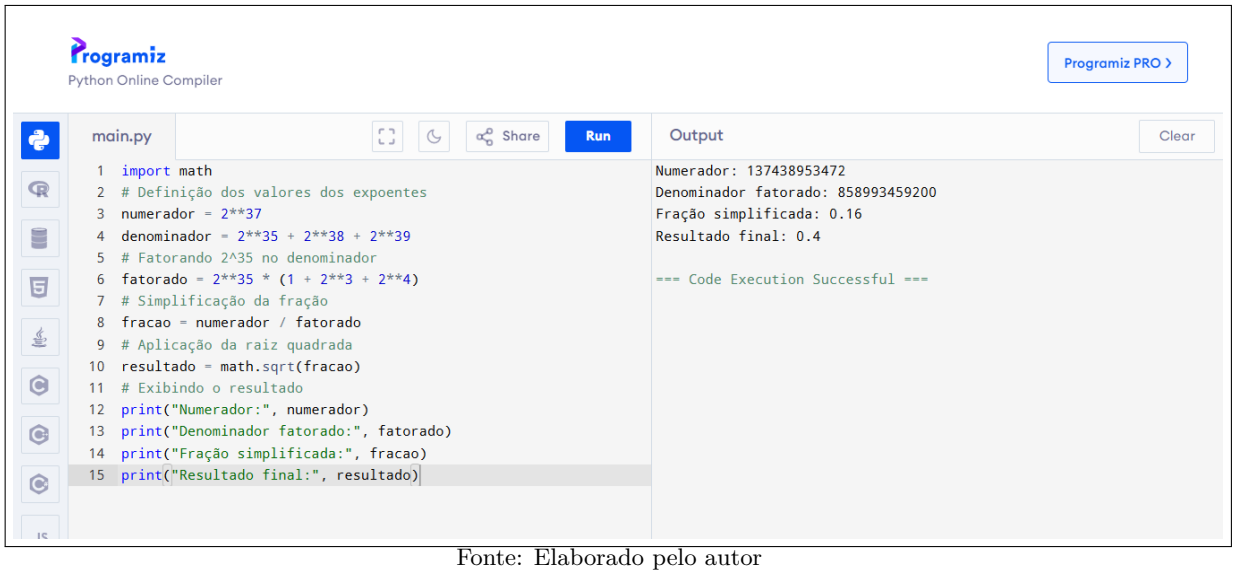
Exibição do resultado: Utilize a função print() para exibir o resultado final do cálculo. Seguindo essa estrutura, os alunos poderão resolver problemas que envolvem cálculos com potências e funções matemáticas de maneira organizada e eficiente em Python.

```
    import math

# Definição dos valores dos expoentes
numerador = 2**37
denominador = 2**35 + 2**38 + 2**39

# Fatorando 2^35 no denominador
fatorado = 2**35 * (1 + 2**3 + 2**4)

# Simplificação da fração
fracao = numerador / fatorado

# Aplicação da raiz quadrada
resultado = math.sqrt(fracao)

# Exibindo o resultado
print("Numerador:", numerador)
print("Denominador fatorado:", fatorado)
print("Fração simplificada:", fracao)
print("Resultado final:", resultado)
```

Ao final desse comando, o estudante clicará no botão *Run* (Executar).

A Figura 39 ilustra a implementação computacional de uma expressão matemática por meio da linguagem Python. A partir da definição das potências e da simplificação algébrica do denominador, o código realiza a divisão e, posteriormente, a extração da raiz quadrada.

Figura 39 – Resolução em Python do exercício 5



Fonte: Elaborado pelo autor

5.2 Equação do 2º grau

A equação de segundo grau ocupa um lugar central no estudo da matemática, revelando padrões e estruturas que perpassam diversas áreas do conhecimento. Na sua forma geral, ela é escrita como:

$$ax^2 + bx + c = 0, \tag{5.1}$$

Onde a, b e c são coeficientes reais, com $a \neq 0$. O método clássico para resolver esta equação é o uso da fórmula de Bhaskara, que leva o nome do matemático indiano Bhaskara Acharya, mas que foi popularizada no Brasil como ferramenta essencial para o ensino de álgebra.

Resolver equações de segundo grau é, portanto, um exercício de engenho e paciência. Ao encontrar as raízes, o aluno não está apenas realizando uma tarefa mecânica, mas desvendando relações que podem modelar fenômenos naturais e sociais.

Para encontrar as soluções de uma equação de segundo grau, utiliza-se a fórmula de Bhaskara, expressa por:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \tag{5.2}$$

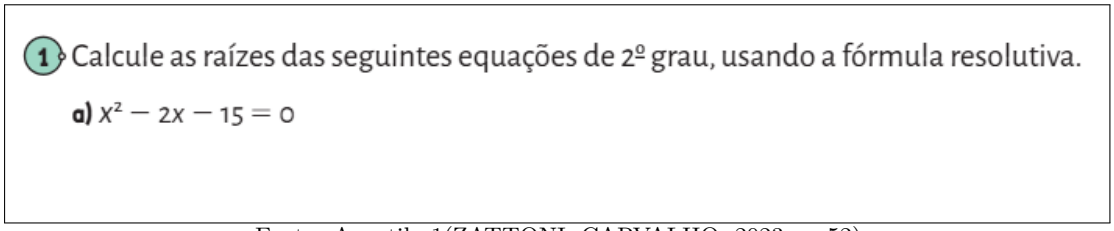
Esta fórmula permite calcular as raízes, ou seja, os valores de x que tornam a equação verdadeira. O termo discriminante, representado por $\Delta = b^2 - 4ac$, nos fornece informações cruciais sobre o tipo de solução: se $\Delta > 0$, tem-se duas raízes reais e distintas; se $\Delta = 0$, tem-se uma raiz real dupla; e se $\Delta < 0$, não há raízes reais.

O estudo da equação de segundo grau, longe de ser um desafio isolado, é um convite ao raciocínio lógico, à criatividade e à compreensão dos padrões que regem o mundo.

5.2.1 Exercício 6

A Figura 40 apresenta um problema clássico da álgebra elementar, no qual se solicita a resolução de uma equação do segundo grau utilizando a fórmula resolutiva, conhecida como fórmula de Bhaskara. Esse tipo de atividade é fundamental no processo de consolidação dos conhecimentos algébricos dos estudantes, pois trabalha diretamente com conceitos como coeficientes, discriminante (Δ) raízes reais e complexas, promovendo a compreensão do comportamento dos gráficos de funções quadráticas.

Figura 40 – Exercício 6



Fonte: Apostila 1(ZATTONI; CARVALHO, 2023, p.52)

Passo a passo para resolver de maneira tradicional:

1.Para resolver a equação de segundo grau com coeficientes $a = 1$, $b = -2$ e $c = -15$, utiliza-se a fórmula de Bhaskara. O discriminante (ou Delta) é calculado pela expressão:

$$\Delta = b^2 - 4ac$$

2.Com os valores fornecidos, tem-se:

$$\Delta = (-2)^2 - 4 \cdot 1 \cdot (-15) = 4 + 60 = 64$$

3.As raízes da equação são dadas por:

$$x = \frac{-b \pm \sqrt{\Delta}}{2a}$$

4.Aplicando os valores, obtendo assim:

$$x_1 = \frac{-(-2) + \sqrt{64}}{2 \cdot 1} = \frac{2 + 8}{2} = 5$$
$$x_2 = \frac{-(-2) - \sqrt{64}}{2 \cdot 1} = \frac{2 - 8}{2} = -3$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
import math

# Coeficientes da equação
a = 1
b = -2
c = -15

# Cálculo do discriminante
delta = b**2 - 4*a*c

# Cálculo das raízes
x1 = (-b + math.sqrt(delta)) / (2*a)
x2 = (-b - math.sqrt(delta)) / (2*a)

# Exibindo os resultados
print(f"As raízes da equação são: x1 = {x1} e x2 = {x2}")
```

A execução do código resultou nas seguintes raízes:

- $x_1 = 5.0$
- $x_2 = -3.0$

Dessa forma, verifica-se que a solução da equação está de acordo com o esperado, utilizando a fórmula de Bhaskara e confirmando os resultados através de uma implementação computacional.

A Figura 41 evidencia a aplicação prática da linguagem de programação Python na resolução de equações quadráticas. Através da utilização da biblioteca math, é possível calcular o discriminante e, posteriormente, aplicar a fórmula de Bhaskara para determinar as raízes reais da equação.

Figura 41 – Resolução em Python do exercício 6

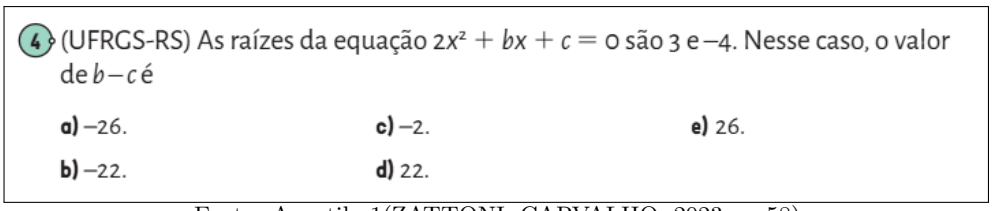
Fonte: Elaborado pelo autor

Este tipo de abordagem da Figura 41 não apenas automatiza o processo de resolução, como também desenvolve nos estudantes habilidades algorítmicas e lógico-matemáticas, integrando o pensamento matemático à prática da programação.

5.2.2 Exercício 7

A Figura 42 apresenta uma questão clássica envolvendo as relações de Girard, em que as raízes da equação do segundo grau são fornecidas como 3 e –4. O desafio proposto consiste em determinar o valor da expressão algébrica, a partir dessas raízes. Tal exercício reforça o entendimento de que, conhecendo as raízes, é possível reconstruir a equação e determinar seus coeficientes, habilidade essencial no estudo da álgebra.

Figura 42 – Exercício 7



Fonte: Apostila 1(ZATTONI; CARVALHO, 2023, p.58)

Passo a passo para resolver de maneira tradicional:

1.Considerando os coeficientes e raízes fornecidos para uma equação quadrática, tem-se os seguintes valores:

- Coeficiente: $a = 2$
- Raízes: $\text{raiz}_1 = 3$ e $\text{raiz}_2 = -4$

2.O cálculo dos coeficientes b e c pode ser realizado utilizando as relações da soma e produto das raízes:

$$\text{soma\_raizes} = \text{raiz}_1 + \text{raiz}_2 = 3 + (-4) = -1$$

$$\text{produto\_raizes} = \text{raiz}_1 \times \text{raiz}_2 = 3 \times (-4) = -12$$

3.Com isso, calcula-se os valores de b e c como:

$$b = -a \times \text{soma\_raizes} = -2 \times (-1) = 2$$

$$c = a \times \text{produto\_raizes} = 2 \times (-12) = -24$$

4.Finalmente, calcula-se a diferença $b - c$:

$$b - c = 2 - (-24) = 26$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
    # Coeficientes
a = 2
raiz1 = 3
raiz2 = -4

# Cálculo da soma e produto das raízes
soma_raizes = raiz1 + raiz2
produto_raizes = raiz1 * raiz2

# Usando as relações
b = -a * soma_raizes
c = a * produto_raizes

# Cálculo de b - c
resultado = b - c

# Exibindo o resultado
print(f"O valor de b - c é: {resultado}")
```

O resultado da execução do código é:

$$b - c = 26$$

Esse exemplo mostra como as relações entre os coeficientes e raízes de uma equação quadrática podem ser utilizadas para realizar cálculos de maneira direta.

A Figura 43 mostra a aplicação prática de programação no ambiente Programiz para resolver uma questão algébrica envolvendo raízes conhecidas de uma equação quadrática.

Figura 43 – Resolução em Python do exercício 7

Fonte: Elaborado pelo autor

A Figura 43 utilizando a linguagem Python, o código realiza o cálculo dos coeficientes da equação por meio das relações de soma e produto das raízes e, em seguida, determina o valor da expressão algébrica b-c. A resposta final, exibida no compilador, confirma o valor 26, validando o uso do pensamento computacional como recurso pedagógico eficaz no ensino da álgebra.

5.3 Semelhança de triângulos

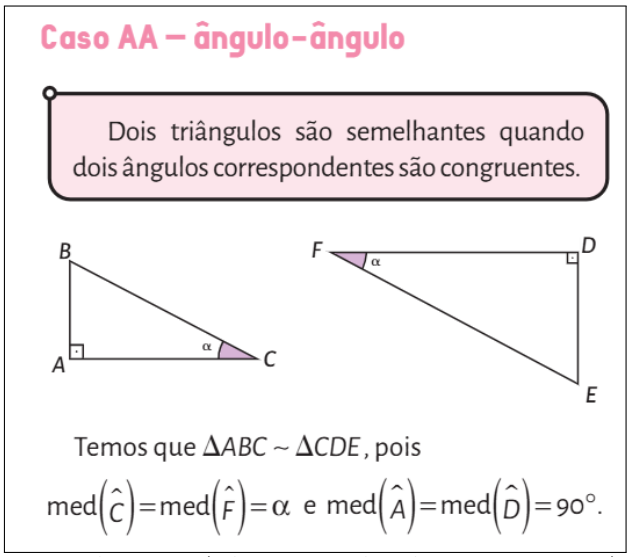
A seleção de exercícios envolvendo triângulos foi realizada de forma criteriosa, com o objetivo de proporcionar aos alunos uma experiência que integrasse os fundamentos da geometria com a prática da programação em Python. O foco esteve na escolha de problemas que permitissem a aplicação direta de conceitos geométricos por meio de algoritmos, promovendo um aprendizado significativo ao unir raciocínio matemático e pensamento computacional.

O estudo dos triângulos semelhantes é um tema essencial na geometria, com aplicações práticas que vão desde a medição indireta de alturas até o entendimento de escalas em mapas e desenhos técnicos. Dois triângulos são considerados semelhantes quando possuem os ângulos correspondentes congruentes e os lados proporcionais. Este conceito é fundamental para a compreensão de proporções, razão e proporcionalidade, e é amplamente utilizado na matemática.

Para determinar se dois triângulos são semelhantes, pode-se utilizar os critérios de semelhança:

- **AA (Ângulo-Ângulo):** Se dois ângulos de um triângulo são congruentes com dois ângulos de outro triângulo, então os triângulos são semelhantes como mostra a Figura 44.

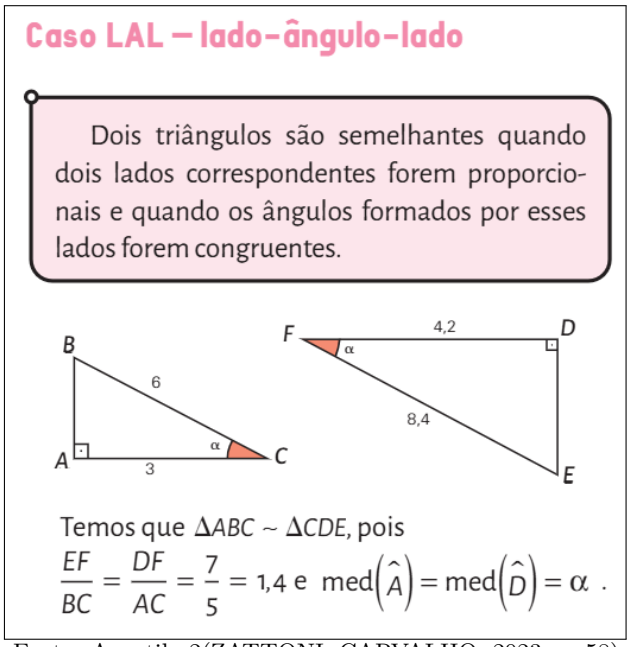
Figura 44 – Caso AA - Ângulo - Ângulo



Fonte: Apostila 2(ZATTONI; CARVALHO, 2023, p.58)

- **LAL (Lado-Ângulo-Lado):** Se dois lados de um triângulo são proporcionais a dois lados de outro triângulo e os ângulos formados por esses lados são congruentes, os triângulos são semelhantes como mostra a Figura 45.

Figura 45 – Caso LAL - Lado - Ângulo - Lado

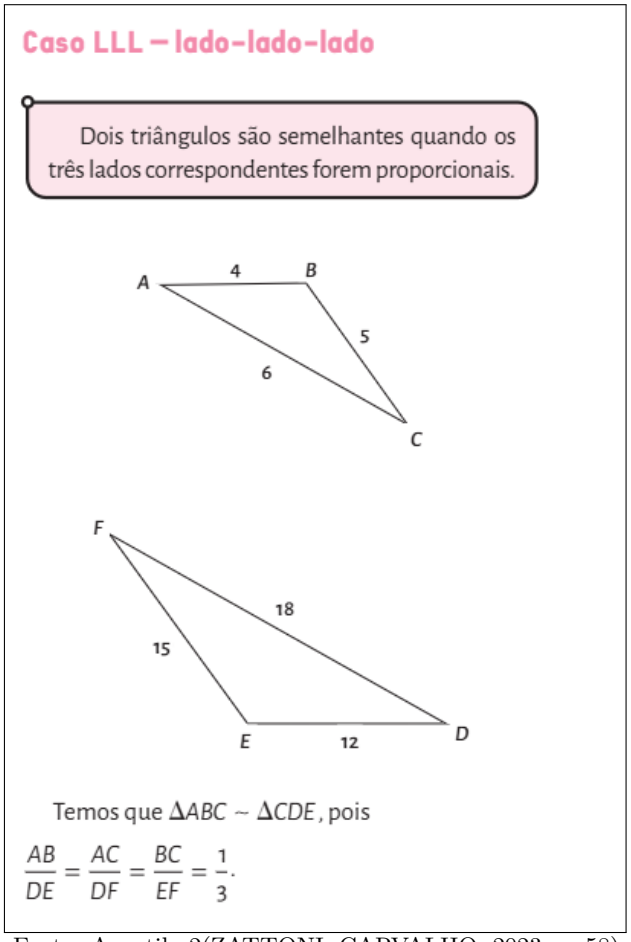


Fonte: Apostila 2(ZATTONI; CARVALHO, 2023, p.58)

- **LLL (Lado-Lado-Lado):** Se os três lados de um triângulo são proporcionais aos

três lados de outro triângulo, os triângulos são semelhantes como mostra a Figura 46.

Figura 46 – Caso LAL - Lado - Lado - Lado

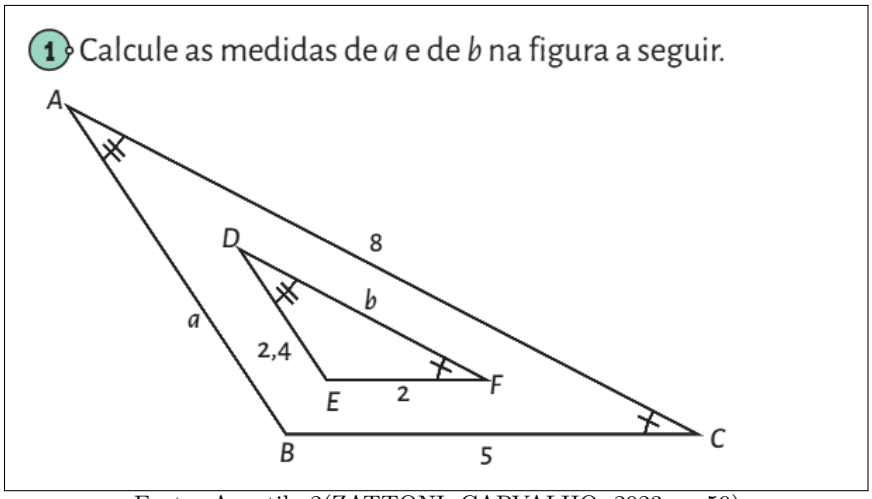


Fonte: Apostila 2(ZATTONI; CARVALHO, 2023, p.58)

5.3.1 Exercício 8

A Figura 47 apresenta uma construção geométrica clássica que aborda a aplicação dos critérios de semelhança entre triângulos para o cálculo de lados desconhecidos. Na figura, os triângulos $\triangle ABC$ e $\triangle DEF$ compartilham ângulos congruentes, conforme indicado pelas marcações, o que garante que os triângulos são semelhantes pelo Caso AA.

Figura 47 – Exercício 8



Fonte: Apostila 2(ZATTONI; CARVALHO, 2023, p.59)

Passo a passo para resolver de maneira tradicional:

1.Para determinar as medidas dos segmentos a e b na Figura 38, utiliza-se as relações de semelhança entre os triângulos. Tem-se os seguintes dados:

- $EF = 2$, $DE = 2.4$, $DF = b$, $BC = 5$, $AC = 8$, $AB = a$

2.Primeiramente, aplica-se a relação de semelhança entre os triângulos EFG e BCD para encontrar o valor de b:

$$\frac{EF}{BC} = \frac{DF}{AC} \implies \frac{2}{5} = \frac{b}{8}$$

3.Multiplicando cruzadamente, tem-se:

$$5b = 16 \implies b = \frac{16}{5} = 3,2$$

4.Agora, para encontrar o valor de a, utiliza-se a relação de semelhança entre os triângulos EFG e ABC:

$$\frac{EF}{BC} = \frac{DE}{AB} \implies \frac{2}{5} = \frac{2,4}{a}$$

5.Multiplicando cruzadamente, obtem-se:

$$2a = 12 \implies a = \frac{12}{2} = 6$$

6.Portanto, os valores calculados são:

- $b = 3,2$
- $a = 6$

Ao adotar os passos a serem tomados, exemplifica como utilizar as propriedades de semelhança de triângulos para calcular medidas proporcionais de segmentos, aplicando relações de razão entre lados correspondentes.

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
# Resolver usando proporção

# Proporção para encontrar 'a'
DE = 2.4
EF = 2
BC = 5

a = (DE * BC) / EF

# Proporção para encontrar 'b'
AC = 8
b = (EF * AC) / BC

print(f"Valor de a: {a}")
print(f"Valor de b: {b}")
```

A Figura 48 apresenta uma abordagem computacional aplicada à resolução de um problema clássico de geometria. Utilizando os conceitos de semelhança entre triângulos e a proporcionalidade entre seus lados correspondentes, o código desenvolvido em Python resolve um sistema de razões com o objetivo de encontrar os valores desconhecidos a e b.

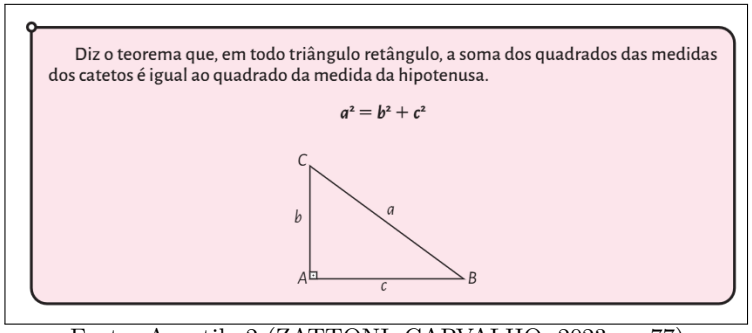
Figura 48 – Resolução em Python do exercício 8

Fonte: Elaborado pelo autor

5.4 Teorema de Pitágoras

O Teorema de Pitágoras é um dos pilares fundamentais da matemática, especialmente no campo da geometria, e afirma que, em um triângulo retângulo, o quadrado da hipotenusa é igual à soma dos quadrados dos catetos. Matematicamente, o teorema é expresso como mostra a Figura 49

Figura 49 – Teorema de Pitágoras



Fonte: Apostila 2 (ZATTONI; CARVALHO, 2023, p.77)

Onde a é a hipotenusa e b e c são os catetos do triângulo retângulo.

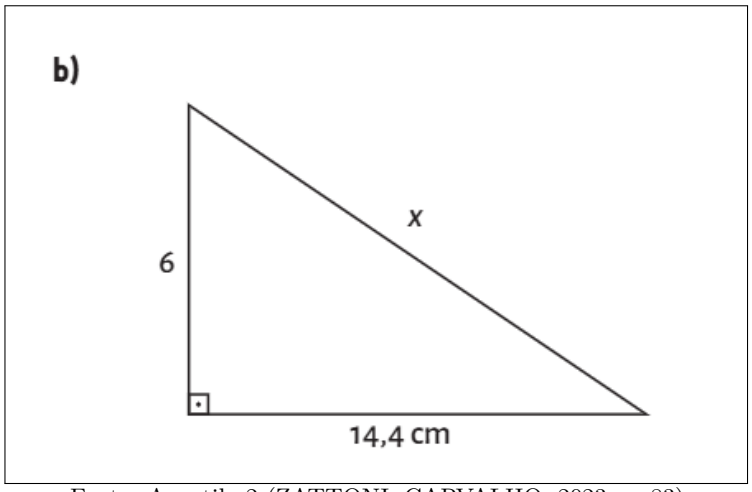
> Considerado um dos mais importantes teoremas da Matemática, consiste em uma relação entre os lados de um triângulo retângulo. Recebe esse nome por ter sido demonstrado primeiro pelo filósofo e matemático Pitágoras (570 a.C. – 497 a.C.).
>
> Há relatos históricos de que babilônios, chineses e egípcios conheciam e utilizavam o teorema muitos anos antes de Pitágoras. Da Antiguidade até o século XX, várias demonstrações foram publicadas e todas confirmam de maneiras diferentes sua validade (ZATTONI; CARVALHO, 2023, p.77).

O teorema é muitas vezes ilustrado através de exemplos simples, como calcular a diagonal de um quadrado ou determinar a distância entre dois pontos no plano cartesiano. Essas aplicações práticas evidenciam a simplicidade e a profundidade do conceito, que se mantém atual e relevante mesmo milhares de anos após sua descoberta.

5.4.1 Exercício 9

A Figura 50 apresenta um triângulo retângulo em que se deseja calcular a hipotenusa x, conhecendo-se os catetos 6 e 14,4 cm. A aplicação direta do Teorema de Pitágoras permite a resolução do problema por meio da relação. Esse tipo de exercício reforça a compreensão da relação fundamental entre os lados de um triângulo retângulo.

Figura 50 – Exercício 9



Fonte: Apostila 2 (ZATTONI; CARVALHO, 2023, p.83)

Passo a passo para resolver de maneira tradicional:

1. Calculando a hipotenusa e o perímetro de um triângulo retângulo com base nos valores dos seus catetos usando o Teorema de Pitágoras. Os valores fornecidos são:

- Cateto $a = 6$ cm
- Cateto $b = 14.4$ cm

2. De acordo com o Teorema de Pitágoras, a hipotenusa x é dada pela fórmula:

$$x = \sqrt{a^2 + b^2}$$

3. Substituindo os valores, tem-se:

$$x = \sqrt{6^2 + 14.4^2} = \sqrt{36 + 207.36} \approx \sqrt{243.36} \approx 15.6 \text{ cm}$$

4.Além disso, o perímetro do triângulo é calculado pela soma dos lados:

$$\text{Perímetro} = a + b + x = 6 + 14.4 + 15.6 = 36 \text{ cm}$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
    import math

# Definindo os lados do triângulo
a = 6  # um cateto
b = 14.4  # outro cateto

# Calculando a hipotenusa (x) usando o Teorema de Pitágoras
x = math.sqrt(a**2 + b**2)

# Calculando o perímetro do triângulo
perimetro = a + b + x

# Exibindo os resultados
print(f"O valor de x (hipotenusa) é: {x:.1f} cm")
print(f"O perímetro do triângulo é: {perimetro:.1f} cm")
```

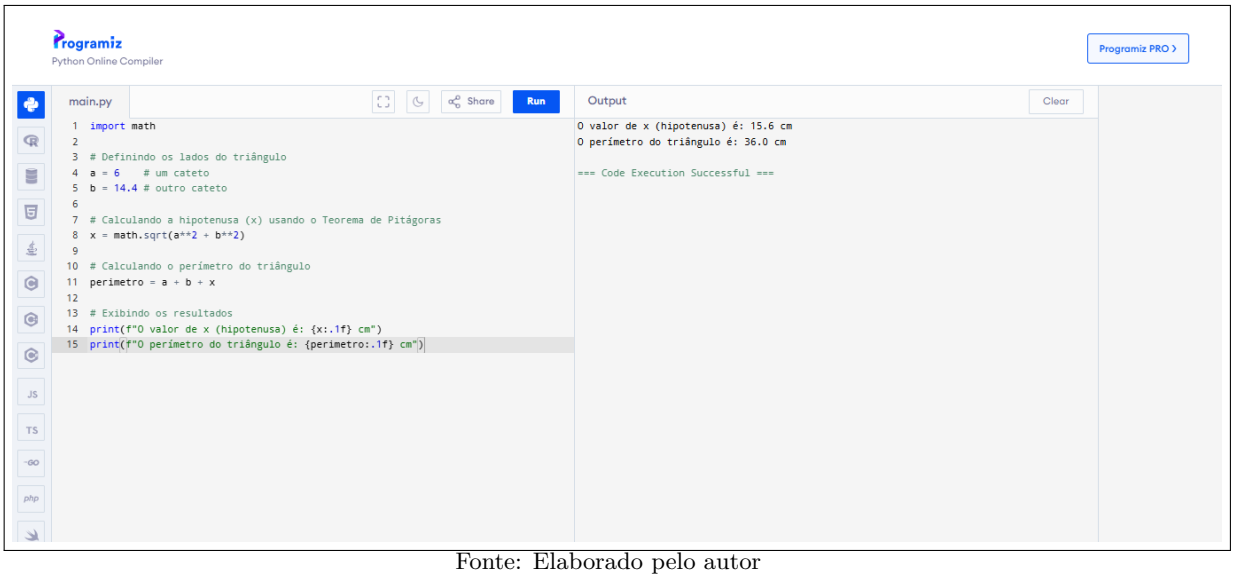
A execução do código resultou nos seguintes valores:

- Hipotenusa $x \approx 15.6$ cm
- Perímetro ≈ 36.0 cm

Dessa forma, verifica-se a aplicação prática do Teorema de Pitágoras e o cálculo do perímetro utilizando os valores dos lados do triângulo.

A Figura 51 demonstra a aplicação computacional do clássico teorema pitagórico utilizando a linguagem Python. Observa-se que os catetos $a = 6$ cm e $b = 14{,}4$ cm são inseridos no programa, e a hipotenusa x é calculada por meio da função `math.sqrt`, que computa a raiz quadrada da soma dos quadrados dos catetos.

Figura 51 – Resolução em Python do exercício 9



Fonte: Elaborado pelo autor

Na Figura 51 percebe-se que o programa também calcula o perímetro do triângulo, evidenciando a possibilidade de ampliar o raciocínio matemático com recursos computacionais. Essa integração entre matemática e programação favorece o desenvolvimento da cultura digital no contexto da educação matemática, promovendo não apenas a resolução de problemas, mas também a interpretação e comunicação de resultados de maneira precisa e tecnológica.

Portanto, tem-se que a hipotenusa do triângulo mede aproximadamente $15,6$ cm e o perímetro do triângulo é 36 cm. Este exemplo ilustra a aplicação prática do Teorema de Pitágoras e o cálculo do perímetro em um triângulo retângulo.

5.5 Trigonometria

A trigonometria é uma das áreas mais fascinantes da matemática, dedicada ao estudo das relações entre os ângulos e os lados dos triângulos.

Os conceitos básicos da trigonometria estão relacionados aos triângulos retângulos, em que são definidas as razões trigonométricas fundamentais: seno, cosseno e tangente. Para um ângulo θ em um triângulo retângulo, tem-se as seguintes definições:

- $\text{seno}(\theta) = \dfrac{\text{cateto oposto}}{\text{hipotenusa}}$
- $\text{cosseno}(\theta) = \dfrac{\text{cateto adjacente}}{\text{hipotenusa}}$
- $\text{tangente}(\theta) = \dfrac{\text{cateto oposto}}{\text{cateto adjacente}}$

Essas relações são conhecidas como relações trigonométricas no triângulo retângulo e fazem parte da Trigonometria, um ramo da Matemática que

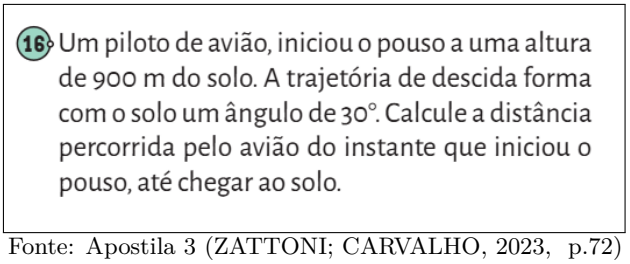
> trata dos métodos de cálculo em triângulos. A Trigonometria teve grande importância nas Grandes Navegações (auxiliando os navegadores a se orientarem por meio da observação do céu noturno), além de ser utilizada na Astronomia, na Física e em diversas outras áreas do conhecimento. (ZATTONI; CARVALHO, 2023, p.55)

Dessa forma, a trigonometria não é apenas um conjunto de regras e fórmulas, mas uma ponte para o entendimento das relações que governam o mundo em que se vive.

5.5.1 Exercício 10

A Figura 43 apresenta uma situação problema que mobiliza o raciocínio trigonométrico a partir de um contexto realista e significativo.

Figura 52 – Exercício 10



Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.72)

Passo a passo para resolver de maneira tradicional:

1.Um piloto de avião inicia o pouso a partir de uma altura de 900 m do solo. A trajetória de descida do avião forma um ângulo de 30° com o solo. Nosso objetivo é calcular a distância percorrida pelo avião, que corresponde à hipotenusa do triângulo formado pela altura e a trajetória de descida.

2.Para resolver o problema, utiliza-se a relação trigonométrica do seno, definida como:

$$\sin(\theta) = \frac{\text{cateto oposto}}{\text{hipotenusa}}$$

3.Substituindo os valores fornecidos:

$$\sin(30^\circ) = \frac{900}{x}$$

4.Sabe-se que $\sin(30^\circ) = 0,5$. Portanto:

$$0,5 = \frac{900}{x}$$

5.Multiplicando ambos os lados por x e isolando x, tem-se:

$$x = \frac{900}{0,5} = 1800\,\text{m}$$

Portanto, a distância percorrida pelo avião desde o início do pouso até tocar o solo é de 1800 m. Este exemplo ilustra como as funções trigonométricas podem ser aplicadas para calcular distâncias em situações práticas envolvendo trajetórias e ângulos.

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
import math

# Dados
altura = 900  # altura em metros
angulo = 30  # ângulo em graus

# Convertendo o ângulo de graus para radianos
angulo_rad = math.radians(angulo)

# Calculando a hipotenusa (distância percorrida pelo avião)
hipotenusa = altura / math.sin(angulo_rad)

# Exibindo o resultado
print(f"A distância percorrida pelo avião até chegar ao solo é:
{hipotenusa:.2f} m")
```

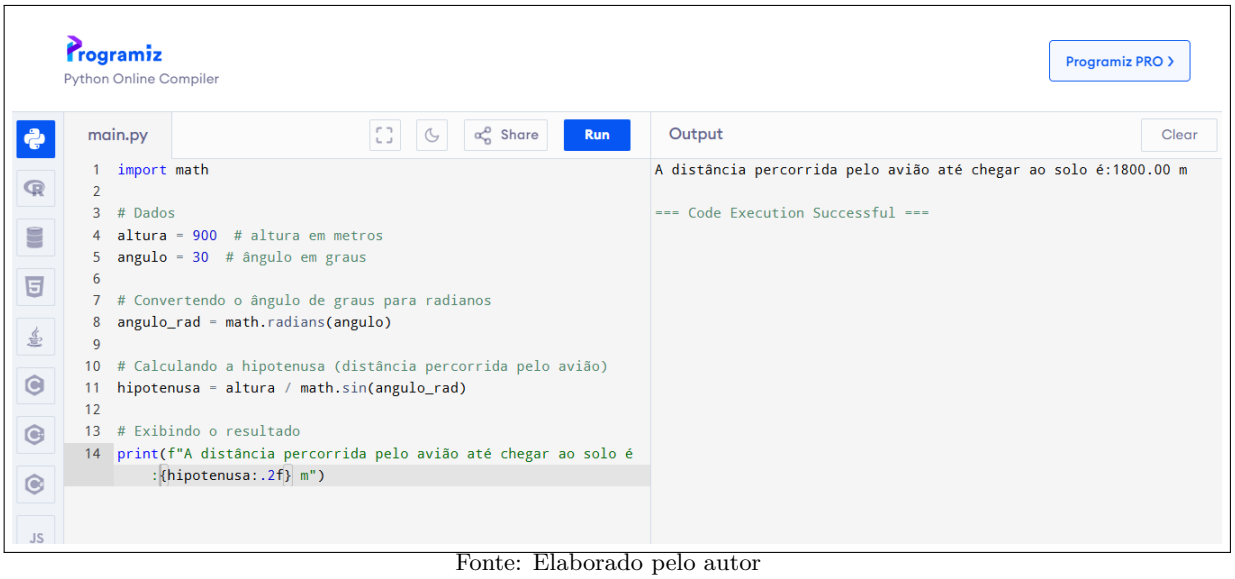
O resultado obtido pela execução do código é:

- A distância percorrida pelo avião até o solo é aproximadamente 1800.00 m.

Dessa forma, pode-se concluir que a aplicação de funções trigonométricas permite calcular de maneira eficiente a trajetória percorrida em situações práticas, como o movimento de um avião.

A Figura 53 ilustra a aplicação computacional da linguagem de programação Python para resolver um problema contextualizado de trigonometria. Neste caso, o uso do compilador Programiz permite automatizar os cálculos, promovendo uma abordagem algorítmica da resolução matemática.

Figura 53 – Resolução em Python do exercício 10



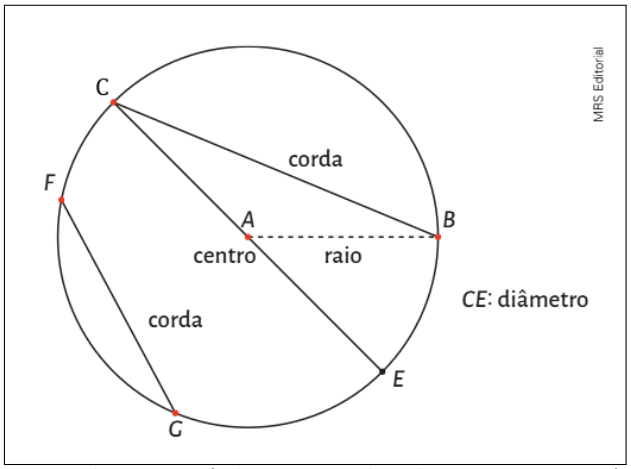
Fonte: Elaborado pelo autor

Na Figura 53 o código implementado mostra a conversão do ângulo de graus para radianos (uma exigência das funções trigonométricas da biblioteca math do Python), seguida do cálculo da hipotenusa do triângulo retângulo formado pela trajetória de descida do avião. Tal resolução evidencia, de forma clara, como a tecnologia pode ser uma aliada no ensino da Matemática, ampliando a compreensão conceitual e operatória dos estudantes.

5.6 Relações Métricas na Circunferência

As relações métricas na circunferência dizem respeito às fórmulas que envolvem comprimentos e segmentos determinados por retas notáveis que interceptam a circunferência. Esses conceitos são fundamentais para a resolução de problemas geométricos e aparecem com frequência em contextos práticos e teóricos na Figura 54 apresenta os elementos de uma circunferência.

Figura 54 – Elementos da circunferência



Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.76)

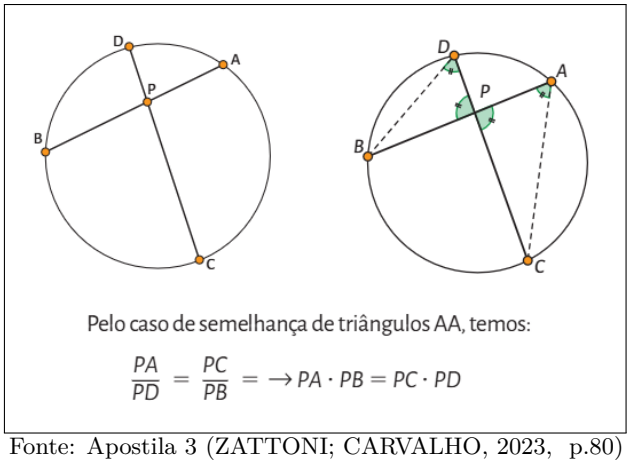
Segmentos Notáveis na Circunferência

- **Corda**: segmento cujas extremidades pertencem à circunferência.
- **Diâmetro**: corda que passa pelo centro da circunferência.
- **Raio**: segmento que liga o centro da circunferência a um ponto da borda.

Relação entre Duas Cordas que se Cruzam Dentro da Circunferência

Na Figura 55 é mostra que se duas cordas se cruzam dentro da circunferência, formando os segmentos A, B, C, D e P, o produto das partes de uma corda é igual ao produto das partes da outra.

Figura 55 – Relação entre Duas Cordas que se Cruzam Dentro da Circunferência



Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.80)

Relação entre Secante

Quando duas retas secantes se encontram fora de uma circunferência, elas cortam a circunferência em dois pontos cada. A partir desse ponto externo, cada secante forma dois segmentos:

- Um **segmento externo** (do ponto até onde ela toca a circunferência),
- E um **segmento interno** (dentro da circunferência, entre os dois pontos de interseção).

A relação importante é:

segmento externo×segmento total = segmento externo da outra secante×segmento total da outra

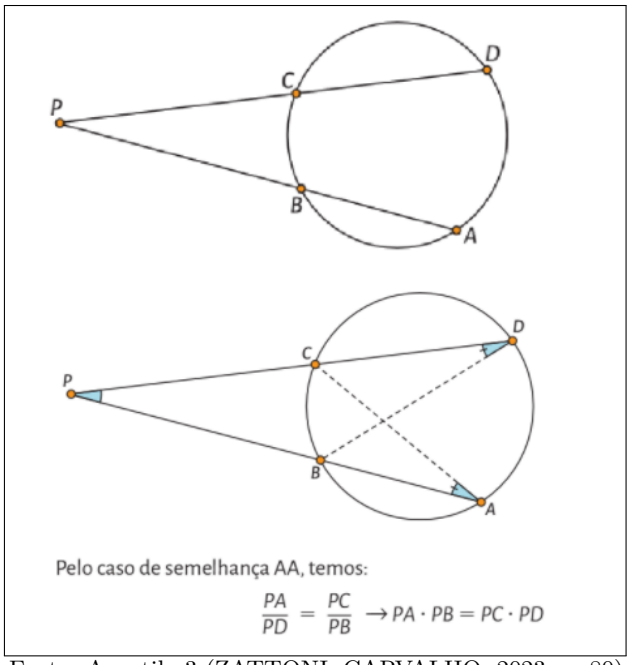
Ou seja:

$$PA \cdot PB = PC \cdot PD$$

"Duas cordas secantes são concorrentes em um ponto exterior de uma circunferência, então o produto das medidas de um segmento secante e sua parte externa é igual ao produto do outro segmento secante e sua parte externa"(ZATTONI; CARVALHO, 2023, p.81).

Na Figura 56 apresenta retas secantes que cortam a circunferência em dois pontos. Quando duas secantes se cruzam fora da circunferência, formam segmentos, e existe uma relação entre esses segmentos.

Figura 56 – Relação entre Duas Cordas secantes



Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.80)

Relação entre Secante e tangente

Se uma reta secante e uma reta tangente se encontram em um mesmo ponto externo P, então:

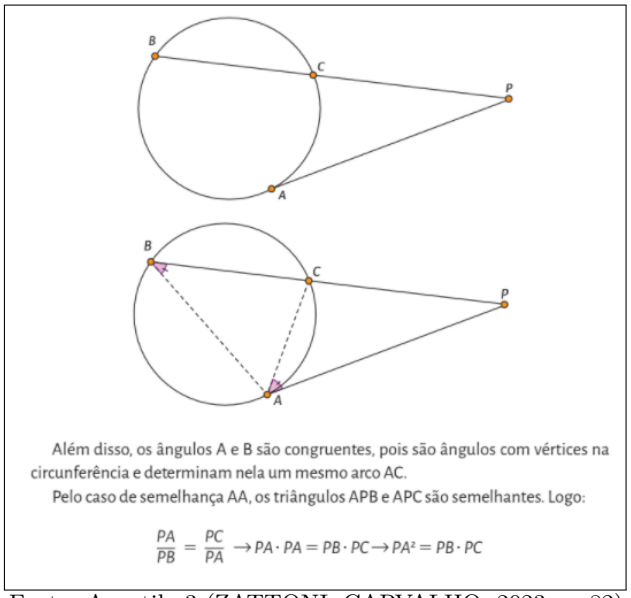
$$PA^2 = PB \cdot PC$$

- PA é o comprimento do segmento tangente;
- PB é o comprimento total da secante (parte externa + parte interna).
- PC é o segmento externo da secante;

"Se uma corda secante e uma tangente à circunferência são concorrentes em um ponto externo P, então o quadrado da medida do segmento tangente à circunferência é igual ao produto da corda secante e sua parte externa"(ZATTONI; CARVALHO, 2023, p.81).

A Figura 57 mostra a relação entre uma reta secante e uma tangente à circunferência que se encontram em um ponto externo. A relação métrica indica que o quadrado da medida da tangente é igual ao produto do segmento externo da secante pelo seu comprimento total.

Figura 57 – Relação entre Duas Cordas secantes



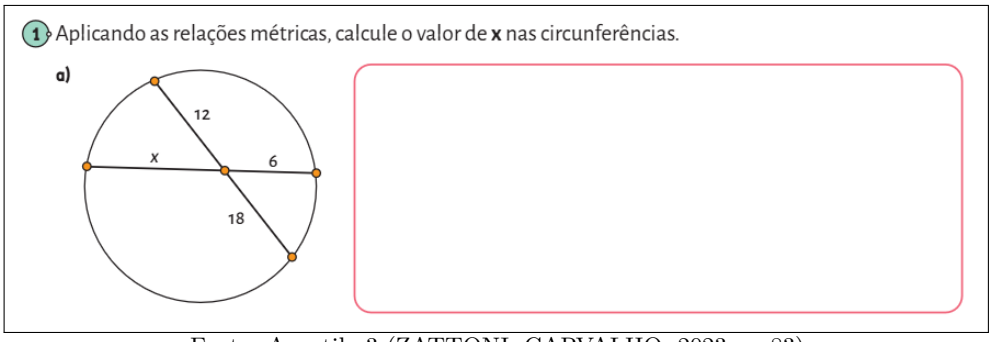
Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.82)

Relação entre Duas Secantes que se Cruzam Fora da Circunferência

5.6.1 Exercício 11

A Figura 58 apresenta uma situação geométrica envolvendo cordas que se cruzam no interior da circunferência. Esse tipo de configuração permite a aplicação de uma importante propriedade da geometria plana: o produto das medidas dos segmentos de cordas que se intersectam em um ponto interno da circunferência é constante, ou seja, se duas cordas se cruzam em um ponto, o produto das partes de uma corda é igual ao produto das partes da outra.

Figura 58 – Exercício 11



Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.83)

Passo a passo para resolver de maneira tradicional:

Aplicando as relações métricas na circunferência para determinar o valor de x.

1. Identificação da Relação Métrica: Quando duas cordas se intersectam no interior da circunferência, a relação entre os segmentos é dada por:

Produto dos segmentos de uma corda = Produto dos segmentos da outra corda.

Assim, no problema:
- Uma corda tem segmentos 12 e 6.
- A outra corda tem segmentos x e 18.

2. Equação da Relação Métrica: Pode-se escrever:

$$6x = 12 \cdot 18.$$

3. Resolução da Equação: Substituindo os valores:

$$6x = 216.$$

Dividindo ambos os lados por 6:

$$x = \frac{216}{6}.$$

Resolvendo:

$$x = 36.$$

4. Resultado Final: O valor de x é: 36.

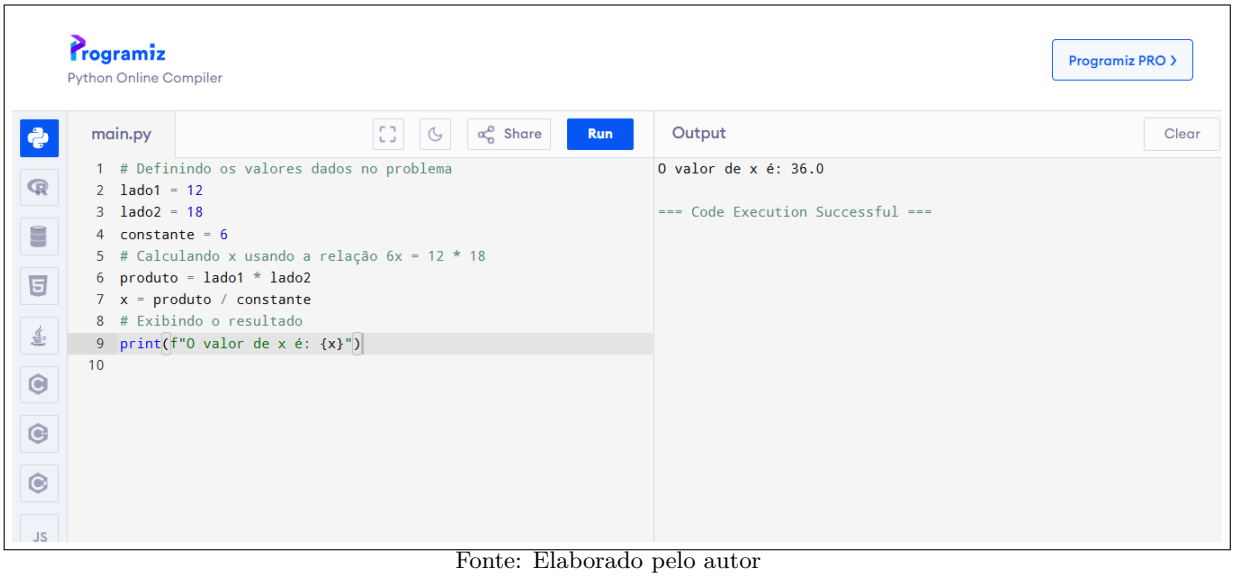
A sugestão para resolução em Python pode ser feita da seguinte forma:

```
    # Definindo os valores dados no problema
lado1 = 12
lado2 = 18
constante = 6
```

```
# Calculando x usando a relação 6x = 12 * 18
produto = lado1 * lado2
x = produto / constante

# Exibindo o resultado
print(f"O valor de x é: {x}")
```

A Figura 59 apresenta o uso da linguagem Python como ferramenta para resolver um problema envolvendo relações métricas na circunferência. O código implementado no ambiente Programiz aplica a propriedade que relaciona os segmentos de cordas que se cruzam no interior da circunferência. Através da definição de variáveis e de um simples cálculo algébrico, o valor de x é obtido de forma prática.

Figura 59 – Resolução em Python do exercício 11

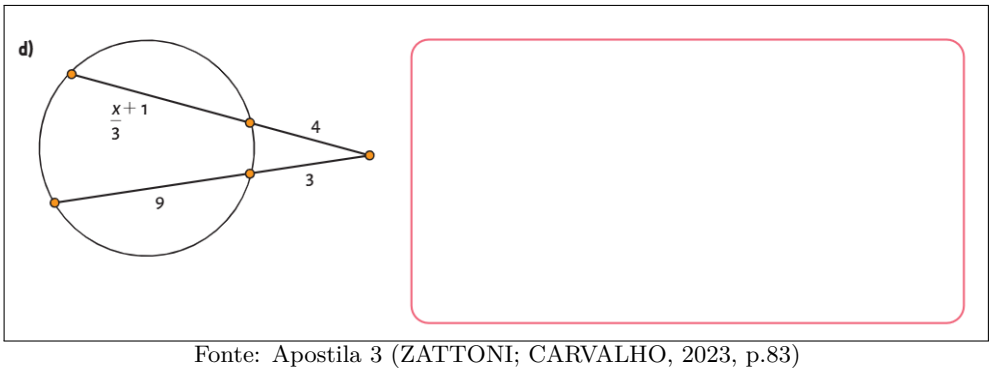


Fonte: Elaborado pelo autor

5.6.2 Exercício 12

Figura 60 apresenta uma configuração clássica de segmentos secantes que se cruzam fora de uma circunferência, permitindo a aplicação das relações métricas da geometria. Esses tipos de exercícios são comuns no estudo das propriedades das circunferências e envolvem o uso do Teorema da Potência de um Ponto.

Figura 60 – Exercício 12



Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.83)

Na Figura 60 tem-se segmentos que se intersectam em um ponto externo à circunferência, e os comprimentos indicados possibilitam a montagem de uma equação para encontrar o valor de uma incógnita.

Passo a passo para resolver de maneira tradicional:

Neste exercício, aplica-se o que estabelece uma relação métrica entre duas secantes traçadas a partir de um ponto exterior a uma circunferência.

O produto entre o segmento externo e o comprimento total da secante é o mesmo para ambas as secantes.

1. A partir da figura, identifica-se os seguintes elementos:

- Primeira secante (superior):
  - Segmento externo: 4
  - Segmento interno: $\frac{x}{3} + 1$
  - Segmento total: $4 + \left(\frac{x}{3} + 1\right)$
- Segunda secante (inferior):
  - Segmento externo: 3
  - Segmento total: $3 + 9 = 12$

2. Aplicando a relação da potência do ponto:

$$4 \cdot \left(4 + \left(\frac{x}{3} + 1\right)\right) = 3 \cdot 12$$

$$4 \cdot \left(\frac{x}{3} + 5\right) = 36$$

$$\frac{4x}{3} + 20 = 36$$

$$\frac{4x}{3} = 16$$

$$4x = 48 \Rightarrow x = \frac{48}{4} = 12$$

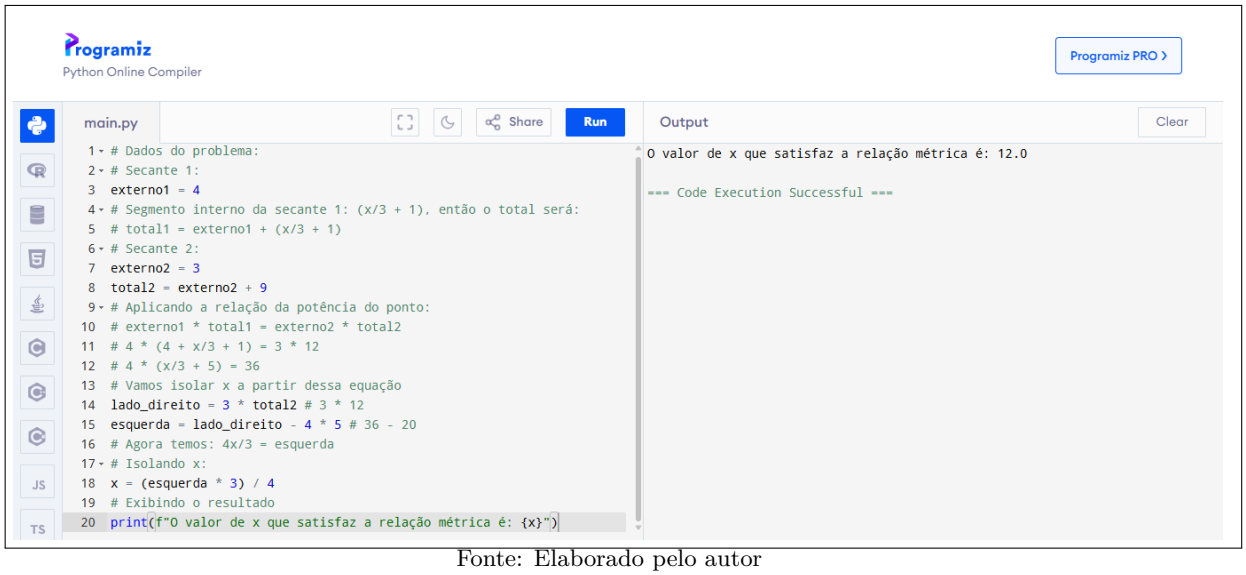
3. Resultado Final: O valor de x é: 12

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
# Dados do problema:
# Secante 1:
externo1 = 4
# Segmento interno da secante 1: (x/3 + 1), então o total será:
# total1 = externo1 + (x/3 + 1)

# Secante 2:
externo2 = 3
total2 = externo2 + 9

# Aplicando a relação da potência do ponto:
# externo1 * total1 = externo2 * total2
# 4 * (4 + x/3 + 1) = 3 * 12
# 4 * (x/3 + 5) = 36

# Isolando x a partir dessa equação
lado_direito = 3 * total2  # 3 * 12
esquerda = lado_direito - 4 * 5  # 36 - 20
# Agora tem-se: 4x/3 = esquerda
# Isolando x:
x = (esquerda * 3) / 4

# Exibindo o resultado
print(f"O valor de x que satisfaz a relação métrica é: {x}")
```

A Figura 61 demonstra a aplicação da linguagem Python na resolução de um problema envolvendo tangentes e secantes em uma circunferência. Utilizando o ambiente *online* Programiz, o código simula a resolução algébrica da equação gerada pela propriedade: o quadrado da medida da tangente é igual ao produto da secante total pela sua parte externa.

Figura 61 – Resolução em Python do exercício 12

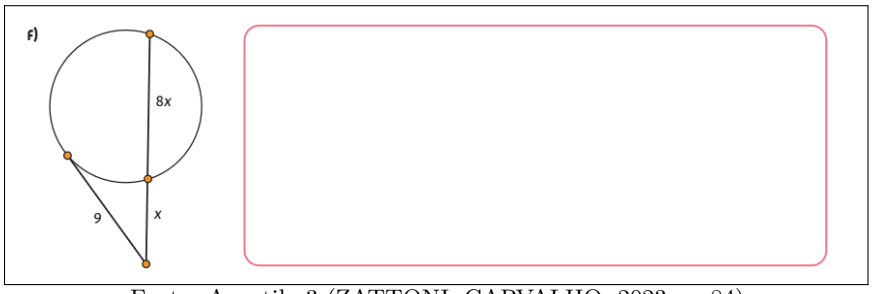


Fonte: Elaborado pelo autor

5.6.3 Exercício 13

A Figura 62 apresenta uma situação que envolve a relação entre a tangente e a secante traçadas a partir de um ponto externo à circunferência. Segundo a propriedade métrica aplicada nesse caso, o quadrado da medida da tangente é igual ao produto da secante total pela sua parte externa.

Figura 62 – Exercício 13



Fonte: Apostila 3 (ZATTONI; CARVALHO, 2023, p.84)

Passo a passo para resolver de maneira tradicional:

Determinando o valor de x aplicando a "relação da potência de um ponto" na circunferência.

1. Relação da Potência de um Ponto: Quando uma secante e uma tangente são traçadas a partir de um ponto exterior à circunferência, tem-se:

$$\text{Produto da secante e sua parte externa} = (\text{tangente})^2.$$

2. Aplicando ao Problema: A secante e a tangente dadas no problema são: - Secante: comprimento total $9 + x$, com parte externa x. - Tangente: comprimento 9.

Pela relação da potência do ponto:

$$x \cdot (9 + x) = 9^2.$$

3. Montagem da Equação: Expandindo a equação:

$$x \cdot 9 + x^2 = 81.$$

Reorganiza-se para a forma padrão:

$$x^2 + 9x - 81 = 0.$$

4. Resolução da Equação Quadrática: Utilizando a fórmula de Bhaskara:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

Substituindo os coeficientes $a = 1$, $b = 9$ e $c = -81$:

$$x = \frac{-9 \pm \sqrt{9^2 - 4 \cdot 1 \cdot (-81)}}{2 \cdot 1}.$$

Calculando:

$$x = \frac{-9 \pm \sqrt{81 + 324}}{2}.$$

$$x = \frac{-9 \pm \sqrt{405}}{2}.$$

Aproximando a raiz:

$$\sqrt{405} \approx 20.12.$$

Assim:

$$x = \frac{-9 + 20.12}{2} \quad \text{(descarta-se a raiz negativa pois } x > 0\text{)}.$$

$$x \approx \frac{11.12}{2}.$$

$$x \approx 5.56.$$

5. Resposta Final: O valor aproximado de x é:

3.

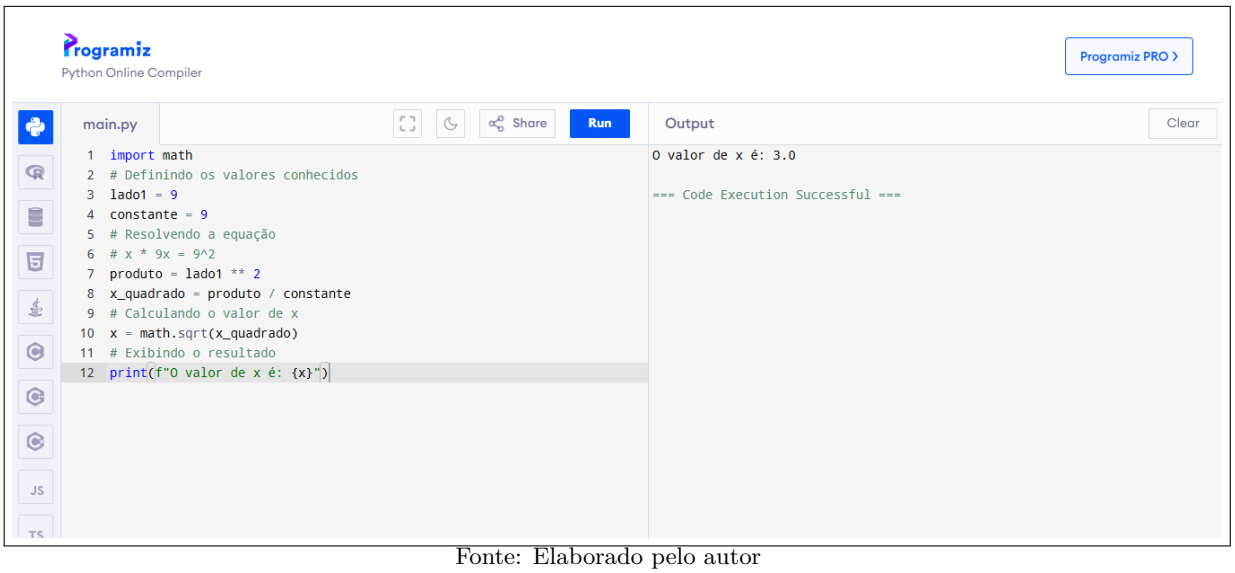
A sugestão para resolução em Python pode ser feita da seguinte forma:

```
    import math

# Definindo os valores conhecidos
lado1 = 9
constante = 9
```

```
# Resolvendo a equação
# x * 9x = 9^2
produto = lado1 ** 2
x_quadrado = produto / constante

# Calculando o valor de x
x = math.sqrt(x_quadrado)

# Exibindo o resultado
print(f"O valor de x é: {x}")
```

A Figura 63 ilustra a utilização da linguagem de programação Python como ferramenta para resolver uma equação envolvendo relações métricas na circunferência. O problema apresentado parte da propriedade onde o quadrado da tangente é igual ao produto da secante total pela sua parte externa.

Figura 63 – Resolução em Python do exercício 13

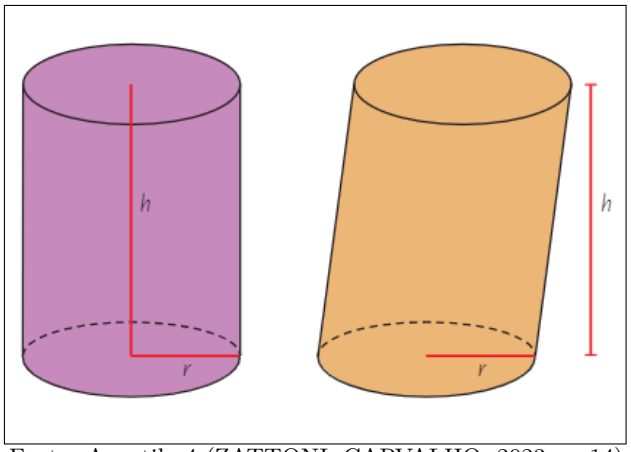


Fonte: Elaborado pelo autor

5.7 Cilindros: Volume

O cilindro é um sólido geométrico que possui duas bases circulares e paralelas unidas por uma superfície lateral curva. Ele pode ser classificado como um sólido de revolução, pois pode ser obtido pela rotação de um retângulo em torno de um de seus lados como apresentado na Figura 64.

Figura 64 – Cilindro



Fonte: Apostila 4 (ZATTONI; CARVALHO, 2023, p.14)

Elementos do Cilindro

- **Base**: cada uma das duas regiões circulares e paralelas.
- **Altura** (h): distância entre as bases.
- **Raio** (r): raio da base circular.

Volume do Cilindro

O volume do cilindro é calculado multiplicando a área da base pela altura:

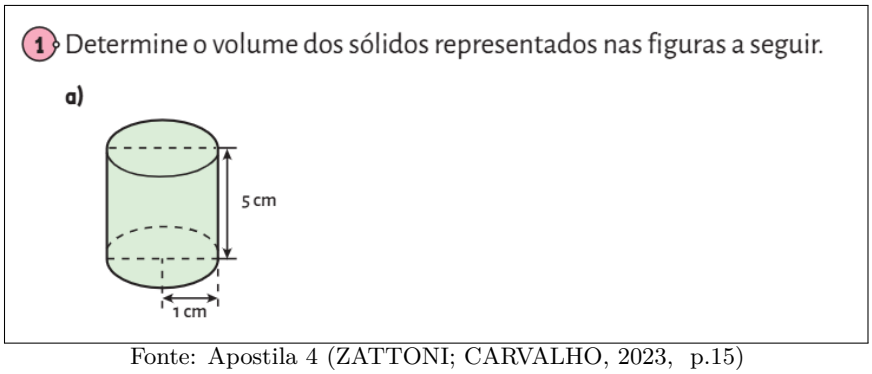
$$V = \pi r^2 h$$

Essa fórmula mostra que o volume do cilindro depende diretamente do tamanho da base circular e da altura do sólido.

5.7.1 Exercício 14

A Figura 65 apresenta um sólido geométrico do tipo **cilindro circular reto**, no qual são fornecidas as medidas do raio da base (1 cm) e da altura (5 cm). O objetivo do exercício é aplicar a fórmula do volume do cilindro, dada por $V = \pi r^2 h$, para calcular sua capacidade.

Figura 65 – Exercício 14



Fonte: Apostila 4 (ZATTONI; CARVALHO, 2023, p.15)

Passo a passo para resolver de maneira tradicional:

Para determinar o volume do cilindro representado na Figura 56.

1. Fórmula do Volume do Cilindro: O volume V de um cilindro é dado por:

$$V = \pi \cdot r^2 \cdot h,$$

onde: - r é o raio da base, - h é a altura do cilindro.

2. Substituição dos Valores: Pelo enunciado: - O raio $r = 1\,\text{cm}$, - A altura $h = 5\,\text{cm}$.

Substituí-se na fórmula:

$$V = \pi \cdot (1)^2 \cdot 5.$$

3. Simplificação: Calculando:

$$V = \pi \cdot 1 \cdot 5.$$

$$V = 5\pi\,\text{cm}^3.$$

4. Resultado Final: O volume do cilindro é:

$$5\pi\,\text{cm}^3.$$

A sugestão para resolução em Python pode ser feita da seguinte forma:

```
# Dados do problema
raio = 1  # Raio da base em cm
altura = 5  # Altura do cilindro em cm

# Cálculo do volume simbolicamente com \u03C0]
volume = f"5\u03C0"  # O valor calculado é 5 *\pi

# Exibindo o resultado
print(f"O volume do cilindro é: {volume} cm³")
```

Na Figura 66 para escrever a letra π no Programiz (ou em qualquer ambiente Python), você pode usar o código *Unicode* (um padrão que permite a representação e manipulação de texto de qualquer sistema de escrita, usando códigos para caracteres individuais) diretamente no texto. O símbolo de π é inserido com `\u03C0` dentro de uma *strings*.

Figura 66 – Resolução em Python com símbolo π do exercício 14

Fonte: Elaborado pelo autor

Caso sem pi

```
    import math

# Dados do problema
raio = 1  # em cm
altura = 5  # em cm

# Fórmula do volume do cilindro: V = pi * r² * h
volume = math.pi * (raio ** 2) * altura

# Exibindo o resultado
print(f"O volume do cilindro é: {volume:.2f} cm³")
```

Na Figura 67 a fórmula usada para o cálculo do volume é $V = \pi r^2 h$, e neste caso, a constante π foi aproximada por 3,14. O código define as variáveis necessárias, executa o cálculo e exibe o resultado final de 15,7 cm^3. A utilização da programação nesse contexto permite uma abordagem mais prática e interdisciplinar, unindo conceitos da matemática e da computação de forma dinâmica e significativa.

Figura 67 – Resolução em Python com aproximação π do exercício 14

Fonte: Elaborado pelo autor

Os exercícios mencionados nessa dissertação do material estruturado do governo do estado de Mato Grosso foram escolhidos com intuito de explorar a linguagem Python por meio do compilador *online* Programiz. Ao trabalhar com potências, raízes, equações do segundo grau, propriedades de triângulos, relações métricas na circunferência e volume de cilindros, faz com que os estudantes não apenas entendam os conceitos, mas também desenvolvam habilidades para resolver problemas com linguagem computacional.

Além disso, explorar assuntos matemático de outra forma que não seja a tradicional acredita-se que alunos tenham mais interação e dinâmica, integrando conhecimentos de outras áreas e estimulando a criatividade, essencial no mundo atual, onde a tecnologia é cada vez mais presente.

6 Considerações Finais

A presente proposta didática tem como eixo central a integração da linguagem de programação Python ao ensino de conceitos matemáticos, articulando-se com o material estruturado do governo do Estado de Mato Grosso. A iniciativa propõe não apenas um método alternativo de resolução de exercícios, mas uma abordagem inovadora, que pretende transformar a experiência dos estudantes ao aprender matemática por meio da tecnologia computacional.

O propósito é dinamizar o ambiente de sala de aula e torná-lo mais interativo, utilizando desafios que instiguem o raciocínio lógico e matemático, ao mesmo tempo em que promovam a familiarização com recursos digitais. Ao incorporar a programação, almeja-se estabelecer um espaço onde os conceitos matemáticos tradicionalmente apresentados de forma abstrata sejam explorados de maneira prática, interativa e visual. Por meio da experimentação e da observação dos resultados gerados pelas suas codificações em tempo real, os alunos poderão desenvolver um entendimento mais profundo e significativo dos conceitos matemáticos trabalhados e da linguagem de programação.

Contudo, a implementação de metodologias que envolvem a tecnologia computacional também impõe desafios consideráveis. Entre eles, destaca-se a limitação no acesso a recursos tecnológicos adequados, como *chromebook*, rede de internet estável, além da necessária capacitação docente para o uso pedagógico de linguagens de programação. Torna-se imprescindível, nesse sentido, que as instituições de ensino e o poder público invistam na melhoria da infraestrutura tecnológica e no desenvolvimento profissional dos professores, para que se estabeleça uma cultura escolar capaz de sustentar práticas pedagógicas inovadoras.

Além das questões de infraestrutura, esta proposta sublinha a importância de uma abordagem pedagógica sólida para a integração eficaz da tecnologia. Métodos que combinem elementos do ensino tradicional com as possibilidades oferecidas por ferramentas tecnológicas podem criar ambientes de aprendizado mais inclusivos e adaptáveis. A flexibilização dos currículos para incluir a programação permitiria que os alunos aprofundassem seu aprendizado de forma mais autônoma, resultando em uma compreensão mais robusta dos conteúdos.

Em uma perspectiva mais abrangente, essa proposta contribui para o debate contínuo sobre o papel das tecnologias educacionais na formação de alunos para um futuro digital. Ao demonstrar que a programação pode ser integrada com êxito em disciplinas como a matemática, ela oferece um modelo potencial para a interdisciplinaridade no aprendizado e para a inovação em outros contextos curriculares.

A escolha pelo compilador *online* Programiz como ferramenta de apoio à proposta se justifica pela sua interface acessível, pelo uso intuitivo e pela facilidade de acesso em dispositivos como chromebooks, dispensando a instalação de programas específicos. Isso reduz consideravelmente as barreiras iniciais e possibilita que os estudantes experimentem a programação de maneira imediata e descomplicada, tornando o processo de aprendizado mais fluido e estimulante. A simplicidade da linguagem Python e a interface intuitiva do Programiz criam um ambiente inovador, reduzindo a intimidação e facilitando a inserção dos jovens estudantes nesse universo.

Em suma, a integração do Python no ensino de conceitos matemáticos, conforme proposto, não apenas enriquece a educação dessa disciplina, mas também fomenta o desenvolvimento de alunos que gostam e se interessam por tecnologias. O potencial da proposta destaca a necessidade de investimento contínuo em tecnologias educacionais e na capacitação de professores e alunos. Embora os desafios sejam evidentes, os benefícios previstos reforçam a importância de investir continuamente na construção de ambientes escolares tecnologicamente equipados e pedagogicamente atualizados, capazes de acolher e potencializar novas práticas de ensino.

Referências

BRASIL. *Ministério da Educação: Base Nacional Comum Curricular*. Brasília, 2017. Acesso em: 20 ago. 2024. Disponível em: <http://basenacionalcomum.mec.gov.br/>. Citado 3 vezes nas páginas 16, 17 e 19.

BRASIL. *Aprovado parecer que define normas sobre o ensino de computação na educação básica*. 2022. Acesso em: 15 out. 2024. Disponível em: <https://www.gov.br/pt-br/noticias/educacao-e-pesquisa/2022/10/aprovado-parecer-que-define-normas-sobre-o-ensino-de-computacao-na-educacao-basica>. Citado na página 20.

BRASIL. *Conselho Nacional de Educação. Resolução CNE/CEB nº 1, de 4 de outubro de 2022*. [S.l.]: MEC, 2022. Disponível em: <https://portal.mec.gov.br/docman/fevereiro-2022-pdf/236791-anexo-ao-parecer-cneceb-n-2-2022-bncc-computacao/file>. Acesso em: 1. set. 2024. Citado na página 21.

BRASIL, M. d. E. *ProInfo - Programa Nacional de Tecnologia Educacional*. 2024. Disponível online. Disponível em: <https://www.gov.br/fnde/pt-br/acesso-a-informacao/acoes-e-programas/programas/bolsas-e-auxilios/lista-de-programas/proinfo-programa-nacional-de-tecnologia-educacional>. Citado na página 20.

COSTA, W. *A História do Python*. Digital Innovation One (DIO), 2023. Acessado em: 20 nov. 2024. Disponível em: <https://www.dio.me/articles/a-historia-do-python>. Citado na página 23.

FOUNDATION, P. S. *Python Downloads*. 2024. <https://www.python.org/downloads/>. Acesso em: 6 dez. 2024. Citado na página 23.

LIMA, L. K. O. S.; SANTOS, E. M. d. As tecnologias digitais no contexto da pandemia: a capacitação de professores da educação básica. In: *Anais do Congresso Nacional de Educação – CONEDU*. Editora Realize, 2020. Disponível em: <https://editorarealize.com.br/editora/anais/conedu/2020/TRABALHO_EV140_MD4_SA19_ID5564_01092020220246.pdf>. Citado na página 19.

ROSA, A. *Estudantes e professores da rede estadual destacam que material estruturado otimiza método de estudos e aprendizagem*. 2022. Acesso em: 6 dez. 2024. Disponível em: <https://www3.seduc.mt.gov.br/-/18865697-apostilas-do-sistema-estruturado-comecam-a-ser-entregues-a-partir-da-proxima-semana-para-as-escolas-estaduais>. Citado na página 53.

ROSA, A. *Estudantes e professores da rede estadual destacam que material estruturado otimiza método de estudos e aprendizagem*. 2022. Acesso em: 6 dez. 2024. Disponível em: <https://www3.seduc.mt.gov.br/-/19220372-estudantes-e-professores-da-rede-estadual-destacam-que-material-estruturado-otimiza-metodo-de-estudos-e-aprendizagem>. Citado na página 53.

SECRETARIA DE EDUCAÇÃO DE PERNAMBUCO. *Currículo de Pernambuco – Ensino Médio*. 2021. <https://portal.educacao.pe.gov.br/ensino-medio/>. Acesso em: 06 dez. 2024. Citado na página 26.

SOUTO, D. L. P. *Transformações expansivas na produção matemática online*. UNESP, 2014. (Coleção PROPG Digital). Acesso em: 20 nov. 2024. Disponível em: <https://repositorio.unesp.br/server/api/core/bitstreams/f254f23f-476b-4b07-af6c-ba350ed8487b/content>. Citado na página 18.

VALENTE, J. A.; FREIRE, F. M. P.; ARANTES, F. L. *Tecnologia e Educação: passado, presente e o que está por vir*. Campinas, SP: NIED/UNICAMP, 2018. Acesso em: 20 nov. 2024. Disponível em: <https://odisseu.nied.unicamp.br/wp-content/uploads/2018/11/Livro-NIED-2018-final.pdf>. Citado 2 vezes nas páginas 20 e 21.

ZATTONI, R.; CARVALHO, T. D. *MAXI: 9º ano: ensino fundamental, anos finais: caderno 1: Matemática: Multidiciplinar*. 1. ed. São Paulo: Somos Sistemas de Ensino, 2023. Citado 23 vezes nas páginas 55, 56, 59, 61, 65, 67, 71, 73, 76, 77, 78, 80, 81, 84, 86, 87, 88, 89, 90, 92, 94, 97 e 98.