MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS GOIÂNIA
CURSO DE LICENCIATURA EM MATEMÁTICA

UMA NOVA PERSPECTIVA DO ENSINO DA ESTATÍSTICA DESCRITIVA NO ENSINO MÉDIO COM PROGRAMAÇÃO EM PYTHON

SABRINA RODRIGUES VIEIRA

GOIÂNIA - GOIÁS
2023

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[x] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico e Educacional - Tipo: _____	

Nome Completo do Autor: Sabrina Rodrigues Vieira
Matrícula: 20191010940030
Título do Trabalho: Uma nova perspectiva do ensino da estatística descritiva no ensino médio com programação em python.

Autorização - Marque uma das opções

1. (x) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Goiânia, 21/06/2023

Sabrina R. Vieira
_____
Assinatura do Autor e/ou Detentor dos Direitos Autorais

SABRINA RODRIGUES VIEIRA

UMA NOVA PERSPECTIVA DO ENSINO DA ESTATÍSTICA DESCRITIVA NO ENSINO MÉDIO COM PROGRAMAÇÃO EM PYTHON

Trabalho de Conclusão do Curso apresentado ao Curso de Licenciatura em Matemática do Instituto Federal de Goiás – IFG – Câmpus Goiânia, como requisito parcial para obtenção do título de Licenciado em Matemática.

Orientador: Prof. Dr. Márcio Dias de Lima

GOIÂNIA - GOIÁS
2023

V658n Vieira, Sabrina Rodrigues.
Uma nova perspectiva do ensino da estatística descritiva no ensino médio com programação em Python / Sabrina Rodrigues Vieira. – Goiânia: Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia, 2023.
84 f.: il.

Orientador: Prof. Dr. Márcio Dias de Lima.

TCC (Trabalho de Conclusão de Curso) – Licenciatura em Matemática, Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia.

1. Matemática - estatística descritiva. 2. Liguagem de programação de computador - Python. I. Lima, Márcio Dias de (orientador). II. Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia. III. Título.

CDD 519.5

Ficha catalográfica elaborada pelo Bibliotecário Alisson de Sousa Belthodo Santos CRB1/ 2.266
Biblioteca Professor Jorge Félix de Souza,
Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia.

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS GOIÂNIA

UMA PERSPECTIVA DO ENSINO DE ESTATÍSTICA DESCRITIVA NO ENSINO MÉDIO COM PROGRAMAÇÃO EM PYTHON

por

SABRINA RODRIGUES VIEIRA

Trabalho de Conclusão de Curso apresentado ao Colegiado do Curso de Licenciatura em Matemática do Instituto Federal de Educação, Ciência e Tecnologia de Goiás – Câmpus Goiânia, como requisito parcial para a obtenção do Diploma de Licenciada em Matemática.

Área de concentração: Matemática Aplicada

Orientador: Dr. Márcio Dias de Lima

Trabalho de Conclusão de Curso defendido e aprovado, em 31 de maio de 2023, pela banca examinadora constituída pelos seguintes membros:

(assinado eletronicamente)

Prof. Dr. Márcio Dias de Lima (Presidente da Banca/IFG/Câmpus Goiânia)

(assinado eletronicamente)

Profª. Drª. Aline Mota de Mesquita Assis (IFG/Câmpus Goiânia)

(assinado eletronicamente)

Prof. Me. Uender Barbosa de Souza (IFG/Câmpus Goiânia)

Documento assinado eletronicamente por:

- **Uender Barbosa de Souza, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 31/05/2023 16:19:06.
- **Aline Mota de Mesquita Assis, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 31/05/2023 16:18:52.
- **Marcio Dias de Lima, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 31/05/2023 16:18:17.

Este documento foi emitido pelo SUAP em 27/05/2023. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifg.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 412731
Código de Autenticação: c375c5c2ea

AGRADECIMENTOS

Gostaria de expressar minha sincera gratidão a todos vocês que me apoiaram durante toda a minha jornada acadêmica e que foram fundamentais para que eu pudesse concluir meu trabalho de conclusão.

Primeiramente, quero agradecer a Deus por me conceder a saúde, força e sabedoria necessária para superar os desafios que enfrentei ao longo do caminho. A ele, minha eterna gratidão.

Aos meus familiares, em especial ao meu pai, madrinha e avó, quero dedicar um reconhecimento especial. Sem o amor, o incentivo e o suporte incondicional que recebi de vocês, não teria sido possível chegar até aqui. Foram eles que me ajudaram em cada etapa do processo, me motivando a seguir em frente mesmo quando as coisas pareciam difíceis demais. Vocês foram importantes para que eu mantivesse o equilíbrio emocional e não desistisse dos meus objetivos.

Também quero agradecer aos meus amigos, que me acompanharam durante toda a jornada e me proporcionaram momentos de descontração, apoio e leveza.

Não posso deixar de mencionar o meu orientador e os professores que me acompanharam ao longo do curso. Suas orientações, ensinamentos e incentivos foram fundamentais para que eu pudesse realizar meu trabalho da melhor forma possível.

Por fim, agradeço à Instituto Federal de Goiás por fornecer a estrutura necessária para que eu pudesse desenvolver meu trabalho. A acessibilidade que encontrei aqui foi fundamental para minha evolução.

RESUMO

Este trabalho apresenta uma nova perspectiva do uso da programação em Python no âmbito do ensino da matemática e estatística descritiva para alunos do Ensino Médio. O objetivo deste estudo é apresentar a importância do ensino da programação em Python para alunos do Ensino Médio, sua aplicação no ensino da estatística descritiva, além de promover o desenvolvimento do pensamento crítico. O ensino de programação Python é ideal para a resolução de problemas matemáticos e científicos, pois é de fácil aprendizagem, tem sintaxe clara, bibliotecas eficientes para cálculos e visualizações matemáticas. A metodologia adotada neste trabalho é a de uso de problemas, que consiste em desenvolver habilidades e competências dos alunos. O trabalho propõe atividades que demonstram a viabilidade e eficácia da abordagem apresentada para o ensino de matemática e estatística com programação em Python. Concluímos que o uso de problemas contextualizados, aliada à programação em Python, é uma abordagem eficaz para o ensino de matemática e estatística para alunos do Ensino Médio, pois promove o desenvolvimento de habilidades cognitivas, teóricas e interpessoais.

Palavras-chave: Matemática; Estatística Descritiva; Programação; Python.

ABSTRACT

This work presents a new perspective on the use of Python programming in the teaching of mathematics and descriptive statistics for high school students. The objective of this study is to demonstrate the importance of teaching programming in Python to high school students, its application in teaching descriptive statistics, in addition to promoting the development of critical thinking. Teaching Python programming is ideal for solving mathematical and scientific problems, as it is easy to learn, has clear syntax, efficient libraries for calculations and mathematical visualizations. The methodology adopted in this work is problem-based learning, which consists of developing students' abilities and skills through problem solving. The work proposes activities that demonstrate the feasibility and effectiveness of the approach presented for teaching mathematics and statistics with programming in Python. Concluding that problem-based learning, combined with Python programming, is an effective approach to teaching mathematics and statistics to high school students, as it promotes the development of cognitive, theoretical and interpersonal skills.

Keywords: Mathematics; Descriptive statistics; Schedule; Python.

LISTA DE CÓDIGOS

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

A programação é uma das habilidades tecnológicas mais importantes que os alunos podem aprender, ajudando-os a desenvolver outras habilidades importantes quanto essa, como resolução de problemas, pensamento lógico e criatividade. Essas habilidades são valiosas em uma variedade de campos, incluindo matemática, ciência da computação, engenharia e muitas outras. Segundo Scaico *et al* (2013, p.93):

> Em primeiro lugar, este tipo de educação permite o desenvolvimento de diversas capacidades que contribuem para melhorar o raciocínio lógico dos estudantes. Programar envolve a habilidade de desenvolver uma solução para um problema, que se for grande requererá o exercício de outras habilidades (como dividir o problema em subproblemas e criar uma solução central).

A tecnologia, a programação e o ensino de matemática estão cada vez mais interligados no mundo atual. O ensino de matemática por meio da programação pode ajudar os alunos a compreender melhor os conceitos matemáticos, desenvolver habilidades importantes como relata Bobsin (2020, p.93):

> [...] a fusão entre o Pensamento Computacional e a Matemática não se trata de qualquer atividade de Matemática, mas de atividades do tipo problemas investigativos, que podem contemplar (mas não apenas): lista de problemas, projetos interdisciplinares, projetos de pesquisa, demonstrações, aplicações em outras áreas do conhecimento, como jogos ou atividades onde segue-se um padrão, onde o estudante possa se identificar, explorar e propor uma solução por ele testada e válida.

O ensino da matemática na Educação Básica tem sido um desafio constante para educadores e alunos, Morais *et al* (2021, p.15) manifesta que:

> Para algumas pessoas, estes conhecimentos são mais difíceis do que para os demais, isto acaba acarretando o desestímulo destas crianças, assim os educadores precisam usar outras metodologias, visando a melhor compreensão do conteúdo.

Visto que, aprendizado de conceitos matemáticos abstratos e a resolução de problemas matemáticos podem ser difíceis para muitos alunos. A programação em Python pode ser uma ferramenta útil para o ensino da matemática, pois permite que os alunos apliquem conceitos matemáticos em projetos concretos e desenvolvam habilidades de resolução de problemas, pois proporciona aos alunos uma preparação para o futuro, uma vez que saber programar é uma habilidade fundamental no mundo moderno. À medida que a tecnologia continua a avançar rapidamente, a programação se torna uma competência essencial em diversas áreas profissionais, como ciência da computação, engenharias, análise de dados e muitas outras

A obra "O Ensino de Matemática por meio da Linguagem de Programação Python" de Pesente, Mator e Avelino (2023) apresenta uma proposta de ensino de matemática utilizando a linguagem de programação Python como ferramenta. Os autores defendem que o uso da programação pode auxiliar no ensino da matemática, tornando o aprendizado mais dinâmico, criativo e interativo.

Souza (2020) enfatiza que a linguagem de programação Python é cada vez mais utilizada em diversos campos científicos e tecnológicos, devido à sua simplicidade e eficiência. Com uma sintaxe clara e legível, Python permite que mesmo os iniciantes na programação possam criar programas e projetos complexos com facilidade. Ao aprender matemática com Python, os alunos estão se preparando para enfrentar os desafios de um mundo cada vez mais orientado pela tecnologia. Eles estão adquirindo habilidades que são altamente procuradas no mercado de trabalho e que os posicionam em uma vantagem competitiva para carreiras futuras. O domínio da programação em Python abre portas para uma ampla gama de oportunidades profissionais, permitindo que os alunos sigam caminhos em constante evolução, se adaptem às demandas do mundo moderno e se tornem parte ativa da revolução digital que está moldando nosso futuro.

Além disso, a vasta biblioteca de módulos e *frameworks* disponíveis para Python tornam-na uma ferramenta poderosa para análise de dados, inteligência artificial, aprendizado de máquina, entre outras aplicações. Roque (2019, p.22) ressalta que:

> Python ser do tipo scripting, o que a torna mais simples em relação a linguagens copiladas como a C++ ou Java. Outras linguagens do tipo scripting foram vistas como competidoras para a Python, por exemplo, JavaScript e Pearl. No entanto JavaScript tem suas aplicações voltadas quase que totalmente para a área de web-sites dinâmicos enquanto que Pearl, segundo o próprio criador, foi criada para agradar hackers[..]

No entanto, muitos estudantes podem encontrar dificuldades de compreender os conceitos matemáticos subjacentes para utilizar efetivamente a programação em Python, devido o déficit em matemática básica. Uma das principais razões para isso é a falta de metodologias ativas, novas ferramentas de ensino como a tecnologia que podem estimular a criatividade e o raciocínio logico. Souza (2020, p.10) afirma que

> [...]faz-se urgente a necessidade do professor de matemática utilizar linguagens de programação para mostrar aos seus alunos como ela é uma ferramenta extraordinária e como pode ser utilizada para a solução de inúmeros problemas envolvendo os conteúdos ensinados na sala de aula.

Costa *et al* (2017) aborda a implementação do ensino de programação utilizando a linguagem Python em escolas públicas de Ensino Médio no Rio de Janeiro. A autora apresenta um relato detalhado de sua experiência ao introduzir a programação em suas aulas, além de discutir os benefícios e desafios dessa abordagem.

Ao longo do texto, a autora aborda a importância do ensino de programação nas escolas, a escolha da linguagem Python, os recursos necessários para implementar o ensino de programação, bem como a metodologia e o planejamento das aulas.

Além disso, compartilha sua experiência ao lidar com a resistência de alguns alunos e colegas de trabalho em relação ao ensino de programação, bem como sua busca por recursos e parcerias para viabilizar a implementação do projeto.

A programação em Python oferece aos alunos a chance de visualizar e manipular dados, gráficos e modelos matemáticos. Eles podem criar programas e algoritmos que executam cálculos complexos, resolvem equações e analisam dados de forma eficiente. Essa abordagem prática torna a matemática mais tangível e acessível, despertando o interesse e a curiosidade dos alunos. Segundo Souza (2023, p.33).

> Considera-se a Linguagem de Programação Python vantajosa, pois é fácil, intuitiva, organizada e multiplataforma (funciona em vários sistemas operacionais). Além disso, é uma linguagem de propósito geral, ou seja, pode ser utilizada com vários objetivos e finalidades.

A programação em Python permite que os alunos experimentem e testem diferentes abordagens para a resolução de problemas matemáticos. Eles podem criar projetos e jogos que envolvam conceitos matemáticos, o que torna o aprendizado mais divertido e desafiador. A interatividade da programação incentiva os alunos a explorar e investigar, estimulando seu engajamento e criatividade. O ensino de matemática com Python não apenas torna a disciplina mais interessante e envolvente para os alunos, mas também os prepara para o futuro, capacitando-os com habilidades relevantes para o mundo profissional atual e emergente. Ao combinar a matemática com a programação, os alunos têm a oportunidade de explorar conceitos de forma prática, aplicada e criativa, desenvolvendo um pensamento lógico e analítico que lhes será útil em diversas áreas de estudo e carreira.

É importante destacar que os laboratórios de programação no Ensino Médio também ajudam os alunos a entender como a tecnologia funciona e a aplicar seus conhecimentos em projetos práticos. Eles permitem que os alunos criem seus próprios aplicativos, jogos e sites, o que pode ser uma excelente maneira de motivá-los e engajá-los no aprendizado. Melo (2014, p.32) relata:

> Laboratórios de informática, em escolas e universidades, além de proverem recursos de Tecnologia Assistiva, devem ser organizados com o Desenho Universal em mente. Essa organização envolve o acesso ao laboratório, a adequação de seu espaço físico, a organização do mobiliário, a disponibilização de recursos de informática em hardware e software que colaborem à autonomia de pessoas com deficiência, entre outros.

De acordo com Miranda e Meggiolaro (2022), uma das principais vantagens de usar a programação em Python na matemática básica é a capacidade de visualizar conceitos matemáticos complexos através de gráficos e modelos. Com isso criando modelos matemáticos que simulam situações do mundo real. Esses modelos podem ser usados para prever resultados ou testar hipóteses, ajudando a tomar decisões controladas em diversas áreas, desde finanças até ciência e engenharia. O autor ainda afirma que a biblioteca *Matplotlib*, por exemplo, permite criar gráficos em 2D e 3D para visualizar funções e dados matemáticos, enquanto a biblioteca *NumPy* pode ser usada para criar matrizes e vetores, facilitando a realização de cálculos matemáticos complexos.

Daniel e Varricchio (2020) profere que a programação em Python pode ser usada para automatizar cálculos matemáticos repetitivos ou complexos, economizando tempo e

esforço. Por exemplo, o pacote *SymPy* é uma biblioteca para a matemática simbólica que permite simplificar e resolver problemas matemáticos, enquanto o pacote *SciPy* pode ser usado para realizar cálculos numéricos e análises estatísticas.

O ensino da Estatística é fundamental no Ensino Médio, pois ela está presente em nosso cotidiano de diversas formas, sendo uma ferramenta importante para a tomada de decisões em diversos contextos sociais, científicos. Segundo a BNCC (Brasil, 2019) reconhece a Estatística como uma disciplina essencial, pois a Estatística está presente no cotidiano das pessoas de diversas maneiras Como Ribeiro (2022, p. 47) enuncia:

> Saber a origem de qualquer segmento do conhecimento, não é uma tarefa tão simples, e com a Estatística não é diferente, pois tudo dependerá da compreensão que fizermos sobre esse saber e isso muda com o passar do tempo. Ainda hoje, no conceito da população em geral, a palavra Estatística traz à memória dados numéricos e gráficos, relacionados a fatos demográficos e econômicos publicados pelo governo.

Ainda em concordância com o autor a BNCC busca capacitar os estudantes a interpretar pesquisas de opinião, análises de mercado, indicadores sociais e outras informações estatísticas presentes na mídia. Essa compreensão crítica possibilita que os alunos se tornem cidadãos protegidos, capazes de tomar decisões embasadas e participar ativamente da sociedade, uma das principais razões para a importância do ensino da Estatística no Ensino Médio é que ela permite que os alunos compreendam melhor e saibam lidar com as informações estatísticas presentes na mídia, como pesquisas de opinião, análises de mercado, indicadores sociais, entre outros. Compreender essas informações e como elas são construídas é essencial para formar uma opinião crítica e consciente sobre a realidade à nossa volta.

Ao resolver problemas de matemática e estatística usando programação em Python, os alunos são incentivados a pensar de forma crítica sobre como os dados são coletados e analisados encontrando soluções eficazes para problemas complexos. Morais *et al* (2021, p.20) afirma que:

> O desenvolvimento de simulações que complementem a análise experimental e estejam diretamente interligadas ao conteúdo teórico, leva o aluno a assumir um papel mais ativo, pois, além da realização do experimento, ele é induzido a buscar o embasamento teórico para a compreensão dos resultados da simulação e correta comparação deste com os dados experimentais, tornando a aprendizagem mais dinâmica.

A programação em Python também pode ser usada para calcular o desvio padrão e a variação. Essas medidas são importantes na estatística para entender como os dados estão distribuídos e quão representativos são de uma população maior. A biblioteca *Statistics* do Python oferece uma ampla gama de funções para calcular a variação e o desvio padrão, bem como outros valores estatísticos importantes.

Os gráficos são uma ferramenta importante para representar visualmente os dados. Em concordância com Lopes (2019, p.10):

> A apresentação dos dados estatísticos através de tabelas ou medidas de centralidade e variabilidade nem sempre proporciona um entendimento adequado dos dados. Assim, com a finalidade de melhorar esse processo, muitos recorrem ao uso dos gráficos. Para isso, é necessário saber o que se pretende mostrar, como elaborar o gráfico e qual o tipo de gráfico mais apropriado para cada tema abordado.

Rocha (2022, p. 42) ressalta que "é possível utilizar a biblioteca *Matplotlib* para gerar gráficos e tabelas básicas.", logo essa biblioteca permite aos alunos criar gráficos facilmente para representar os dados de uma forma clara e compreensível. Isso pode ajuda os alunos a visualizar padrões e tendências em dados numéricos e comunicar seus resultados de forma eficaz.

Nesse contexto, seguem os objetivos gerais e especifico deste trabalho.

- Objetivo geral:
  - Mostrar a importância do ensino da programação em Python para os alunos do Ensino Médio e sendo aplicado no ensino da estatística descritiva evidenciando como essa disciplina pode ser aplicada na prática.

- Objetivos específicos:
  - Promover o pensamento crítico e o raciocínio lógico dos alunos por meio da programação em Python
  - Estudar os conceitos básicos da linguagem de programação Python, como variáveis, funções e estruturas de dados, permitindo a análise e visualização de dados estatísticos.
  - Estudar sobre estatística descritiva como medidas de tendência central (média aritmética, moda e mediana) e medidas de dispersão (variância e desvio padrão).

- Capacitar os alunos a desenvolver habilidades práticas e teóricas em estatística e programação
- Desenvolver e aperfeiçoar as habilidades interpessoais como trabalho em equipe e comunicação

A segunda parte do trabalho discute sobre aplicação de uma metodologia baseada no uso de problemas contextualizados, enfatizando a aprendizagem ativa e o desenvolvimento de habilidades práticas, o trabalho em equipe e a aplicação de soluções criativas para problemas que são relevantes para o mundo real.

Na terceira parte são apresentados os fundamentos teóricos que embasam a proposta de ensino por meio da programação e irá revisar a literatura já existente sobre o uso da programação em Python no ensino da matemática e estatística destacando a importância da programação para ajudar os alunos a desenvolver habilidades essenciais, como resolução de problemas, raciocínio lógico e criatividade. A fusão do pensamento computacional e da matemática pode ajudar os alunos a entender melhor os conceitos matemáticos, tornando o aprendizado mais interativo e dinâmico.

A quarta parte do trabalho será fundamentado em conceitos básicos de programação em Python, como códigos, algoritmos, estrutura de dados, estruturas de controle, como elas são usadas para controlar o fluxo de execução do programa e explicitar a representação de dados de forma mais visual, clara e intuitiva, podendo ajudar os alunos a compreender conceitos estatísticos, média, mediana, moda, desvio padrão e variação, de forma mais fácil permitindo que os discentes interajam com os dados, criando suas próprias visualizações e explorando diferentes cenários.

A quinta parte consiste em mostrar o uso da programação em Python na sala de aula e como pode ajudar os alunos a desenvolver habilidades práticas e teóricas, promovendo pensamento crítico e a resolução de problemas. Os exemplos de como a calcular média, mediana e moda, podendo ser usado para entender a aplicação da estatística descritiva mediante de situações reais e mostrando uma forma metodológica de trabalhar em grupos, podendo ajudar os alunos a desenvolver habilidades interpessoais, como colaboração, comunicação e resolução de conflitos.

Nas considerações finais serão destacados a importância da contribuição da linguagem de programação em Python para a área educacional, podendo ser utilizada como uma ferramenta complementar para o ensino de estatística descritiva, tornando o aprendizado mais interativo, dinâmico e prático, apresenta uma nova perspectiva para o ensino da

matemática e estatística, além de fornecer privilégios para pesquisas futuras sobre o uso da programação em Python no ensino dessas áreas.

2 METODOLOGIA

A estatística descritiva é uma área importante da estatística que consiste em resumir e descrever dados por meio de medidas numéricas. As medidas de tendência central e desvio padrão são algumas das medidas mais comuns utilizadas nessa área. A programação em Python é uma ferramenta poderosa que pode ser utilizada para realizar cálculos e análises estatísticas de forma rápida e precisa.

Para promover o pensamento crítico, é importante ensinar aos alunos como identificar, analisar e resolver problemas do mundo real. A programação em Python pode ser usada como uma ferramenta para ajudar os alunos a desenvolver essas habilidades. Por exemplo, um problema do mundo real pode ser a previsão do tempo para o dia seguinte com base em dados meteorológicos históricos. Os alunos podem usar a linguagem Python para escrever um programa que processa esses dados e faz uma previsão precisa do tempo.

Precisa ser apresentado aos alunos os conceitos básicos de programação em Python, como variáveis, estruturas de controle de fluxo, funções e bibliotecas, que são relevantes para a análise de dados. Ao mesmo tempo, é importante que os discentes compreendam a teoria estatística por trás desses conceitos básicos e saibam como aplicá-la corretamente.

A metodologia de ensino da estatística descritiva com programação, proposta neste estudo, consiste em uma abordagem embasada uso de problemas contextualizados que visa enfatizar a aprendizagem ativa e o desenvolvimento de habilidades práticas relevantes para o mundo real.

Essa metodologia inclui aulas expositivas para a apresentação dos conceitos fundamentais, seguidas por exercícios contextualizando problemas da realidade dos alunos para consolidar o aprendizado. Além disso, as atividades em grupo são desenvolvidas para estimular o trabalho em equipe e o desenvolvimento de habilidades interpessoais.

A metodologia também prevê a elaboração de um projeto final, que permitirá a aplicação dos conceitos aprendidos em um problema real. Por fim, é realizada uma avaliação contínua do desempenho dos alunos ao longo do curso e disponibilizado material extra para aqueles que desejam aprofundar seus conhecimentos.

Para as aulas expositivas, serão utilizados recursos como apresentações em slides, vídeos explicativos e exemplos práticos de aplicação da estatística descritiva em situações reais.

Para os exercícios, os alunos utilizarão o Google *Colaboratory* também conhecido como "*Colab*", é um produto desenvolvido pela área de pesquisas científicas do Google,

denominado Google *Research* que, segundo Silva (2020, p.3), "O Google *Colab* ou "Colaboratório" é um ambiente digital disponível na nuvem, gratuito e hospedado pelo Google. O objetivo desta ferramenta é prover serviços de desenvolvimento em Python". Dessa forma, o *Colab* permiti que indivíduos possam escrever e executar códigos Python diretamente pelo navegador, o que o torna uma ferramenta especialmente indicada para a realização de atividades relacionadas à aprendizagem de máquina, análise de dados e educação. O serviço é oferecido gratuitamente pelo Google, e não requer instalação de *software* no computador do usuário, pois todo o ambiente de programação é disponibilizado na nuvem.

As atividades em grupo serão realizadas, com o objetivo de promover a colaboração e trabalho em equipe, e consistirão em projetos que envolvam análise de dados utilizando a Estatística Descritiva em Python.

O projeto final é uma atividade que visa aplicar os conhecimentos adquiridos em um problema real. Nessa fase, os alunos têm a oportunidade de escolher um problema que seja de seu interesse ou receber um desafio proposto pelo professor. Essa etapa exige que os alunos apliquem as habilidades de coleta e análise de dados adquiridos durante o curso, utilizando ferramentas de programação em Python para lidar com o problema padrão. Esse processo requer o desenvolvimento de soluções criativas, o trabalho em equipe e a aplicação de técnicas de comunicação para apresentar os resultados de forma clara e objetiva.

Na avaliação contínua do desempenho dos alunos, serão utilizados diversos recursos, como exercícios de fixação, trabalhos em grupo e projetos individuais.

3 O QUE É PROGRAMAÇÃO?

A programação é uma atividade fundamental no mundo moderno, que permite a criação de softwares e sistemas de computação que realizam tarefas específicas.

Matos, Filho e Kiouranis (2019) disserta sobre o surgimento da ideia de programação como uma necessidade para a automatização de tarefas repetitivas e complexas que eram executadas manualmente. Logo no início do século XIX, matemático britânico Charles Babbage (1791-1871) criou o conceito de "máquina analítica". A máquina analítica de Babbage seria capaz de executar qualquer cálculo matemático, mas precisaria de instruções precisas para fazer isso. Ada Lovelace, matemática britânica, trabalhou com Babbage para desenvolver algoritmos para a máquina analítica.

A programação moderna, no entanto, começou a ganhar forma na década de 1950, com o desenvolvimento dos primeiros computadores eletrônicos. Felisbino (2013, p.34) diz que:

> A partir da década de 1950 o desenvolvimento dos computadores tornou-se cada vez mais rápido, seu desempenho foi aumentando, seu tamanho diminuindo e o acesso do público a essa tecnologia continua crescente até os dias atuais. A evolução do acesso aos sistemas computacionais está relacionada com surgimento das linguagens de programação de mais alto nível que facilitaram a utilização dos computadores pelo público em geral.

Rocha (2018) Relata que programadores como Grace Hopper e John Backus desenvolveram as primeiras linguagens de programação, como o Fortran e o COBOL. De acordo com Sebesta (2018) essas linguagens alcançaram a programação mais fácil e acessível a um número maior de pessoas, no entanto a programação começou a evoluir rapidamente, com a criação de novas linguagens de programação e aprimoramento de tecnologias de computação.

Hoje, a programação é uma habilidade essencial para profissionais em diversas áreas, desde a ciência da computação até a medicina, finanças e engenharia. Gotardo (2015, p.17) adverte que:

> Uma linguagem de programação é um método padronizado que usamos para expressar as instruções de um programa a um computador programável. Ela segue um conjunto

> de regras sintáticas e semânticas para definir um programa de computador. Regras sintáticas dizem respeito à forma de escrita e regras semânticas ao conteúdo.

Pode-se dizer que é uma atividade que envolve a criação de software, que consiste em instruções ou algoritmos para um computador executar. Essas instruções podem ser escritas em diferentes linguagens de programação e podem ser usadas para realizar uma variedade de tarefas, desde a criação de um simples programa de calculadora até o desenvolvimento de um sistema operacional complexo.

Os programas de computador são escritos em linguagem de programação, uma linguagem especializada, usada para escrever instruções que podem ser executadas por um computador. Existem muitas linguagens de programação diferentes, cada uma com suas próprias características e propósitos, como por exemplo: Java, Python, C#, C++, PHP, *JavaScript*, entre outras.

É perceptível na sociedade atual o avanço das tecnologias digitais nas últimas décadas, com isso se faz necessário abordar a relação da programação computacional nas escolas de Ensino Médio, pois além de ser uma ferramenta estendida da escrita digital onde podemos abrir espaços para diversos campos do conhecimento, pode ter uma interdisciplinaridade com matemática. Segundo Batista (2019, p.30):

> Acredita-se que quanto antes o conhecimento em informática for apresentado a uma pessoa, maior será a facilidade em aprender. Por isso, a importância de inserção do ensino de lógica de Programação nas escolas contemplando crianças e adolescentes que estão na fase de criação de suas ideias.

Os recursos tecnológicos estão cada vez mais presentes na vida das pessoas, O artigo 26 da Lei 9.394/1996, conhecida como Lei de Diretrizes e Bases da Educação Nacional (LDB) (Brasil, 2017, art.26), diz que:

> Os currículos da educação infantil, do Ensino Fundamental e do Ensino Médio devem ter base nacional comum, a ser complementada, em cada sistema de ensino e em cada estabelecimento escolar, por uma parte diversificada, exigida pelas características regionais e locais da sociedade, da cultura, da economia e dos educandos.

> O artigo 26 estabelece que os currículos do Ensino Fundamental e Médio devem ter uma base nacional comum, complementada por uma parte diversificada considerando as características regionais e locais da sociedade, da cultura e da economia, dos educandos.

A Base Nacional Comum Curricular - BNCC (BRASIL, 2018) fomenta que a programação pode ser vista como uma competência na aprendizagem dos alunos, pois permite que os alunos desenvolvam habilidades relacionadas à resolução de problemas, pensamento crítico, colaboração e criatividade. Além disso, pode ser vista como uma forma de alfabetização digital, pois ajuda os alunos a entenderem como a tecnologia funciona e como podem utilizá-la de maneira eficaz.

> A BNCC reconhece a importância da programação e estabelece alguns objetivos de aprendizagem relacionados a ela. Por exemplo, no Ensino Médio, designando que os alunos devem ser capazes de utilizar ferramentas digitais de comunicação, colaboração, pesquisa e produção de conteúdo, compreender e criar algoritmos para solucionar problemas.

Existem diversas iniciativas que buscam incluir a programação nas escolas e universidades, desde cursos extracurriculares até disciplinas obrigatórias. A programação também pode ser utilizada em projetos interdisciplinares, que envolvem outras áreas do conhecimento, como matemática, física, biologia e artes.

No entanto, ainda existem desafios a serem enfrentados para que a programação seja amplamente criada na educação. Um deles é a falta de professores capacitados para ensinar tal habilidade. Além disso, é preciso garantir que a programação seja acessível a todos, independentemente de seu nível socioeconômico ou localização geográfica.

Apesar dos desafios, essa ferramenta de ensino pode trazer benefícios tanto para os alunos quanto para a sociedade como um todo. Portanto, é importante que as instituições de ensino invistam na inclusão da programação em seus currículos, a fim de preparar seus alunos para um futuro cada vez mais digital e tecnológico.

> Morais, Basso e Fagundes (2017), coloca em ênfase que a programação é uma habilidade importante que deve ser ensinada na escola, pois ajuda os alunos a desenvolverem habilidades de resolução de problemas de forma lógica e analítica, ampliando o raciocínio lógico, que são úteis em muitas outras áreas da vida, como o mercado de trabalho. A maioria das profissões atualmente exige algum conhecimento de programação, habilidades valorizadas pelos empregadores, podendo abrir muitas portas para as carreiras dos alunos, como desenvolvimento de software, engenharia de software, design de jogos e muito mais.

A programação pode ajudar a tornar o ensino da matemática mais personalizado, pois os professores podem monitorar o progresso de cada aluno e adaptar o ensino de acordo

com suas necessidades individuais. Por exemplo com a linguagem de programação em bloco, é possível criar exercícios adaptativos que se ajustam às habilidades e níveis de aprendizado dos alunos. Moram (2017, p.76) evidência que:

> Um dos programas mais utilizados para aprender por meio de programação lúdica é o Scratch. Foi desenvolvido por Michel Resnick no MIT, com o objetivo de incentivar a aprendizagem da programação de forma intuitiva, por meio da montagem de blocos de comandos.

De acordo com a BNCC (2018) o uso da linguagem de programação no ensino da matemática pode ajudar os alunos a desenvolver habilidades de pensamento crítico, resolução de problemas, comunicação, colaboração e aprendizagem autônoma. Os educadores devem considerar incorporar uma programação em suas aulas de matemática para ajudar a preparar os alunos para um mundo cada vez mais digital e tecnológico.

Losekan (2022) ressalta que a programação pode ajudar os alunos a desenvolverem habilidades importantes para o mercado de trabalho. Os alunos que aprendem alguma linguagem de programação principalmente as orientadas a objetos como Python ou C# podem se beneficiar dessa demanda e terem melhores oportunidades de emprego.

Os professores podem monitorar o progresso de cada aluno e adaptar o ensino de acordo com suas necessidades individuais. A programação pode ser usada para ensinar conceitos matemáticos básicos, como adição, subtração, multiplicação e divisão. Os discentes podem escrever códigos simples para executar operações matemáticas básicas facilitando no desenvolvimento da aprendizagem da matemática em sala de aula. Um exemplo Código 1 que pede ao usuário que insira dois números e, em seguida, execute a operação de adição pode ajudar os alunos a entender como a adição funciona na prática. Ao escrever códigos, os alunos também podem ver como a matemática é usada no cotidiano e entender como é importante ter habilidades matemáticas.

```
# Solicitar ao usuário que insira dois números
numero1 = float(input("Digite o primeiro número: "))
numero2 = float(input("Digite o segundo número: "))

# Realizar a operação de adição
soma = numero1 + numero2

# Exibir o resultado da adição
print("A soma dos números é:", soma)
```

Código 1: Exemplo de uma operção de adição

Na primeira linha, começou solicitando ao usuário que insira o primeiro número usando a função input(). A função float()é usada para converter a entrada do usuário em um número de ponto flutuante. Em seguida, solicitamos ao usuário que insira o segundo número usando a mesma abordagem. Na próxima linha, realizamos a operação de adição dos dois números, mantendo o resultado na variável soma. Por fim, exibimos o resultado da adição usando a função print(). O texto "A soma dos números é:" é seguido pelo valor da variável soma.

A dissertação "Estudo de algoritmos e programação de computadores para resolver problemas de matemática no Ensino Médio" de Juliana Perpétua Elias, apresentada à Universidade de São Paulo em 2019, tem como objetivo investigar como o estudo de algoritmos e programação de computadores pode contribuir para a resolução de problemas de matemática no Ensino Médio, apresentando uma revisão bibliográfica sobre a importância do ensino de programação de computadores na educação, destacando a sua relevância para o desenvolvimento do pensamento lógico e crítico, além de sua aplicação em diversas áreas do conhecimento, incluindo a matemática.

Em seguida, mostra um estudo de caso realizado em uma escola de Ensino Médio, no qual foi aplicado um projeto pedagógico envolvendo o ensino de programação de computadores para a resolução de problemas de matemática. O projeto foi implementado em turmas de Ensino Médio e os resultados foram avaliados através de questionários aplicados aos alunos e professores.

Os resultados do estudo mostraram que o ensino de programação de computadores pode contribuir significativamente para a resolução de problemas de matemática no Ensino Médio, desenvolvendo habilidades importantes para a formação dos alunos, como o pensamento crítico, criatividade e colaboração. Além disso, a obra apresenta recomendações para a inclusão do ensino de programação de computadores no currículo escolar e para a formação de professores para a sua aplicação em sala de aula.

3.1 PROGRAMAÇÃO EM PYTHON NO ENSINO DA MATEMÁTICA

A aprendizagem matemática é uma parte essencial do currículo escolar e desempenha um papel importante no desenvolvimento cognitivo dos discentes. A matemática é uma disciplina complexa que requer habilidades de raciocínio, pensamento lógico, resolução de problemas e tomada de decisões. A capacidade de realizar essas habilidades é fundamental para o sucesso acadêmico e profissional, entretanto o pensamento matemático acontece desde as primeiras civilizações. Pesente (2019, p.33) diz que:

> A matemática se faz presente no desenvolvimento da humanidade desde os primórdios dos tempos, em esculturas, desenhos rupestres, papiros, entre outros. Basicamente tudo o que de forma empírica for analisado possui conceitos matemáticos, como exemplo as pirâmides do Egito, onde os egípcios, um povo avançado em sua época, definiu formas e cálculos matemáticos para a construção de suas pirâmides.

A matemática é uma das áreas mais importantes do conhecimento humano e está presente em praticamente todos os aspectos da vida moderna, desde a tecnologia até as finanças pessoais.

Desse modo, como foi visto no tópico anterior, com o advento da tecnologia, novas ferramentas foram aprimoradas para auxiliar no ensino e aprendizagem da matemática, tornando o processo mais eficiente e acessível para um público mais amplo.

Perosa (2021) destaca que, o ensino da programação também desempenha um papel importante podendo permitir que os alunos apliquem os conceitos matemáticos de uma maneira mais prática e tangível, o que pode ajudá-los a entender melhor como os conceitos matemáticos se aplicam na rotina em sociedade. Aprender a programar pode ajudar os alunos a desenvolver habilidades como pensamento crítico e lógico, resolução de problemas e estimular a criatividade. A programação pode ajudar a preparar os alunos para o futuro, em um mercado de trabalho cada vez mais voltado para a tecnologia. Como diz Carniello e Zanotello (2020, p.194):

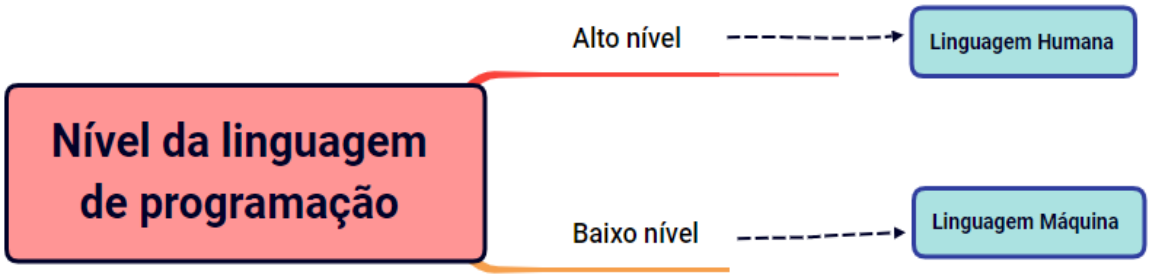
> O pensamento computacional, o qual contribui no desenvolvimento de habilidades como raciocínio lógico, autonomia, pensamento crítico, colaboração, trabalho em equipe, empatia e capacidade de resolver problemas complexos, pode se tornar uma alternativa a esse desafio que se impõe à Educação contemporânea.

A utilização da linguagem Python tem ganhado cada vez mais espaço no ensino de Matemática, tanto no Ensino Médio quanto no Ensino Superior. Rolim (2021, p.23) norteia que:

> Nesse cenário, a linguagem de programação via Python possibilitará que os estudantes desenvolvam estratégias para a resolução de problemas em diferentes perspectivas, contribuindo com o aprimoramento do raciocínio lógico. Permitindo, assim, a visualização, implementação, testagem e depuração de diversas possibilidades para a resolução de problemas.

O motivo para isso é o fato de Python ser uma linguagem de programação de alto nível, assim como é apresentado na Figura 1 as linguagens de alto nível são aquelas mais próximas da linguagem humana, com sintaxe simples e fácil de aprender.

Figura 1 - Diagrama dos níveis de linguagem de programação



Fonte: Compilação da própria autora (2023).

Manzano e Oliveira (2019 p.25) explica que:

> As linguagens de alto nível possibilitam maior facilidade de comunicação com um computador, pelo fato de serem próximas à comunicação humana, pois se baseiam em palavras do idioma inglês. Destacam-se, nessa categoria, linguagens de programação como FORTRAN, COBOL, BASIC, PASCAL, C, JAVA, Lua, C++, entre outras. Esse tipo de linguagem é mais facilmente assimilado por seres humanos. Assim sendo, o número de pessoas que as conhece é bastante grande.

Python é uma linguagem de programação popular e versátil que tem sido cada vez mais utilizada como ferramenta educacional no ensino de matemática além de possuir inúmeras

bibliotecas específicas para a Matemática, como *numpy*, *scipy* e *matplotlib*, que permitem a manipulação e visualização de dados de forma eficiente.

Uma das principais razões pelas quais a programação em Python é gratificante para o ensino da matemática é a capacidade de realizar exercícios complexos rapidamente, como de problemas de geometria, trigonometria e outras áreas da matemática.

Além disso, Carneiro (2022) salienta que a programação em Python pode ser usada para ensinar habilidades matemáticas fundamentais, como resolução de problemas, lógica e pensamento crítico. Os alunos podem aprender a decompor um problema matemático complexo em partes menores, aplicar as técnicas matemáticas apropriadas para resolver cada parte e, em seguida, combinar as soluções para resolver o problema completo.

A programação em Python pode ser uma ferramenta valiosa para o ensino da matemática. Ela oferece aos alunos uma maneira prática de aplicar os conceitos matemáticos e ajuda a tornar a aprendizagem mais interativa e envolvente.

A utilização da programação em Python no ensino de matemática traz diversos benefícios para os alunos, como:

- Estimulação da criatividade: permite que os alunos desenvolvam soluções criativas para problemas matemáticos, o que pode ajudá-los a pensar com mais criatividade e encontrar novas abordagens para solucionar desafios.
- Desenvolvimento de habilidades em programação: ao aprender a programar em Python, os alunos desenvolvem habilidades em lógica de programação, pensamento computacional e resolução de problemas, que são valiosas não apenas para a matemática, mas para diversas áreas.
- Aprendizado mais significativo: pode ajudar os alunos a compreender conceitos matemáticos de forma mais clara e intuitiva, além de facilitar a análise e interpretação de dados.
- Melhoria da motivação e engajamento: pode tornar o aprendizado mais divertido e interessante para os alunos, o que pode aumentar a motivação e engajamento com a matéria.
- Preparação para o mercado de trabalho: é uma habilidade muito valorizada no mercado de trabalho atualmente, o que pode ser benéfico para os alunos que desejam seguir carreiras na área de tecnologia.

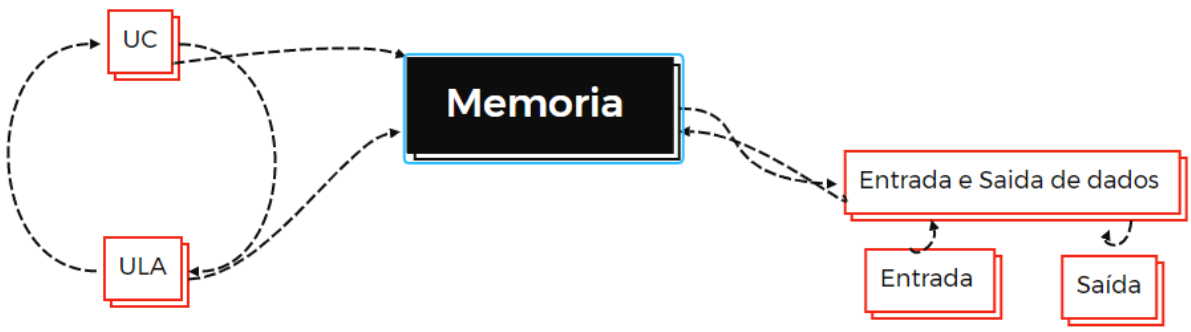
A programação em Python é uma ferramenta poderosa para o ensino de matemática para a educação básica, pois permite que os alunos tenham contato com conceitos matemáticos de forma lúdica e interativa, desenvolvendo habilidades em programação. Podendo trazer diversos benefícios para os alunos, como a estimulação da criatividade, desenvolvimento de habilidades em programação, aprendizado mais significativo, melhoria da motivação, engajamento e preparação para o mercado de trabalho.

3.2 ALGORITMOS DE PROGRAMAÇÃO

Algoritmos são sequências de instruções bem definidas que permitem a resolução de problemas computacionais. Eles são usados para automatizar tarefas, realizar cálculos e executar processos complexos em máquinas. Os algoritmos podem ser sentidos de diferentes maneiras, como em pseudocódigo, fluxogramas ou linguagens de programação. Logo, segundo Bannin (2018) para que o algoritmo se torne um programa computacional precisa ser aplicado em alguma linguagem de programação.

Gotardo (2015) coloca em ênfase o trabalho de John Von Neumann, matemático da Hungria e um estudioso importante na área da computação, que montou um modelo de pensamento computacional e o processo que acontece quando um programa é armazenado na memória do computador.

Figura 2 - Modelo de von Neumann de armazenamento de um programa



Fonte: Compilação da própria autora (2023).

O autor ainda detalha esses componentes mostrados no fluxograma de Neumann na Figura 2 da seguinte maneira (Gotardo, 2015, p.15):

> • Unidade Lógica e Aritmética (ULA): um computador deve ser capaz de realizar operações básicas sobre dados. É nesta unidade que as operações acontecem. • Unidade de Controle (UC): esta unidade é responsável por "orquestrar" as demais unidades do computador. Ela decodifica as instruções de um programa num grupo de comandos básicos, blocos de construção, e gerência como se dá a execução destes comandos com as demais unidades. • Memória: é o local onde os programas e os dados são armazenados durante a execução do programa, ou seja, o processamento dos dados. • Unidade de Entrada e Saída: esta unidade é responsável por receber dados externos ao computador e enviar de volta os resultados obtidos.

Com isso ao se escrever um programa ele forma um conjunto de dados que aos serem executados de forma sequencial com o que foi escrito são armazenados na memória do computador podendo assim fazer a transformação, envio e recebimento desses dados.

Os algoritmos funcionam seguindo uma sequência de passos bem definidos. Eles recebem uma entrada e processam uma saída.

Por exemplo, um algoritmo para calcular a média de três números seriados:

1. Receber três números como entrada;
2. Somar os três números;
3. Dividir a soma por 3;
4. Produzir a média como saída.

Este algoritmo recebe três números como entrada, realiza a soma desses números, divide o resultado por 3 e produz a média como saída.

Batista (2019) expõe que os algoritmos de programação têm um papel importante na educação, nesse caso ajudam os alunos a desenvolver habilidades de resolução de problemas, raciocínio lógico e pensamento crítico. Além disso, eles podem tornar o ensino da matemática mais personalizado e concreto, permitindo que os alunos visualizem conceitos abstratos de uma maneira mais tangível. Com uma demanda crescente por profissionais com habilidades em programação, o ensino de algoritmos de programação pode ajudar os alunos a se prepararem para o mercado de trabalho e terem melhores oportunidades de emprego no futuro.

Pesente (2019) enfatiza que a programação como ferramenta para o ensino da matemática é uma aprendizagem significativa, pois com construção de algoritmos nesse

processo, os discentes adquirem habilidades de expressar algoritmos matemáticos de uma maneira mais acessível e intuitiva. Quando os alunos aprendem a programar, eles aprendem a pensar de forma lógica e desenvolvem habilidades importantes de resolução de problemas, que são fundamentais para o aprendizado da matemática. Galvão (2021, p.8) enuncia que:

> Os algoritmos fazem parte da matemática, muito antes da computação digital. Processos de adições, subtrações, multiplicações e divisões entre números com dois ou mais dígitos são ensinados em forma de algoritmos, mesmo sem a associação com a nomenclatura do termo. Uma infinidade de conceitos matemáticos pode ser traduzidos em algoritmos, tais como: operações aritméticas com frações, cálculo de máximo divisor comum (MDC), cálculo de mínimo múltiplo comum (MMC), fatoração de um número, cálculo de raízes de uma equação do 2º grau, construções geométricas, etc.

Os algoritmos de programação podem ajudar os alunos a visualizar conceitos abstratos de matemática. Por exemplo, um programa que desenha um gráfico de uma função matemática pode ajudar os alunos a entender como a função se comporta e como ela pode ser manipulada. Além disso, usar programas para simular situações matemáticas complexas, ajuda a tornar a matemática mais concreta e tangível.

3.3 LABORATÓRIO DE COMPUTAÇÃO

O laboratório de computação pode ser um recurso essencial para o ensino de programação no Ensino Médio. Onde os alunos têm acesso a computadores e softwares de programação, bem como a orientação de professores e tutores experientes.

Nascimento, Santos e Neto (2018) defendem, a importância de um laboratório de computação para o ensino básico de programação e destacam a importância de experimentar com diferentes linguagens de programação na prática.

O laboratório de computação pode ajudar a tornar o ensino de programação mais acessível para os alunos. “A tecnologia educacional poderá contribuir de forma construtiva para novas práticas pedagógicas, baseando-se em novas concepções dos conhecimentos. A ferramenta apresenta-se também como suporte de auxílio no desenvolvimento pedagógico” (Leal, 2018, p.14).

O acesso a computadores e softwares de programação pode ser um desafio para alguns estudantes fora do ambiente escolar. O laboratório de computação pode fornecer a esses

estudantes um ambiente acessível e equipado com as ferramentas necessárias para aprender a programação.

Em resumo, o laboratório de computação pode ser uma ferramenta valiosa para apoiar o ensino de programação no Ensino Médio. Os estudiosos enfatizam a importância da prática na resolução de problemas e pensamento crítico no aprendizado de programação. Podendo ser usado para ensinar aos estudantes como utilizar a tecnologia de forma responsável e ética, o que é enfatizado pela BNCC, aprender sobre segurança cibernética, privacidade e direitos autorais, enquanto aprendem a programar no laboratório de computação.

No que se refere ao ensino de programação, a BNCC destaca a importância de fornecer aos alunos acesso a tecnologias digitais e oportunidades para experimentar, criar e solucionar problemas com a programação.

3.4 LÓGICA DE PROGRAMAÇÃO

A lógica de programação é uma disciplina fundamental na área de computação, pois é responsável por definir a sequência de instruções que um programa deve seguir para realizar uma determinada tarefa. É a habilidade de criar um fluxo de instruções lógicas que podem ser executadas sequencialmente e que levam a um resultado desejado. De acordo com Gonçalves (2017, p.13):

> A lógica de programação visa utilizar essa máxima da organização do pensamento e aplicá-la na programação de computadores com o intuito de desenvolver raciocínio lógico e artifícios que auxiliem na produção de uma solução logicamente correta e satisfatória que resolva o problema desejado com certa qualidade.

Para criar um *software* eficiente e robusto, é necessário que o desenvolvedor possua uma boa compreensão dos princípios da lógica de programação. Gonçalves (2017) ressalta que a lógica de programação e o pensamento lógico estão intimamente relacionados. A lógica de programação é a aplicação prática do pensamento lógico, pois permite que sejam criados algoritmos e programas de computador que resolvam problemas complexos de forma estruturada e organizada.

Para que um programador possa criar programas eficientes, é necessário que ele possua um bom raciocínio lógico. Isso significa que ele deve ser capaz de analisar informações

de forma lógica e coerente, desenvolver soluções criativas e eficientes para problemas complexos.

A matemática e a lógica de programação estão intimamente relacionadas, pois ambas se baseiam na resolução de problemas lógicos. Ao aprender lógica de programação, os alunos desenvolvem habilidades importantes que podem ser aplicadas em muitas áreas da matemática, como álgebra, geometria e análise. Silva (2019, p.23) diz que:

> A Matemática está ligada desde a parte mais abstrata nos componentes de computadores, até a parte mais concreta dos componentes da máquina. Um exemplo bem simples é que ela é usada na resolução dos problemas e uso da lógica, até porque, não há como fazer operações Matemáticas, sem o uso da Matemática.

A lógica de programação ajuda os alunos a desenvolverem habilidades de resolução de problemas, pois eles precisam criar soluções lógicas para os desafios que confrontam na programação, identificar e definir o problema, entender as condições necessárias para a resolução do problema, reconhecer as etapas necessárias para alcançar a solução, testar e corrigir a solução até que ela funcione corretamente.

Aprender lógica de programação também ajuda os alunos a desenvolverem habilidades de raciocínio lógico, que são fundamentais na matemática, pois é necessário que aprendam a pensar de forma lógica e organizada, identificando padrões e estabelecendo relações entre diferentes elementos. Isso ajuda os alunos a entenderem melhor as propriedades matemáticas e resolverem problemas de forma mais eficiente. Batista (2019, p.31) ressalta que:

> [...]ainda dizem que os programadores compartilham projetos interativos, além de aprenderem matemática e conceitos computacionais, bem como pensar criativamente, raciocinar sistematicamente e trabalhar de forma colaborativa. O principal objetivo não e preparar pessoas para carreiras de programadores, mas sim criar uma nova geração criativa, que use a programação para expressar suas ideias.

Além disso, a lógica de programação ensina os alunos a pensar de forma crítica e testar suas soluções de forma rigorosa. Isso é importante porque na matemática é fundamental

que as soluções sejam precisas e corretas. Aprender a testar e validar soluções é uma habilidade valiosa que os alunos podem aplicar em todas as áreas da matemática.

A BNCC destaca a importância da lógica de programação como uma das habilidades essenciais para o desenvolvimento dos estudantes na era digital em que vivem. Segundo a BNCC, a lógica de programação deve ser ensinada de forma transversal, ou seja, integrada a diferentes áreas do conhecimento, como a matemática e criatividade, colaboração e comunicação. Por exemplo, a criação de algoritmos envolve a identificação de padrões e a aplicação de fórmulas matemáticas, o que ajuda os alunos a entenderem melhor os conceitos matemáticos, além das habilidades matemáticas, isso porque a programação envolve a criação de soluções inovadoras para problemas complexos, o que requer a colaboração e comunicação entre os membros da equipe.

3.5 LÓGICA MATEMÁTICA

A lógica matemática é um ramo da matemática que se dedica ao estudo da validade dos argumentos matemáticos. Ela é baseada em um conjunto de regras formais que permitem a construção de argumentos matemáticos válidos. É uma disciplina fundamental para a compreensão dos fundamentos da matemática, permitindo uma análise rigorosa de teoremas e a construção de novos resultados. Segundo Fajardo (2017, p.3):

> A lógica surgiu basicamente com dois propósitos: o de formalizar as “leis do pensamento” (essa expressão foi utilizada por outro pioneiro da lógica: George Boole), que utilizamos constantemente para argumentar e chegar a conclusões corretas a partir de premissas dadas, e o de estabelecer uma linguagem mais apropriada para a matemática e a filosofia, para evitar as armadilhas dos paradoxos e dos sofismas.

Com isso papel da lógica é ordenar o pensamento e permitir que sejam mantidas relações entre as informações de forma lógica e coerente. Desse modo a lógica matemática é baseada em um conjunto de regras formais que permitem a construção de argumentos matemáticos válidos. Essas regras são inspiradas em símbolos e operadores lógicos, como negação, conjunção e disjunção, e permitem a manipulação formal de expressões matemáticas.

A lógica matemática é essencial para a análise de sistemas formais, como a aritmética e a geometria. Esses sistemas são baseados em um conjunto de axiomas e regras de

inferência, que permitem a dedução de novos resultados a partir dos axiomas. A matemática é utilizada para garantir a consistência desses sistemas e para a construção de novos resultados a partir dos axiomas. Fajardo (2017, p.8) enuncia novamente que:

> Portanto, na matemática moderna, uma demonstração é uma sequência de fórmulas matemáticas, em uma linguagem lógica apropriada, em que cada fórmula ou é um axioma ou é obtida a partir de fórmulas anteriores através de uma regra de inferência. Um teorema é qualquer uma dessas fórmulas que ocorrem em uma demonstração.

Ferreira e Mendonça (2017) ressalta que a lógica matemática também é utilizada em áreas aplicadas da matemática, como a teoria da computação e a inteligência artificial. A teoria da computação é o ramo da matemática que estuda os algoritmos e a computabilidade, enquanto a inteligência artificial se dedica ao estudo de sistemas inteligentes. A matemática é utilizada nesses campos para a construção de sistemas formais que representam a manipulação de informações complexas.

Segundo Silva (2019) a lógica de programação também é útil para a compreensão da matemática lógica, uma vez que ela permite a representação e manipulação de expressões matemáticas em um formato que é mais fácil de ser compreendido pelos seres humanos. Por exemplo, um programa de computador pode ser utilizado para realizar cálculos matemáticos complexos, que podem ser difíceis de serem realizados manualmente. Além disso, a lógica de programação pode ser utilizada para a visualização de conceitos matemáticos, permitindo que os alunos vejam as relações entre diferentes conceitos de forma mais clara.

Um exemplo de como a lógica de programação pode ser utilizado para auxiliar na compreensão da lógica matemática é o uso de linguagens de programação como Python. Os gráficos são uma ferramenta importante na matemática, permitindo que os alunos visualizem as relações entre variáveis. A lógica de programação pode ser utilizada para a construção de gráficos de forma automatizada, permitindo que os alunos visualizem essas relações de forma mais clara.

A lógica de programação, por sua vez, é uma linguagem formal que descreve a sequência de instruções que um programa deve seguir para resolver um problema. Ela é baseada em símbolos e operadores lógicos, como *if*, *else* e *while*, e permite a descrição formal do comportamento de um programa.

Uma das principais relações entre a lógica matemática e a lógica de programação é a aplicação dos conceitos de verdadeiro e falso. Na lógica matemática, as expressões são verdadeiras ou falsas com base na sua estrutura e nas regras de inferência. Na lógica de programação, as instruções são executadas ou não com base nas condições definidas pelas expressões *booleanas*. A capacidade de entender a lógica das expressões *booleanas* é fundamental para a compreensão da estrutura dos programas e para a construção de algoritmos eficientes. Bertolini, Cunha e Fortes (2017, p.12) expõe:

> A lógica matemática é de fundamental importância para as linguagens de programação necessárias para a construção de programas de computador (softwares). É com base na lógica matemática que as linguagens de computador são descritas. Em lógica, uma linguagem de computador é dita como linguagem formal, pois o formalismo é dado pela representação matemática.

Outra relação importante entre a lógica matemática e a lógica de programação é a capacidade de construir argumentos lógicos válidos. Na matemática, um argumento é válido se as suas premissas são verdadeiras e sua conclusão segue logicamente a partir dessas premissas. Na lógica de programação, um programa é correto se ele resolver o problema padrão de acordo com as especificações. A habilidade de construir argumentos lógicos válidos é fundamental tanto para a resolução de problemas matemáticos como para a construção de programas eficientes e corretos.

Na lógica matemática, as estruturas são utilizadas para representar objetos matemáticos, como grupos, anéis e campos. Na lógica de programação, as estruturas são utilizadas para representar dados, como vetores, matrizes e listas. A capacidade de entender a estrutura dos objetos matemáticos é fundamental para a compreensão da matemática avançada, enquanto a capacidade de entender a estrutura dos dados é fundamental para a construção de programas eficientes.

4 LINGUAGEM DE PROGRAMAÇÃO PYTHON

A linguagem de programação Python é uma das linguagens mais populares. Nunes (2022 p.28) acentua que:

> A linguagem de programação Python se tornou uma das mais populares nos últimos anos, ultrapassando as consolidadas linguagens Java e C. É uma linguagem muito poderosa, além de possuir uma sintaxe simples e uma grande quantidade de bibliotecas disponíveis. Python também conta com uma grande comunidade ativa de usuários, a linguagem é de código aberto com desenvolvimento da comunidade.

Segundo Erse (2021) é conhecida por sua simplicidade, legibilidade de código e eficiência e é usada em muitas áreas, incluindo ciência de dados, inteligência artificial, desenvolvimento de jogos e desenvolvimento web. Foi projetada com o objetivo de ser simples, fácil de aprender e utilizar, com uma sintaxe clara e intuitiva. Python é uma linguagem de programação de alto nível, interpretada, orientada a objetos e dinâmica. Na programação orientada a objetos, os objetos representam entidades ou conceitos do mundo real. Cada objeto possui atributos (variáveis) e comportamentos (métodos). Esses objetos interagem entre si por meio de mensagens, trocando informações e executando ações.

Foi criada no final dos anos 1980 por Guido Van Rossum, programador holandês, enquanto trabalhava no Centro de Matemática e Informática (CWI) na Holanda. Os autores Ramos, Nascimento e Bischoff (2022) ressalta que o nome "Python" foi inspirado no programa de televisão britânico "*Monty Python's Flying Circus*", que era popular entre os programadores da época. Desde o seu lançamento inicial, a linguagem Python passou por várias atualizações e evoluções, com novas funcionalidades.

Os objetivos da linguagem Python incluem simplicidade, legibilidade, portabilidade, biblioteca padrão abrangente e suporte à programação orientada a objetos. Entre os benefícios da linguagem Python, destacam-se sua facilidade de aprendizado, legibilidade de código, grande comunidade de usuários, versatilidade e eficiência. Cordeiro (2016, p.14) diz que:

> A linguagem foi projetada com a filosofia de enfatizar a importância do esforço do programador sobre o esforço computacional. Prioriza a legibilidade do código sobre a velocidade ou expressividade. Combina uma sintaxe concisa e clara com os recursos

> poderosos de sua biblioteca padrão e por módulos e frameworks desenvolvidos por terceiros.

Tem uma ampla variedade de estruturas de dados integradas que permitem que os programadores armazenem e manipulem dados de maneira eficiente. É uma linguagem de programação amplamente utilizada em análise de dados e estatística. Lacerda (2021, p.8) destaca que:

> O Python é uma grande ferramenta e ela é comumente usada na comunidade de ciência de dados, por conta de sua facilidade de interpretação e pelo crescente número de bibliotecas de análise de dados.

Com ela, é possível realizar cálculos estatísticos complexos de forma rápida e eficiente, além de possibilitar a criação de gráficos e visualizações para ajudar na interpretação dos resultados.

4.1 CONCEITOS BÁSICOS: VARIÁVEIS, NÚMEROS E OPERADORES

Em Python, os blocos de código são delimitados por indentação, o que significa que o código dentro de um bloco deve ser indentado com um determinado número de espaços ou tabulações. A indentação refere-se à forma como o código é segura visualmente usando espaços ou tabulações no início das linhas de código. É usada para definir blocos de código e delimitar a estrutura de controle do programa, como loops, condicionais, funções e classes. A maioria dos editores de texto modernos têm recursos para ajudar com a indentação, como a identificação automática quando o programador pressiona a tecla "*tab*".

Isso torna o código mais fácil de ler e entender, especialmente para programadores que estão começando a aprender a linguagem. Segundo Jesus (2018, p.55):

> Enquanto que os blocos são delimitados explicitamente em C, Java e PHP por chaves, em Pascal seria Begin e End e Fortran por palavras-chave como then e endif, em Python blocos são delimitados por espaços ou tabulações formando uma indentação visual. Não existem símbolos de “abre” e “fecha.

Para usar o interpretador Python corretamente, é importante saber como acessá-lo e como usá-lo para executar o código. Menezes (2010, p.26) explica que:

> O interpretador Python é uma grande ferramenta para o aprendizado da linguagem. O interpretador é o programa que permite digitar e testar comandos escritos em Python e verificar os resultados instantaneamente.

Independentemente de como o interpretador Python é acessado, é importante usar uma sintaxe correta para garantir que o código seja executado corretamente.

Os comentários são usados para explicar o que o código faz e ajudar outros programadores a entender o código. Os comentários são ignorados pelo interpretador Python e, portanto, não sofreram a execução do código e são precedidos pelo símbolo "#". Jargas (2008, p.36) enfatiza que:

> É para isto que existem os comentários, eles são uma maneira do programa deixar recados no código, explicar o que faz uma linha ou um bloco e até deixar rastreados para ele mesmo ler depois.

A função "*print*" é usada para imprimir ou exibir dados na saída padrão do sistema. É uma ferramenta importante para exibir informações na tela ou em um arquivo. “A palavra *print* é uma função utilizada para enviar dados para a tela do computador. Ao escrevermos *print* ("Olá"), ordenamos ao computador que exiba o texto “Olá!” na tela” (Menezes, 2010, p.26) ela pode ser usada de várias maneiras para ajudar na depuração de código, exibir resultados de cálculos e gerar saída de dados para relatório e outros fins.

```
1 print (“Hello, World!”)
```

Código 2: Função print.

No Código 2 a função "*print*" em Python é usada para exibir uma *string* "*Hello, world!"* na saída padrão do sistema. Podendo imprimir diferentes tipos de dados, como *strings*, números, variáveis, listas, dicionários e objetos. Também é possível especificar um separador entre os argumentos, usando o argumento opcional "*sep*". Por padrão, o separador é um espaço em branco.

Variáveis em Python são espaços na memória do computador que são usados para armazenar valores. Segundo Guedes (2018) esses valores podem ser de vários tipos, como *strings*, números ou *booleanos*. As variáveis permitem que o programador armazene informações e as manipule ao longo do programa.

Para criar uma variável em Python, usamos o sinal de igual (=) para atribuir um valor a um nome de variável. O nome da variável pode ser qualquer combinação de letras, números e sublinhados.

Existem algumas regras para a criação de variáveis, tais como:

- O nome da variável deve começar com uma letra ou com um sublinhado (_);
- O nome da variável pode conter letras, números e sublinhados;
- O nome da variável não pode começar com um número;

É recomendado usar nomes descritivos para que o código seja fácil de entender:

- *Strings*: são sequências de caracteres. Para criar uma variável do tipo *string*, basta escrever o valor entre aspas simples (' ') ou aspas duplas (" ").
- Numéricos: inclui números inteiros, números de ponto flutuante e números complexos. Podem ser usados para armazenar valores numéricos.
- *Booleanas*: podem ser *True* (verdadeiro) ou *False* (falso). Podem ser usados para armazenar valores *booleanos*.

Ao criar uma variável em Python, o interpretador determina o tipo de variável com base no valor que foi atribuído a ela. No Código 3 temos um exemplo onde é possível especificar explicitamente o tipo de variável usando funções como int(), str() ou bool().

```
1 idade = int(30)
2 altura = float(1.75)
3 nome = str("Maria")
4 verdadeiro = bool(True)
```

Código 3: Exemplos de variáveis especificadas (numérica, *string* e boleana).

Existem dois tipos principais de números em Python: inteiros (int) e números de ponto flutuante (*float*). Os números inteiros são números inteiros sem ponto decimal e podem ser positivos ou negativos. Eles são representados pelo tipo de dados "int". Já os números de ponto flutuante são números que possuem uma parte decimal. Eles são representados pelo tipo de dados "*float*". Novamente Guedes (2018, p.26) enfatiza que:

> Uma variável é definida como numérica quando a mesma armazena números inteiros (class “int”) ou de ponto flutuante (class “float”). Números inteiros são aqueles valores negativos e positivos que não possuem parte decimal, como exemplo: 1, 2 ,7, 20, -10 e 100000. Já os números de ponto flutuante ou decimais são os que possuem uma parte decimal separada sintaticamente da parte inteira por um “ponto”, por exemplo: 1.0, 2.5, 20.9, -10.8 e 100000.1.

Baseado em Marinho *et al* (2018) é possível realizar operações aritméticas com números inteiros e de ponto flutuante. Existem vários tipos de operadores aritméticos que podem ser usados para realizar operações matemáticas com números e podemos usar os operadores de comparação em Python para comparar valores e retornar um valor *booleano* (*True* ou *False*).

Os símbolos de operações aritméticas são usados para realizar operações matemáticas em Python. Alguns dos principais operadores aritméticos são:

- Adição (+): No Código 4 o operador de adição é usado para somar dois valores.

```
1 a = 5
2 b = 3
3 c = a + b
4 print(c)
5 # Output: 8
```

Código 4: Soma de dois valores.

- Subtração (-): No Código 5 o operador de subtração é usado para subtrair um valor de outro.

```
1 a = 5
2 b = 3
3 c = a - b
4 print(c)
5 # Output: 2
```

Código 5: Subtração de dois valores.

- Multiplicação (*): No Código 6 o operador de multiplicação é usado para multiplicar dois valores.

```
1 a = 5
2 b = 3
3 c = a * b
4 print(c)
5 # Output: 15
```

Código 6: Multiplicação de dois valores.

- Potenciação (**): No Código 7 o operador de potência é usado para aumentar um número a uma potência.

```
1 a = 5
2 b = 3
3 c = a ** b
4 print(c)
5  # Output: 125
```

Código 7: Potenciação.

- Divisão (/): No Código 8 o operador de divisão é usado para dividir um valor por outro.

```
1 a = 5
2 b = 3
3 c = a / b
4 print(c)
5 # Output:  1.6666666666666667
```

Código 8: Divisão de dois valores.

- Módulo (%): No Código 9 o operador de módulo é usado para obter o resto da divisão de um número por outro.

```
1 a = 5
2 b = 3
3 c = a % b
4  imprimir (c)
5  # Saída: 2
```

Código 9: Resto de uma divisão.

É importante notar que os operadores aritméticos têm uma ordem de precedência, assim como na matemática. A ordem padrão é: primeiro, potência (**), seguido por multiplicação (*), divisão (/) e módulo (%), adição (+) e subtração (-). No entanto, podemos usar parênteses para forçar uma ordem diferente, se necessário como no Código 10.

```
1 a = 5
2 b = 3
3 c = 4
4 d = (a + b) * c
5 print(d)
6  #  Output: 32
```

Código 10: Ordem de uma operação na linguaguem de programação em Python.

Nesse exemplo, os parênteses são usados para forçar a adição antes da multiplicação. Abaixo, segue a descrição e exemplos dos principais operadores de comparação em Python:

- "==" (igual a): O Código 11 retorna *True* se os dois valores são iguais, *False* caso contrário.
-

```
1 a =  5
2 b =  5
3 print (a == b)
4   # Output: True
```

Código 11: Exemplo de um código que usa o aperador ==.

- "!=" (diferente de): O Código 12 retorna *True* se os dois valores são diferentes, *False* caso contrário.
-

```
1 a = 5
2 b = 3
3 print(a != b)
4  # Output: True
```

Código 12: Exemplo de um código que usa o aperador !=.

- ">" (maior que): O Código 13 retorna *True* se o primeiro valor é maior do que o segundo valor, *False* caso contrário.
-

```
1 a = 5
2 b = 3
3 print(a > b)
4 # Output: True
```

Código 13: Exemplo de um código que usa o aperador.

- "<" (menor que): O Código 14 retorna *True* se o primeiro valor é menor de que o segundo valor, *False* caso contrário.

```
1 a = 5
2 b = 3
3 print(a < b)
4 # Output: False
```

Código 14: Exemplo de um código que usa o aperador <.

- ">=" (maior ou igual a): O Código 15 retorna *True* se o primeiro valor é maior ou igual ao segundo valor, *False* caso contrário.

```
1 a = 5
2 b = 5
3 print(a >= b)
4  # Output: True
```

Código 15: Exemplo de um código que usa o aperador >=.

- "<=" (menor ou igual a): O Código 16 retorna *True* se o primeiro valor é menor ou igual ao segundo valor, *False* caso contrário.
-

```
1 a = 3
2 b = 5
3 print(a <= b)
4  # Output: True
```

Código 16: Exemplo de um código que usa o aperador <=.

Esses operadores são muito úteis em Python para comparar valores e tomar decisões com base nesses valores.

4.2 ESTRUTURA DE CONDICIONAL

Em Python, uma estrutura de condicional é usada para tomar uma decisão com base em uma condição. Guedes (2018, p.33) evidência:

> As decisões ajudam a selecionar quando uma parte do programa deve ser ativada e outras vezes deve ser ignorada. Na linguagem de programação Python existem basicamente 3 (três) formas de implementação de estruturas de controle com desvio de fluxo, são elas: a) “if”: estrutura de decisão na forma simples; b) “if-else”: estrutura de decisão na forma composta; e, c) “if-elif”: estrutura de decisão na forma encadeada.

Existem dois tipos principais de estruturas de condicional em Python:

- A condicional simples (*if)* e condicional composta *(if-else e if-elif-else).*
- A palavra-chave "*if*" é seguida por uma condição que deve ser avaliada como verdadeira ou falsa.
- A condição pode ser qualquer expressão que possa ser avaliada como verdadeira ou falsa.
- O bloco de código que segue a condição é executado se a condição for avaliada como verdadeira.
- O bloco de código deve ser recuado em quatro espaços, para indicar que ele faz parte do bloco "*if*".
- A condição deve estar entre parênteses.
- O bloco de código que segue o "*if*" deve estar recuado em quatro espaços.
- Se a condição de avaliação for verdadeira, o bloco de código será executado. Caso contrário, ele será ignorado.

```
1 idade =  20
2
3 if idade >=  18 :
4       print ( "Você é maior de idade!" )
```

Código 17: Exemplo de uma estrutura de condicional simples.

No Código 17, a condição é se a "idade >= 18", o programa imprime “Você é maior de idade!” logo a condição verdadeira porque a idade é igual a 20. O bloco de código que segue o "*if*" é executado, que neste caso é a impressão da mensagem "Você é maior de idade!".

É importante lembrar que, em uma estrutura de condicional simples, o bloco de código será executado apenas se a condição for verdadeira. Se a condição for falsa, o bloco de código será ignorado.

Estrutura condicional composta *if-else:*

- A palavra-chave "*if*" é seguida por uma condição que deve ser avaliada como verdadeira ou falsa.
- O bloco de código que segue a condição é executado se a condição for avaliada como verdadeira.
- A palavra-chave "*else*" é usada para definir um bloco de código que será executado se a condição for avaliada como falsa.
- O bloco de código que segue o "*else*" será executado se a condição for avaliada como falsa.

- A condição deve estar entre parênteses.
- O bloco de código que segue o "*if*" deve estar recuado em quatro espaços.
- O bloco de código que segue o "*else*" deve estar recuado em quatro espaços.

```
idade =  16

if idade >=  18 :
     print ( "Você é maior de idade!" )
 else :
     print ( "Você é menor de idade." )
```

Código 18: Exemplo de um código usando estrututa de condicional composta.

No Código 18, a condição é "idade >= 18", que é falsa porque a idade é igual a 16, que é menor que 18. O bloco de código que segue o "*else*" é executado, que neste caso é a impressão da mensagem "Você é menor de idade."

Estrutura condicional *if-elif-else*:

- A palavra-chave "*if*" é seguida por uma condição que deve ser avaliada como verdadeira ou falsa.
- O bloco de código que segue a condição é executado se a condição for avaliada como verdadeira.
- A palavra-chave "*elif*" é usada para adicionar condições adicionais. O bloco de código que segue o "*elif*" será executado se todas as condições anteriores forem falsas e a condição atual for verdadeira.
- Regras:
  - O bloco de código que segue o "*if*" deve estar recuado em quatro espaços.
  - O bloco de código que segue o "*elif*" deve estar recuado em quatro espaços.
  - O bloco de código que segue o "*else*" deve estar recuado em quatro espaços.

```
nota =  7

if nota >=  9 :
     print ( "Você foi aprovado com louvor!" )
 elif nota >=  7 :
     print ( "Você foi aprovado." )
 else :
     print ( "Você foi reprovado." )
```

Código 19: Exemplo de código usando estrututa de condicional composta usando *if-elif-else*.

O Código 19 verifica a nota de uma pessoa e informa se ela foi aprovada ou reprovada, além de fornecer uma mensagem personalizada de acordo com a nota obtida.

4.3 VETORES

4.3.1 Listas

Em programação, os vetores são estruturas de dados que armazenam coleções ordenadas de elementos do mesmo tipo. Em Python, uma lista é uma estrutura de dados que permite armazenar uma coleção ordenada de elementos "são sequências ordenadas de valores, as quais podem ser inseridas com dois colchetes, onde os valores que compõem essas listas são chamados de elementos ou itens" (Leitão, 2021, p.24). Os elementos de uma lista podem ser de qualquer tipo, como inteiros, *strings*, *booleanos*, entre outros. As listas são definidas por meio de colchetes [], e seus elementos são separados por virgulas como mostra o Código 20.

```
1 lista = [1, 2, 3, "quatro", Verdadeiro]
```

Código 20: Exemplo de uma lista em Python.

Nesse exemplo, a lista contém cinco elementos: os inteiros 1, 2 e 3, a *string* "quatro" e o valor *booleano True*. As listas em Python são mutáveis, ou seja, é possível modificar seus elementos, adicionar ou remover elementos. Algumas operações comuns em listas incluem:

- Acesso a elementos: é possível acessar elementos de uma lista por meio de seu índice numérico, que começa em 0. Por exemplo, para acessar o segundo elemento da lista acima (o número 2), podemos utilizar lista [1].
- Modificação de elementos: é possível modificar um elemento de uma lista atribuindo um novo valor a ele. Por exemplo, para modificar o terceiro elemento da lista acima (o número 3), podemos usar lista [2] = 4.
- Adição de elementos: é possível adicionar novos elementos ao final de uma lista utilizando o método *append*(). Por exemplo, para adicionar o número 5 à lista acima, podemos utilizar lista.append(5).
- Remoção de elementos: é possível remover elementos de uma lista utilizando os métodos remove() ou pop(). O método remove() remove a primeira ocorrência de um elemento na lista, enquanto o método pop() remove um elemento de uma posição

específica (ou a última posição, se nenhum índice for especificado). Por exemplo, para remover uma *string* "quatro" da lista acima, podemos usar lista.remove("quatro").

4.3.2 Tuplas

Uma tupla é uma estrutura de dados semelhante a uma lista, mas com uma diferença fundamental, as tuplas são imutáveis, ou seja, não é possível modificar seus elementos após a criação da tupla. Pedroso (2019, p. 27) enfatiza a definição das tuplas da seguinte forma:

> A tupla é uma lista imutável. O que diferencia uma tupla de uma lista é que uma lista pode ter elementos adicionados a ela (ou removidos) a qualquer momento, enquanto a tupla, após definida, não permite a adição ou remoção de elementos.

São definidas por meio de parênteses (), e seus elementos são separados por vírgulas. Por exemplo o Código 21.

```
1 tupla = (1, 2, 3, "quatro", True)
```

Código 21: Exemplo de uma tupla em Python.

Nesse exemplo, uma tupla contém cinco elementos: os inteiros 1, 2 e 3, uma *string* "quatro" e o valor *booleano True*.

As tuplas em Python são usadas quando se deseja garantir que os dados não sejam alterados ao longo do tempo. Por exemplo, se você tiver uma lista de coordenadas (x, y), pode usar uma tupla em vez de uma lista, pois essas coordenadas não mudarão. Outro exemplo comum é quando você tem um conjunto de constantes em seu programa que não deve ser alterado.

As tuplas em Python também são usadas em situações em que é necessário retornar vários valores de uma função. Em vez de retornar uma lista ou um dicionário, você pode retornar uma tupla com os valores desejados. Por exemplo, para remover uma *string* "quatro" da lista acima, podemos usar lista.remove("quatro").

4.3.3 Dicionários

Um dicionário é uma estrutura de dados que armazena uma coleção de pares chave-valor. Ao contrário das listas e tuplas, os elementos em um dicionário não são indexados por números inteiros, mas por meio de suas chaves. As chaves em um dicionário devem ser únicas e imutáveis (por exemplo, *strings* ou tuplas), enquanto os valores podem ser de qualquer tipo de dados. Segundo a autora Pedroso (2019, p. 27):

> O dicionário é uma estrutura de dados interna da linguagem Python. Ele consiste de um conjunto de pares da forma chave-valor, onde as chaves, que podem ser de qualquer tipo imutável, são únicas. Por ser um conjunto, seus elementos não seguem uma ordem específica.

Os dicionários são definidos por meio de chaves {} e pares chave-valor separados por dois pontos (:), e os pares são separados por vírgulas. Por exemplo o Código 22:

```
1 dicionario = {"nome": "João", "idade": 30, "cidade": "São Paulo"}
```

Código 22: Exemplo de um dicionário em Python.

Nesse exemplo, o dicionário contém três pares chave-valor: a chave "nome" tem o valor "João", a chave "idade" tem o valor 30 e a chave "cidade" tem o valor "São Paulo".

Os dicionários em Python são usados quando se deseja armazenar dados de forma organizada e acessá-los por meio de pensamentos chaves. Por exemplo, um dicionário pode ser usado para armazenar informações sobre pessoas em uma lista telefônica, onde o nome da pessoa é a chave e o número de telefone é o valor correspondente. Ou ainda, um dicionário pode ser usado para armazenar configurações de um programa, onde as chaves são as opções de configuração e os valores são os valores correspondentes.

Os dicionários em Python podem ser acessados por meio de suas chaves. Por exemplo, para obter o valor correspondente à chave "idade" no dicionário acima no Código 22, basta escrever como o Código 23:

```
1 idade = dicionario["idade"]
```

Código 23: Exemplo de um código para acessar o dicionário no Código 22.

4.4 ESTRUTURA DE REPETIÇÃO

As estruturas de repetição em Python são usadas para executar um bloco de código repetidamente, até que uma determinada condição seja verdadeira ou que um número definido de repetições seja alcançado. Existem duas estruturas de repetição principais em Python: o *loop* "*for*" e o *loop* "*while*".

O *loop* "*for*" é usado para iterar sobre uma sequência (lista, tupla, dicionário, conjunto, *string* etc.) e executar um bloco de código para cada item da sequência. Segundo Camargo *et al* (2022, p.28):

> O comando for permite percorrer os itens de uma coleção (por exemplo, uma string ou uma estrutura de dados do tipo lista). Para cada um dos itens da coleção, será executado o comando declarado na estrutura de repetição (loop). Ao percorrer a lista de valores, a variável definida no comando for receberá, a cada iteração, um dos itens da coleção.

```
1 for i in range(10):
2     print(i)
```

Código 24: Código usando o *loop* "*for*".

Por exemplo, o Código 24 imprime os números de 0 a 9.

O *loop* "*while*" é usado para executar um bloco de código repetidamente enquanto uma condição é verdadeira. Seguindo a linha de pensamento dos mesmos autores Camargo *et al* (2022, p. 31):

> O comando while faz com que um trecho de código seja repetido enquanto uma condição for verdadeira. Quando o resultado da condição for falso, a execução da repetição é interrompida (saindo do loop), passando para o próximo comando após o while.

```
1 i = 0
2 while i < 10:
3     print(i)
4     i += 1
```

Código 25: Código usando o loop "while".

Por exemplo, o Código 25 imprime os números de 0 a 9 usando um *loop* "*while*".

Para escrever o código das estruturas de repetição em Python incluem:

- Indentação: O bloco de código dentro do *loop* "*for*" ou "*while*" deve ser indentado com quatro espaços.
- Condição de termo: Em um *loop* "*while*", é importante ter uma condição de termo para evitar um *loop* infinito.
- Variável de iteração: Em um *loop* "*for*", uma variável de iteração deve ser definida para iterar sobre a sequência. É uma boa prática escolher um nome significativo para a variável.
- Sequência: Em um *loop* "*for*", a sequência a ser iterada deve ser definida. Isso pode ser uma lista, tupla, dicionário, conjunto, *string* etc.
- Range(): A função "range()" pode ser usada para gerar uma sequência de números em um *loop* "*for*". A sintaxe básica da função "range()" é "range (início, fim, passo)", onde "início" é o número de início, "fim" é o número de fim (não incluso) e "passo" é o intervalo entre os números.

Além dessas estruturas de repetição, Python oferece três palavras-chave adicionais que podem ser usadas dentro de um *loop* para modificar seu comportamento: "*break*", "*continue*" e "*pass*".

A palavra-chave "*break*" é usada para interromper imediatamente um *loop*. Quando encontrada dentro de um *loop*, ela encerra a execução do *loop* e faz com que o programa continue a partir do código que segue o *loop*. Por exemplo, o Código 26 usa um *loop* "*while*" para imprimir os números de 0 a 4, mas interrompe a execução quando o valor de i chega a 2:

```
1 i = 0
2 while i < 5:
3     print(i)
4     if i == 2:
5         break
6     i += 1
```

Código 26: Código usando a função “break.”

A saída desse código será: 0 1 2

A palavra-chave "*continue*" é usada para pular a execução de um *loop* único e continuar com a próxima iteração. Quando encontrada dentro de um *loop*, ela interrompe a

execução do bloco de instruções atual e continua com a próxima iteração. Por exemplo, o Código 27 usa um *loop* "*for*" para imprimir apenas os números combinados de 0 a 4.

```
for i in range(5):
    if i % 2 == 0:
        continue
    print(i)
```

Código 27: Código usando a função "*continue*".

A saída desse código será: 1 3

A palavra-chave "*pass*" é usada para indicar que nenhum código deve ser executado quando uma determinada condição é verdadeira. Ela é usada como um espaço reservado quando o código ainda não foi escrito. O Código 28 usa um *loop* "*for*" para imprimir os elementos de uma lista, mas usa a palavra-chave "*pass*" como um espaço reservado para um código que ainda não foi escrito:

```
lista = [1, 2, 3, 4, 5]
for elemento in lista:
    if elemento == 3:
        pass  #espaço reservado para o código que será adicionado
posteriormente
    else:
        print(elemento)
```

Código 28: Código usando a função "*pass*".

A saída desse código será: 1 2 4 5.

4.5 BIBLIOTECAS EM PYTHON QUE PODEM SER APLICADAS NA ESTATÍSTICA DESCRITIVA

Numpy e *Pandas* são duas bibliotecas importantes em Python, utilizadas principalmente em análise de dados e estatística descritiva. *Numpy* é uma biblioteca que oferece suporte para matrizes multidimensionais e operações matemáticas. Segundo Lima (2022, p.33):

> O numpy é uma biblioteca voltada para o processamento de dados em larga escala, que apresentam arranjos multidimensionais e de grande tamanho. Suas funções são aplicadas para análise estatística e matemática dos dados.

Enquanto a biblioteca *Pandas* Segundo a mesma autora (Lima, 2022, p.33):

> O pandas é uma biblioteca licenciada para uso em Python e de código aberto, sua ampla quantidade de funções são em sua maioria voltadas para análises estatísticas e exploratórias de dados.

É uma biblioteca que fornece estruturas de dados de alto nível e ferramentas de análise de dados. Ambas as bibliotecas são amplamente utilizadas na análise de dados, estatística descritiva e entre outros.

Lopes (2019) define que *numpy* e *pandas* têm funções que facilitam o cálculo das medidas de tendência central e podem ser usados para calcular a variância e o desvio padrão de um conjunto de dados em Python.

Com relação as medidas de tendência central, suponha que temos um conjunto de dados armazenados em uma lista ou *array numpy*. Por exemplo, a função *numpy.mean*() pode ser usada para calcular a média de um conjunto de dados, enquanto a função *pandas*. Series.mode() pode ser usada para calcular a moda.

O Código 29 mostra como calcular a mediana, *numpy* e *pandas*, vale ressaltar que elas têm funções separadas. A função *numpy.median*() pode ser usada para calcular a mediana de um conjunto de dados, enquanto a função *pandas*. *Series.median*() pode ser usada para calcular a mediana de uma série de *pandas*.

```
1  import numpy as np
2  import pandas as pd
3
4  data = [10, 20, 30, 40, 50, 50, 60]
5
6  # Calculando a média
7  mean = np.mean(data)
8  print("Média:", mean)
9
10 # Calculando a moda
11 mode = pd.Series(data).mode()
12 print("Moda:", mode[0])
13
14 # Calculando a mediana
15 median = np.median(data)
16 print("Mediana:", median)
```

Código 29: Código para calcular a média, moda e mediana de uma lista de dados usando as bibliotecas *numpy* e *pandas*.

O resultado pode ser observado no Código 30:

```
Média: 38.57142857142857
Moda: 50
Mediana: 40.0
```

Código 30: Resultado do cálculo das medidas de tendência central do Código 29.

Matplotlib é uma biblioteca de visualização de dados em Python que permite criar vários tipos de gráficos, incluindo gráficos de setores, linhas e barras. É uma das bibliotecas mais populares para visualização de dados em Python, é amplamente utilizada na análise de dados e na estatística descritiva. Lima (2022, p33) diz que:

> O matplotbit, tal qual o seaborn, é uma biblioteca de visualização de dados, seu conjunto de funções é voltado para a produção de gráficos 2D. Suas plotagens podem compor scripts Python para distribuição em diversas aplicações em Python e alimentar servidores web.

Os gráficos de setores, também conhecidos como gráficos de pizza, são usados para mostrar a distribuição de um conjunto de dados em categorias. Cada categoria é representada como uma fatia de pizza, com a área da fatia proporcional à proporção de valores nessa categoria “Os gráficos de setores, semelhantes a uma pizza, são representados pela divisão proporcional da área, dividida em fatias, de acordo com o percentual (frequência) de cada categoria” (Castro 2012, p.34).

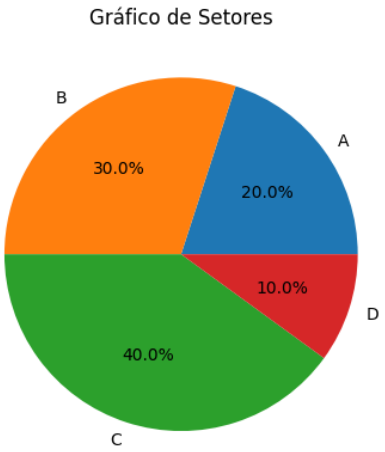
No Código 31 mostra um modo de criar um gráfico de setores usando o *Matplotlib*.

```python
import matplotlib.pyplot as plt

# Dados das categorias
categorias = ['A', 'B', 'C', 'D']
valores = [20, 30, 40, 10]

# Criando o gráfico de setores
fig, ax = plt.subplots()
ax.pie(valores, labels=categorias, autopct='%1.1f%%')

# Definindo o título do gráfico
ax.set_title('Gráfico de Setores')

# Exibindo o gráfico
plt.show()
```

Código 31: Criação de um gráfico de setores usando a função plt.pie() no Python.

Isso produzirá um gráfico de setores com quatro fatias, representando as categorias A, B, C e D como mostra a Figura 3.

Figura 3: Gráfico de Setores.



Fonte: Compilação da própria autora (2023).

Os gráficos de barras são usados para comparar valores entre diferentes categorias “Os gráficos de barras são utilizados para representar uma série de dados organizados em categorias, representados por barras (retângulos).” (Castro, 2022, p.33). Cada categoria é representada por uma barra, com a altura da barra representando o valor da categoria.

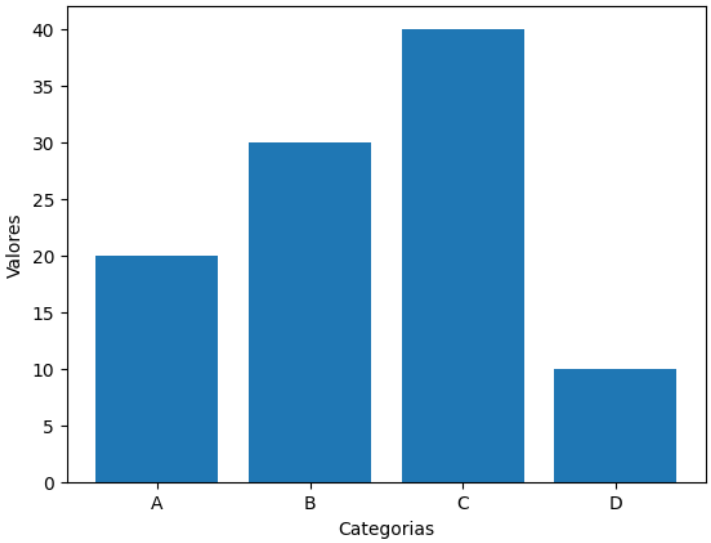
Para criar um gráfico de barras usando o *Matplotlib*, podemos usar a função plt.bar() como mostra o Código 32.

```
importar matplotlib.pyplot como plt

# Dados das categorias
categorias = [ 'A' , 'B' , 'C' , 'D' ]
valores = [ 20 , 30 , 40 , 10 ]

# Criando o gráfico de barras
plt . bar(categorias, valores)

# Definindo os títulos dos eixos
plt . xlabel( 'Categorias' )
plt . ylabel( 'Valores' )

# Exibindo o gráfico
plt . mostrar()
```

Código 32: Criação de um gráfico de barras usando a função plt.bar() no Python.

Isso produzirá um gráfico de barras com quatro barras assim como na Figura **4**, representando as categorias A, B, C e D.

Figura 4: Gráfico de barras



Fonte: Compilação da própria autora (2023).

Um histograma é uma representação gráfica que exibe a distribuição dos dados em uma determinada variável. Segundo Penedo et al (2020, p.19) "é um gráfico de barras que mostra a distribuição de dados por categorias" e consiste em um conjunto de barras adjacentes, onde a largura de cada barra representa um intervalo de valores e a altura indica a frequência com que esses valores ocorrem.

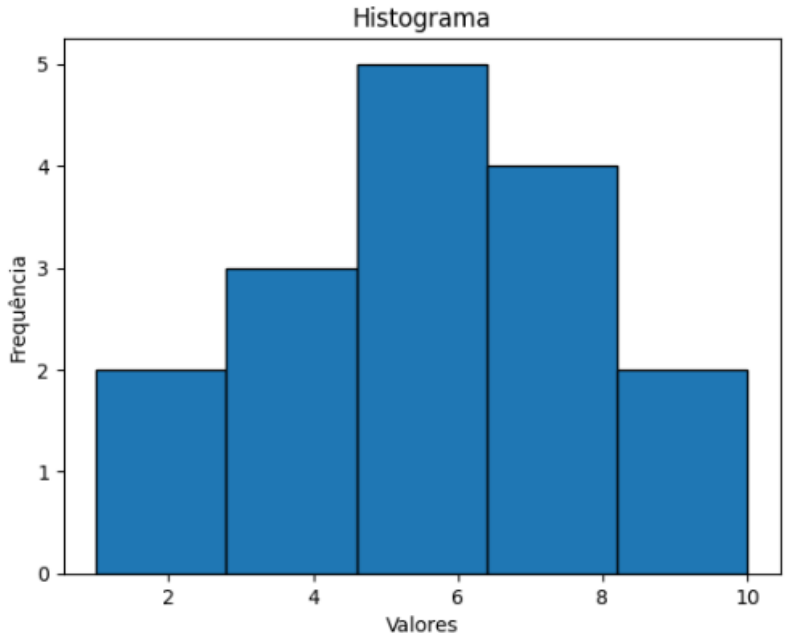
No Código 33 mostra um modo de criar um histograma usando o *Matplotlib*.

```
import matplotlib.pyplot as plt
# Dados de exemplo
dados = [1, 2, 3, 3, 4, 5, 5, 5, 6, 6, 7, 8, 8, 8, 9, 10]
# Criando o histograma
plt.hist(dados, bins=5, edgecolor='black')
# Configurando o título e os rótulos dos eixos
plt.title("Histograma")
plt.xlabel("Valores")
plt.ylabel("Frequência")
# Exibindo o histograma
plt.show()
```

Código 33: Criação de um histograma usando a função plt.show() no Python.

Ao executar esse código, produzirá um histograma, como na Figura 5 onde as barras representam a frequência dos valores nos intervalos especificados.

Figura 5: Histograma



Fonte: Compilação da própria autora (2023).

Um boxplot, também conhecido como diagrama de caixa, é um gráfico estatístico que representa a distribuição de um conjunto de dados numéricos por meio de quartis. É uma ferramenta útil para visualizar a variação, assimetria e outliers nos dados. Neto et al (2023, p.1) ressalta que "O boxplot é um tipo de gráfico usado regularmente na pesquisa científica e a sua construção é possível por meio de diversos softwares estatísticos".

No Código 34 mostra um modo de criar um boxplot usando o *Matplotlib*.

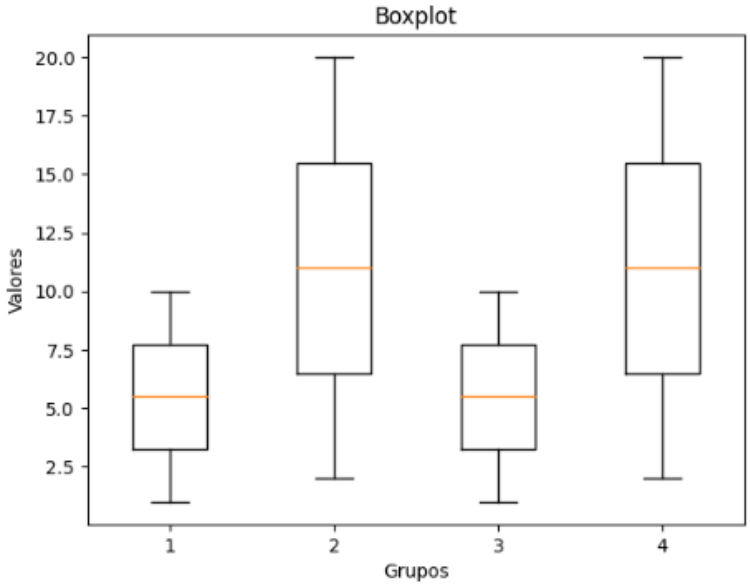
```
import matplotlib.pyplot as plt
# Dados de exemplo
dados1 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
dados2 = [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
dados3 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
dados4 = [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
# Criando o boxplot
plt.boxplot([dados1, dados2, dados3, dados4])
# Configurando o título e os rótulos dos eixos
plt.title("Boxplot")
plt.xlabel("Grupos")
plt.ylabel("Valores")
# Exibindo o boxplot
plt.show()
```

Código 34: Criação de um boxplot usando a função plt.show() no Python.

Isso produzirá o boxplot como na Figura 6 onde cada caixa representa um grupo de dados. O retângulo central da caixa indica a mediana, a linha que o atravessa representa a

mediana. A parte inferior da caixa indica o primeiro quartil (25% dos dados) e a parte superior indica o terceiro quartil (75% dos dados).

Figura 6: Boxplot



Fonte: Compilação da própria autora (2023).

4.6 FUNÇÕES

Uma função é um bloco de código que pode ser chamado repetidamente a partir de diferentes partes do programa. Em Python, podemos criar nossas próprias funções usando a palavra-chave def. Banin (2018, p.200) diz que:

> Em Python, uma função é definida por meio de um cabeçalho que contém quatro elementos: a palavra reservada def, um nome válido, parâmetros entre parênteses e o caractere “:”. Esse cabeçalho é sucedido por um bloco de comandos identados que constitui o corpo da função.

Para criar uma função em Python, primeira precisa definir seu nome e quais argumentos ela receberá. Em seguida, podemos adicionar o código que desejamos que a função execute e, finalmente, retornar um valor se necessário.

```
def media(lista):
    soma = 0
    for num in lista:
        soma += num
    return soma / len(lista)
```

Código 35: Criação de uma função que calcula a média de uma lista de números em Python.

No Código 35 definimos um argumento lista, que é a lista de números para a qual queremos calcular a média. Em seguida, criamos uma variável soma e usamos um *loop for* para somar todos os números da lista. Finalmente, dividimos a soma pelo comprimento da lista e retornamos o resultado.

No Código 36 podemos testar nossa função com uma lista de números:

```
números = [2, 4, 6, 8, 10]
resultado = média(números)
print(resultado)
```

Código 36: Código onde mostra o teste função que foi criada no Código 35 com uma lista de números.

Assim como em medidas de tendência central, em medidas de dispersão também podemos criar funções em Python para facilitar o cálculo desses valores em diferentes partes do nosso programa.

Uma medida de dispersão comum é o desvio padrão, que é uma medida de quão distantes os valores de uma distribuição estão na média. Podemos criar uma função em Python para calcular o desvio padrão de uma lista.

```
import math

def desvio_padrao(lista):
    media = sum(lista) / len(lista)
    soma_quadrados = sum((x - media) ** 2 for x in lista)
    variancia = soma_quadrados / (len(lista) - 1)
    return math.sqrt(variancia)
```

Código 37: Criação de uma função que calcula desvio padrão de uma lista de números em Python.

No Código 37, importamos o módulo *math* para usar a função sqrt, que calcula a raiz quadrada de um número. Em seguida, definimos um argumento lista, que é a lista de números para a qual queremos calcular o desvio padrão. Dentro da função, calculamos a média da lista usando a função *sum* dividindo pelo comprimento da lista. Em seguida, usamos um *loop for* para calcular a soma dos quadrados das diferenças entre cada número da lista e a média. Em

seguida, dividimos essa soma pelo grau de liberdade (n-1) para obter a variação amostral. Finalmente, retornamos à raiz quadrada da variação para obter o desvio padrão.

Podemos testar nossa função com uma lista de números como mostra o Código 38:

```
números = [2, 4, 6, 8, 10]
resultado = desvio_padrão(números)
print(resultado)
```

Código 38: Lista de números.

Isso deve imprimir 2.8284271247461903, que é o desvio padrão dos números em nossa lista.

4.7 IMPORTAÇÃO DE ARQUIVOS

Para importar um arquivo em Python, você pode usar uma declaração *import*. Isso permite que você use as funções e classes definidas no arquivo em seu programa.

No Código 39 está um exemplo de como importar a biblioteca *Numpy* e usar suas funções para calcular medidas de tendência central:

```
 import numpy as np

# Criando uma lista de números
numeros = [1, 2, 3, 4, 5]

# Calculando a média usando NumPy
media = np.mean(numeros)

# Calculando a mediana usando NumPy
mediana = np.median(numeros)

# Imprimindo a média e a mediana
print('Média:', media)
print('Mediana:', mediana)
```

Código 39: Exemplo de como calcular a média de uma lista de números usando a biblioteca *Numpy*.

Neste exemplo, a biblioteca *NumPy* é importada usando uma declaração *import*, seguida pelo nome da biblioteca, que neste caso é *numpy*, com o apelido "np". O “np” é usado para facilitar o acesso às funções da biblioteca. Em seguida, criamos uma lista de números chamada "numeros" com os valores [1, 2, 3, 4, 5]. Usando a função *mean* do *NumPy*, calculamos a média dos valores da lista e armazenamos o resultado na variável "média". Usando a função *median* do *NumPy*, calculamos a mediana dos valores da lista e armazenamos o

resultado na variável "mediana". Por fim, imprimimos a média e a mediana usando a função *print*.

Em seguida, uma lista de números é criada e mantida na variável números. A função *mean*() da biblioteca *NumPy* é usada para calcular a média dos números na lista, e a função *median*() é usada para calcular a mediana.

No Código 40 um exemplo de como calcular a variância de uma lista de números usando a biblioteca *NumPy* do Python:

```
import numpy as np

# Criando uma lista de números
numeros = [1, 2, 3, 4, 5]

# Calculando a variância usando NumPy
variancia = np.var(numeros)

# Imprimindo a variância
print(variancia)
```

Código 40: Exemplo de como calcular a variância de uma lista de números usando a biblioteca *Numpy*.

Este exemplo importa a biblioteca *NumPy*, cria uma lista de números e calcula uma variação usando a função var() da biblioteca *NumPy*. O resultado é armazenado na variável variância e é impresso na tela.

No Código 41 está um exemplo de como calcular o desvio padrão de uma lista de números usando a biblioteca *NumPy* do Python:

```
import numpy as np

# Criando uma lista de números
numeros = [1, 2, 3, 4, 5]

# Calculando o desvio padrão usando NumPy
desvio_padrao = np.std(numeros)
```

Código 41: Exemplo de como calcular o desvio padrão de uma lista de números usando a biblioteca *NumPy*.

5 UMA NOVA PERSPECTIVA DA PROGRAMAÇÃO EM PYTHON NO ENSINO DA ESTATÍSTICA DESCRITIVA NO ENSINO MÉDIO

A utilização de tecnologias educacionais como instrumento de ensino tem sido uma tendência crescente na área da educação. Entre essas tecnologias, a programação em Python tem se destacado como uma ferramenta pedagógica importante no ensino da Matemática, especialmente no contexto da estatística descritiva. Ao introduzir a análise de dados, a programação em Python proporciona uma nova perspectiva no aprendizado matemático, capacitando os alunos a adquirir habilidades relevantes para diversas áreas de atuação.

Ao utilizar conceitos básicos de programação em Python, como variáveis, estruturas de controle de fluxo, funções e bibliotecas, os alunos são capacitados a compreender os processos envolvidos na análise de dados. No entanto, é crucial que eles também compreendam a teoria estatística subjacente a esses conceitos, a fim de aplicá-la corretamente.

Nesse sentido, é fundamental propor atividades que estimulem o pensamento lógico, a socialização, o desenvolvimento cognitivo e o pensamento matemático dos alunos. Essas atividades têm como objetivo auxiliá-los a compreender a teoria estatística e a aplicá-la corretamente em suas análises de dados. Dessa forma, promove-se uma abordagem holística do ensino, que combina o uso da programação em Python com uma compreensão sólida dos princípios estatísticos, preparando os alunos para enfrentar desafios reais e desenvolver habilidades relevantes para o mundo contemporâneo.

5.1 PRIMEIRA ATIVIDADE: UMA NOVA FORMA DE INVESTIGAR

Objetivo dessa atividade é introduzir o conceito de mediana e sua importância na análise de dados, desenvolver habilidades práticas em programação em Python e estatística descritiva, promovendo o pensamento crítico e a resolução de problemas.

Materiais necessários: Computador com acesso ao Google *Colab*, para realizar uma análise dos dados usando a linguagem de programação Python; acesso à internet, para buscar informações referente ao problema proposto. Materiais de escrita, como lápis e papel, para anotações e registro das análises. Se for o caso, materiais para apresentação dos resultados da análise, como um projetor ou uma lousa.

A atividade terá início com a apresentação de um problema específico pelo professor. Por exemplo, ele pode apresentar um conjunto de dados sobre a variação de poluição produzida nos últimos 5 anos no Brasil e pedir aos alunos para calcular a mediana que é um importante indicador estatístico.

Em seguida, introduzirá o conceito de mediana e explicará como realizar o cálculo para encontrar a mediana, que é uma medida de tendência central que representa o valor central de um conjunto de dados ordenados. A fórmula para calcular a mediana é a seguinte:

- Se o número de dados é ímpar, a mediana é o valor central da lista ordenada.
- Se o número de dados é par, a mediana é a média dos dois valores centrais da lista ordenada.

Dessa forma, o professor trabalhará com os alunos a capacidade de analisar e interpretar dados numéricos. Em seguida será introduzido também a construção de um gráfico de setores (também conhecido como gráfico de pizza) como uma forma visual de representar a distribuição dos dados.

A introdução deste conceito está de acordo com a Base Nacional Comum Curricular (BNCC), que sugere o desenvolvimento de habilidades estatísticas e probabilísticas pelos estudantes. Depois o professor mostrará como fazer o mesmo cálculo e o gráfico utilizando a linguagem de programação Python. Essa etapa é importante, pois a tecnologia faz parte do cotidiano dos alunos, e o uso de ferramentas computacionais pode tornar o aprendizado mais atrativo e dinâmico, incentivando a participação e o engajamento dos estudantes.

Para aprofundar o aprendizado, o professor poderá dividir os alunos em grupos e fornecer a cada grupo um conjunto de dados diferente. Os alunos deverão utilizar Python para calcular a mediana do conjunto de dados atribuído a seu grupo e construir o gráfico de setores.

Depois que todos os grupos obtiveram a mediana, o professor promove uma discussão em grupo sobre os resultados obtidos. Os alunos devem compartilhar como resolveram o problema e discutir as semelhanças e diferenças entre os diferentes conjuntos de dado e representar visualmente esses dados por meio de um gráfico de setores. O professor propõe uma atividade em que os alunos devem coletar dados de uma pesquisa de interesse do grupo e usar Python para calcular a mediana dos resultados e representar visualmente esses dados por meio de um gráfico de setores.

```python
import numpy as np

poluicao = [45.2, 32.1, 54.8, 39.6, 47.3]
mediana = np.median(poluicao)

print("A mediana da poluição nos últimos 5 anos é:", mediana)
```

Código 42: Cálculo da mediana dos dados coletados pelos alunos.

O Código 42 é um exemplo de como calcular a mediana de um conjunto de dados, que pode ser criada através da fórmula, será introduzida no início da aula. É importada uma biblioteca *NumPy*, depois será criada uma lista chamada poluição, que contém os valores de poluição nos últimos 5 anos. Neste caso, foram utilizados valores fictícios para fins de exemplo. Em seguida, é utilizada a função np.median() da biblioteca *NumPy* para calcular a mediana da lista poluição. A mediana é um valor que representa o valor central de um conjunto de dados ordenados, ou seja, é o valor que divide os dados em duas partes iguais. Por fim, o valor da mediana é armazenado na variável mediana e é exibido na tela por meio da função print(). A mensagem que é exibida na tela é "A mediana da poluição nos últimos 5 anos é:", seguida do valor da mediana.

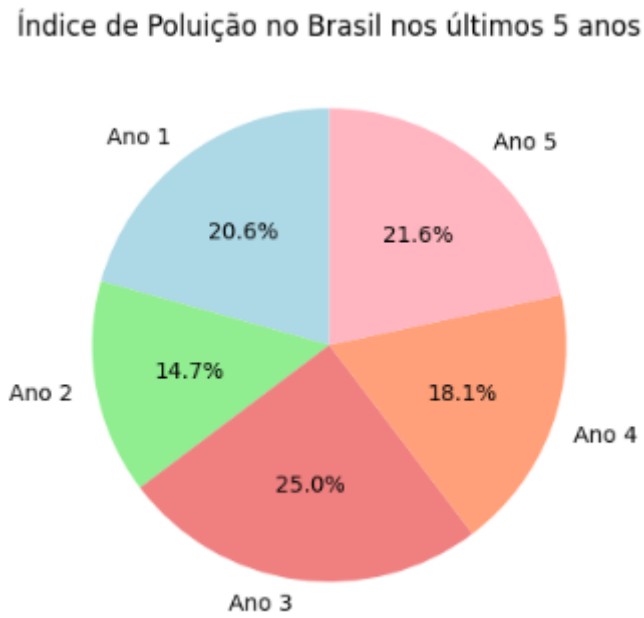
Para criar um gráfico de setores (ou gráfico de pizza) com base nos índices de poluição no Brasil nos últimos 5 anos, os alunos podem usar a biblioteca Matplotlib em Python como no Código 43.

```python
import matplotlib.pyplot as plt
poluicao = [45.2, 32.1, 54.8, 39.6, 47.3]
# Configurando rótulos e cores das fatias
labels = ['Ano 1', 'Ano 2', 'Ano 3', 'Ano 4', 'Ano 5']
colors = ['lightblue', 'lightgreen', 'lightcoral', 'lightsalmon', 'lightpink']
# Criando o gráfico de setores
plt.pie(poluicao, labels=labels, colors=colors, autopct='%1.1f%%',
startangle=90)
# Configurando o título
plt.title("Índice de Poluição no Brasil nos últimos 5 anos")
# Exibindo o gráfico
plt.show()
```

Código 43: Exemplo de um gráfico de setores com os dados do índice da poluição no Brasil nos últimos 5 anos.

Esse código produzirá, um gráfico de setores representando os índices de poluição nos últimos 5 anos. Cada parte destacada representa um ano e a porcentagem correspondente como na Figura 7 .

Figura 7: Gráfico de setores



Fonte: Compilação da própria autora (2023).

O projeto final será avaliado considerando a clareza, a organização e a criatividade na apresentação dos resultados. Será avaliado se a apresentação foi estruturada de forma lógica e coerente e se os alunos souberam, de forma eficaz, como prosseguir a partir dos dados analisados.

Em suma, a avaliação será baseada em uma combinação de critérios, incluindo a precisão dos cálculos, a colaboração em grupo, o uso de problemas contextualizados e a apresentação do projeto final. O objetivo é avaliar não apenas o conhecimento dos alunos em estatística descritiva e programação em Python, mas também suas habilidades de trabalho em equipe e comunicação.

Essa atividade busca tornar o aprendizado da estatística descritiva mais lúdico e interativo, ao mesmo tempo em que desenvolve habilidades práticas em programação em Python, promove o pensamento crítico e o uso de problemas contextualizados. Além disso, a atividade incentiva a aplicação dos conceitos aprendidos em problemas reais, como na proposta de pesquisa na atividade.

5.2 SEGUNDA ATIVIDADE: ANALISANDO UM PROBLEMA REAL NO ENSINO MÉDIO - TABELA EM PYTHON PARA REGISTRO DOS DADOS

Essa atividade tem como objetivo principal desenvolver habilidades dos alunos relacionadas à análise e interpretação de dados, bem como ao cálculo e compreensão das medidas de tendência central e proporcionar aos alunos a oportunidade de aplicar esses conceitos em situações concretas, utilizando exemplos práticos do cotidiano escolar.

Os materiais necessários para essa atividade incluem: Computador com acesso ao Google *Colab*, para realizar uma análise dos dados usando a linguagem de programação Python; dados das avaliações antigas de matemática do ensino fundamental. Materiais de escrita, como lápis e papel, para anotações e registro das análises. Se for o caso, materiais para apresentação dos resultados da análise, como um projetor ou uma lousa.

O professor explicará aos alunos o que são as medidas de tendência central e qual a importância delas para a estatística descritiva, tendo como exemplo as notas em matemática. Além de calcular as medidas de tendência central, como a média, mediana e moda, o professor pode introduzirá aos alunos a construção de um gráfico de barras como uma forma visual de representar a distribuição dos dados de forma comparativa.

O gráfico de barras é usado para mostrar a quantidade ou frequência de cada categoria em um conjunto de dados. Nesse caso, cada categoria será representada por uma barra, cuja altura ou comprimento será proporcional à quantidade ou frequência correspondente. Para isso, os alunos podem coletar para os dados suas notas de matemática de provas ou trabalhos anteriores. Em seguida, os alunos devem registrar suas notas em uma tabela em Python, depois calcular a média, a mediana e a moda desses dados.

Suponha que a média das notas dos alunos seja 7,5, a mediana seja 8 e a moda seja 9. Com esses dados, pode-se inferir que a maioria dos alunos obteve notas acima da média e da mediana, sendo que a nota 9 é a mais frequente.

Em seguida os alunos vão representar visualmente esses dados por meio de um gráfico de barras. Isso contribuirá claramente para o desenvolvimento de habilidades estatísticas e de interpretação de informações, além de estimular o pensamento crítico e a capacidade de comunicar dados de forma e eficaz.

Desse modo os alunos serão divididos em grupos e cada grupo receberá uma tabela em Python para registrar as informações coletadas durante a atividade. Com os dados coletados, cada grupo deve utilizar a biblioteca *NumPy* em Python para calcular a média, moda e mediana

das avaliações dos membros do próprio grupo. Depois, os grupos devem apresentar seus resultados e discutir as possíveis razões das diferenças encontradas entre eles.

Para tornar a atividade mais competitiva, os alunos podem ser divididos em grupos e cada grupo pode coletar dados sobre as notas em matemática de uma turma diferente. Em seguida, cada grupo pode calcular a média, a mediana e a moda das notas coletadas. O grupo que apresentar os resultados mais precisos receberá uma pontuação maior, incentivando a competição saudável entre os alunos.

Os dados obtidos podem ser úteis em diferentes situações, por exemplo, a escola pode utilizar essa informação para tomar decisões sobre a forma de ensino da matéria, identificando os tópicos em que a maioria dos alunos apresenta mais dificuldades. Os professores podem utilizar esses dados para entender melhor o desempenho de seus alunos e adaptarem suas aulas de acordo com suas necessidades.

Segue abaixo um exemplo do Código 44 em Python para calcular a média, moda e mediana da nota dos alunos da sala de aula usando a biblioteca *NumPy*.

```
import numpy as np

# Definindo os dados coletados (notas de avaliações antigas de
matemática)
notas_alunos = [8.0, 7.5, 6.0, 9.0, 7.0, 8.5, 9.5, 6.5, 8.0, 7.0, 8.0]

# Calculando a média
media = np.mean(notas_alunos)

# Calculando a moda
moda = np.mode(notas_alunos)
# Calculando a mediana
mediana = np.median(notas_alunos)

# Exibindo os resultados
print("Média: ", media)
print("Moda: ", moda)
print("Mediana: ", mediana)
```

Código 44: Cálculo da média, moda e mediana das avaliações de matemática dos alunos.

Primeiro, é importada a biblioteca *NumPy*, que é utilizada para trabalhar com operações matemáticas em *arrays.* Em seguida, a lista de notas dos alunos é definida, foram utilizados valores fictícios para fins de exemplo. A média aritmética é continuada com a função "np.mean()", que recebe uma lista de notas como argumento e retorna a média. A moda é continuada com a função "np.mode()", que recebe uma lista de notas como argumento e retorna o valor mais frequente. Já a mediana é seguida com a função "np.median()", que também recebe

uma lista de notas como argumento e retorna o valor central. Por fim, os resultados são exibidos com a função "*print*()", mostrando a média, moda e mediana.

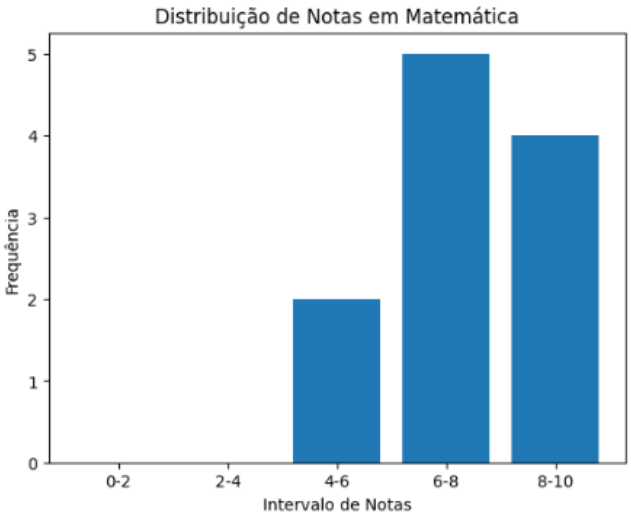
Para criar um gráfico de barras baseado nos dados de notas de matemática, os alunos podem utilizar o Código 45.

```
import matplotlib.pyplot as plt
# Definindo as categorias (intervalos de notas)
categorias = ['0-2', '2-4', '4-6', '6-8', '8-10']
# Definindo as frequências de cada categoria
frequencias = [0, 0, 2, 5, 4]
# Criando o gráfico de barras
plt.bar(categorias, frequencias)
# Adicionando rótulos e título ao gráfico
plt.xlabel('Intervalo de Notas')
plt.ylabel('Frequência')
plt.title('Distribuição de Notas em Matemática')
# Exibindo o gráfico
plt.show()
```

Código 45: Exemplo de um gráfico de barras com as notas das avaliações de matemática anteriores dos alunos.

Ao rodar o código completo, os alunos obterão um gráfico de barras como na Figura 8 que representa a distribuição das notas em matemática nos intervalos definidos. Cada barra corresponderá a um intervalo de notas, e a altura da barra representará uma frequência (quantidade de notas) em cada intervalo.

Figura 8: Gráfico de barras da distribuição das notas em matemática



Fonte: Compilação da própria autora (2023).

A avaliação dos alunos será baseada na qualidade da coleta e análise dos dados, bem como na precisão dos cálculos realizados. Além disso, a capacidade dos alunos em discutir

e interpretar os resultados obtidos também será levada em consideração. O professor pode avaliar cada grupo através de uma apresentação dos resultados, onde os alunos deverão explicar como realizaram a coleta e análise dos dados, bem como justificar seus resultados e discutir as possíveis razões das diferenças encontradas entre os grupos. Também pode ser atribuída uma pontuação para cada etapa do processo, como a coleta dos dados, a precisão dos cálculos e a interpretação dos resultados.

A atividade contribui para o desenvolvimento de habilidades matemáticas e científicas dos alunos, bem como para a compreensão de como as medidas de tendência central podem ser úteis em diferentes situações do cotidiano escolar. Além disso, a competição saudável entre os alunos pode tornar a atividade mais dinâmica e envolvente, incentivando o engajamento e a participação ativa dos estudantes.

5.3 TERCEIRA ATIVIDADE: TABELA EM PYTHON PARA REGISTRO DOS DADOS

Essa atividade utiliza a estrutura de dados em Python para aprender estatística descritiva em medidas de dispersão. O objetivo é ensinar aos alunos a importância do desvio padrão e variância para a análise de dados e como elas podem ser aplicadas em situações reais. Por exemplo, utilizando dados reais de desmatamento no Amazonas e com isso criar uma tabela em Python para coletar e analisar esses dados.

Materiais necessários: Computador com acesso ao Google *Colab*, para realizar uma análise dos dados usando a linguagem de programação Python; acesso à internet para buscar os dados de desmatamento no Amazonas. Materiais de escrita, como lápis e papel, para anotações e registro das análises. Se for o caso, materiais para apresentação dos resultados da análise, como um projetor ou uma lousa.

A metodologia proposta começa com uma introdução teórica e uma análise do problema proposto, visto que, a degradação da floresta amazônica é um problema ambiental que preocupa o mundo inteiro, e as diferentes porcentagens de área desmatada ao longo dos anos podem ser explicadas por diversos fatores.

Para entender melhor essas razões, os alunos irão colher dados sobre o desmatamento na região amazônica e analisá-los em grupos, utilizando ferramentas de busca na internet. Durante uma análise, eles podem perceber que os dados apresentam diferentes níveis de dispersão, ou seja, há anos em que o desmatamento foi muito baixo e outros em que foi muito alto. Em seguida os alunos serão divididos em grupos e terão como suporte a internet e ferramentas de busca para coletar dados adicionais sobre o desmatamento na região

Amazônica. Essa coleta de dados é fundamental para complementar a análise e entender as possíveis razões para as diferentes porcentagens de área desmatada ao longo dos anos. Os alunos podem buscar informações sobre as políticas públicas para a conservação da floresta, o avanço das atividades agropecuárias e de mineração, além de desastres naturais e eventos climáticos extremos, como secas e incêndios florestais. Com essas informações, os grupos podem identificar tendências e padrões nos dados, permitindo uma análise mais precisa e completa.

Além disso, as medidas de dispersão, como o desvio padrão, podem auxiliar na interpretação dos dados coletados. O desvio padrão é uma medida estatística que indica a dispersão dos valores em relação à média. Um desvio padrão alto indica que os valores estão mais dispersos, ou seja, há uma variação maior entre eles. Já um desvio padrão baixo indica que os valores estão mais próximos da média, ou seja, há uma variação menor entre eles. Além da análise de distribuição, os alunos também utilizarão o histograma como uma ferramenta para visualizar a distribuição dos dados. O histograma é um gráfico de barras que mostra a frequência com que determinados valores ocorrem em um conjunto de dados. Essa representação visual permite identificar padrões e tendências nos dados de desmatamento ao longo dos anos.

Para calcular o desvio padrão em Python, é necessário primeiro coletar dados e calcular a média aritmética da variável em questão. Seguindo uma atividade proposta pelo professor, os alunos já terão coletado e organizado os dados em uma tabela em Python como é mostrado no Código 46 para criação da tabela.

```
# Importando a biblioteca Pandas para manipulação de dados em tabelas
import pandas as pd

# Criando a tabela de registro dos dados
dados = {'Ano': [2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018,
2019],'Porcentagem de desmatamento': [11.15, 13.69, 17.36, 20.03,
22.05, 25.95, 29.26, 32.44, 35.14, 37.68]}
tabela = pd.DataFrame(dados)

# Calculando a média, variância e desvio padrão
media = tabela['Porcentagem de desmatamento'].mean()
variancia = tabela['Porcentagem de desmatamento'].var()
desvio_padrao = tabela['Porcentagem de desmatamento'].std()
# Imprimindo os resultados
print('Média: {:.2f}'.format(media))
print('Variância: {:.2f}'.format(variancia))
print('Desvio padrão: {:.2f}'.format(desvio_padrao))
```

Código 46: Código do cálculo da média e desvio padrão dos dados coletados pelos alunos sobre o desmatamento na Amazônia.

O Código 46 tem como objetivo calcular a média, a variância e o desvio padrão de uma tabela que contém dados sobre o desmatamento na Amazônia nos anos de 2010 a 2019. Foram utilizados valores fictícios para fins de exemplo.

A primeira linha do código importa a biblioteca *Pandas*, que é uma biblioteca muito utilizada para manipulação de dados em mesas. Em seguida, a tabela de registro dos dados é criada com os dados de ano e porcentagem de desmatamento.

Depois, a média, a variância e o desvio padrão são calculados a partir dos dados da coluna 'Porcentagem de desmatamento' da tabela criada anteriormente.

A linha seguinte, "*print*('Média: {:.2f}'.format(media))", imprime a média com duas casas decimais após a vírgula, utilizando a função ".format()" para formatar a saída da impressão. As linhas a seguir fazem a mesma coisa para a variância e o desvio padrão.

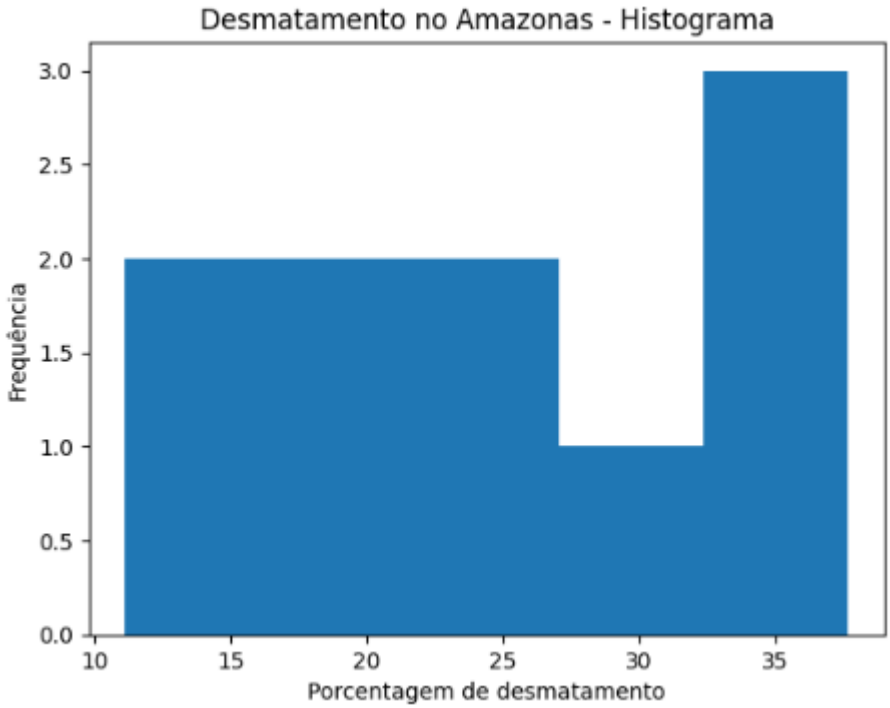
Para criar um histograma baseado nos dados de desmatamento no Amazonas, utilizando os dados fornecidos os alunos podem usar esse o exemplo do Código 47.

```
importar  matplotlib.pyplot  como  plt

# Dados de desmatamento
ano = [ 2010 , 2011 , 2012 , 2013 , 2014 , 2015 , 2016 , 2017 , 2018 , 2019 ]
%_desmatamento = [ 11.15 , 13.69 , 17.36 , 20.03 , 22.05 , 25.95 , 29.26 , 32.44 ,
35.14 , 37.68 ]

# Criando o histograma
plt . hist(porcentagem_desmatamento, bins = 'auto' )

# Adicionando rótulo aos eixos
plt . xlabel( 'Porcentagem de desmatamento' )
plt . ylabel( 'Frequência' )

# Definindo o título do gráfico
plt . title( 'Desmatamento no Amazonas - Histograma' )

# Exibindo o histograma
plt . mostrar()
```

Código 47: Exemplo de um histograma o o índice de desmatamento da Amazônica nos últimos 10 anos.

Em seguida quando os alunos forem executar o código completo, obterão um histograma que representa a distribuição dos dados de desmatamento no Amazonas como mostra a Figura 9.

Figura 9: Desamatamento do Amazônia – Histograma



Fonte: Compilação da própria autora (2023).

A avaliação dos alunos será feita de forma contínua durante todo o processo da atividade. Primeiramente, será avaliada a compreensão dos alunos em relação ao problema proposto e a capacidade de buscar informações relevantes sobre o desmatamento na região amazônica. Ao final da atividade, os alunos serão avaliados com base em sua participação, desempenho, aprendizado e compreensão dos conceitos e técnicas trabalhadas na atividade.

Durante a análise dos dados em grupo, será avaliada a capacidade dos alunos em trabalhar em equipe, compartilhar ideias e analisar os dados de forma crítica e reflexiva. Também será avaliado a capacidade dos alunos em utilizar a linguagem Python para coletar e organizar os dados em uma tabela, bem como para calcular o desvio padrão e interpretar os resultados obtidos.

Com essa atividade, os alunos poderão desenvolver habilidades em análise de dados, trabalho em grupo e o uso de problemas contextualizados, além de aplicar conceitos de estatística descritiva e programação em Python. Permitindo que os alunos discutam e comparem os resultados obtidos, analisando as possíveis causas das diferenças de compatibilidade e o impacto que o desmatamento pode causar na região amazônica. Dessa forma, a atividade não só contribui para o desenvolvimento do pensamento lógico-matemático dos alunos, como também para a conscientização ecológica e social.

5.4 QUARTA ATIVIDADE: DADOS REAIS PARA GESTÃO FINACEIRA.

A atividade proposta tem como objetivo promover a educação financeira dos alunos por meio da análise dos dados financeiros de suas próprias famílias. Para isso, será utilizada uma abordagem prática, na qual os alunos terão acesso a um conjunto de dados reais de gastos e receitas da sua própria família. A partir dessa análise, busca-se desenvolver a compreensão dos alunos sobre a importância da gestão financeira pessoal e os impactos que essa gestão pode ter na vida financeira das pessoas. A atividade tem como objetivo fomentar o uso de ferramentas tecnológicas, como a linguagem de programação Python, para a análise e interpretação de dados financeiros.

Materiais necessários: Computador com acesso ao Google *Colab*, para realizar uma análise dos dados usando a linguagem de programação Python; dados financeiros da família de cada aluno nos últimos 5 meses, como gastos e receitas de diferentes períodos. Materiais de escrita, como lápis e papel, para anotações e registro das análises. Se for o caso, materiais para apresentação dos resultados da análise, como um projetor ou uma lousa.

Para realizar uma atividade, será necessário um conjunto de dados com informações sobre os gastos e receitas da família dos alunos. Esse conjunto de dados pode ser fornecido pelos próprios alunos ou pelo professor, caso seja necessário preservar a privacidade dos dados pessoais.

A análise dos dados por meio do desvio padrão e variância permitirá aos alunos identificar padrões de comportamento financeiro de sua família e onde estão sendo gastos com recursos necessários. Por exemplo, se a análise dos dados revelar que há uma grande variação nos gastos com alimentação fora de casa, os alunos poderão propor soluções como cozinhar mais em casa ou procurar alternativas mais eficientes de alimentação. A análise dos dados pode ajudar os alunos a identificar oportunidades de economia e definir metas financeiras para a família, como economizar para uma viagem de férias ou para a compra de um bem resistente.

Uma das ferramentas que os alunos podem utilizar para analisar esses dados é o boxplot. O boxplot é um gráfico que apresenta a distribuição dos dados de forma visual importante, permitindo identificar medidas estatísticas

Os alunos irão usar a linguagem Python para analisar as medidas de dispersão dos dados coletados. Eles devem calcular a variância e o desvio padrão dos gastos com alimentação fora de casa da família, de acordo com o Código 48.

```
# Importando a biblioteca numpy
import numpy as np
# Definindo uma lista com os gastos em alimentação fora de casa da
família
gastos_alimentacao = [1000, 1200, 900, 1500, 800]
# Calculando a variância dos gastos em alimentação fora de casa da
família
variancia_alimentacao = np.var(gastos_alimentacao)
# Calculando o desvio padrão dos gastos em alimentação fora de
casa da família
desvio_padrao_alimentacao = np.std(gastos_alimentacao)
# Imprimindo os resultados
print("Variância dos gastos em alimentação fora de casa da
família: ", variancia_alimentacao)
print("Desvio padrão dos gastos em alimentação fora de casa da
família: ", desvio_padrao_alimentacao)
```

Código 48: Código do cálculo desvio padrão e da variância dos dados coletados pelos alunos dos gastos alimenticios fora de casa.

Primeiro, é importada a biblioteca *NumPy* e em seguida, é definida uma lista chamada "gastos_alimentacao", que contém os valores dos gastos em alimentação fora de casa da família. Foram utilizados valores fictícios para fins de exemplo.

Para calcular uma variância, é utilizada a função "np.var", que recebe como argumento a lista de gastos com alimentação fora de casa. A variância é uma medida estatística que indica o quão distantes os valores estão da média. Quanto maior a variância, mais dispersos são os valores.

Para calcular o desvio padrão, é utilizada a função "np.std", que também recebe como argumento a lista de gastos em alimentação fora de casa. O desvio padrão é outra medida estatística que indica o grau de dispersão dos valores em relação à média. Quanto maior o desvio padrão, maior é a variação dos valores em relação à média.

Os resultados são impressos na tela utilizando a função "*print*". O resultado da variância é exibido primeiro, seguido do resultado do desvio padrão. Essas informações podem ser usadas para analisar os gastos em alimentação fora de casa da família e buscar maneiras de reduzir esses custos, se necessário.

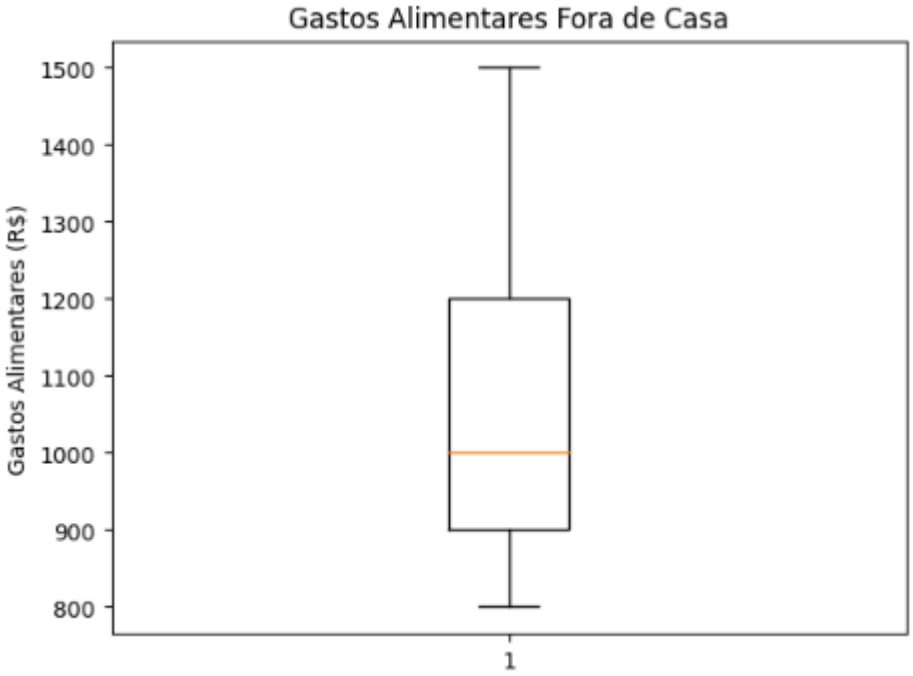
Para criar um boxplot baseado nos dados de gastos com alimentação fora de casa de uma família nos últimos 5 meses, os alunos podem usar o Código 49 como um exemplo.

```
import matplotlib.pyplot as plt

# Dados de gastos alimentares
gastos_alimentacao = [1000, 1200, 900, 1500, 800]

# Criando o boxplot
plt.boxplot(gastos_alimentacao)

# Adicionando rótulo ao eixo y
plt.ylabel('Gastos Alimentares (R$)')

# Definindo o título do gráfico
plt.title('Gastos Alimentares Fora de Casa')

# Exibindo o boxplot
plt.show()
```

Código 49: Exemplo de um boxplot com com os gatos alimentares de uma familia nos últimos 5 meses.

Ao executar o código completo, você obterá um boxplot como na Figura 10. O boxplot é composto por uma caixa que representa o intervalo interquartil (25% a 75% dos dados), uma linha que representa a mediana (50% dos dados) e linhas que se estendem para indicar a variação dos dados além desse intervalo. Isso permite visualizar a distribuição dos gastos com alimentação e identificar possíveis valores atípicos.

Figura 10: Boxplot dos gastos alimentares fora de casa



Fonte: Compilação da própria autora (2023).

Ao executar o código completo, você obterá um boxplot que representa os gastos com alimentação fora de casa da família nos últimos 5 meses.

Para avaliação do desempenho dos alunos é importante considerar a compreensão dos conceitos de gestão financeira pessoal e da importância do uso de ferramentas tecnológicas para análise de dados financeiros.

O professor pode criar um roteiro de análise dos dados financeiros e avaliar a qualidade das análises realizadas pelos alunos, bem como sua capacidade de propor soluções para os problemas identificados. Também é possível avaliar a habilidade dos alunos em utilizar a linguagem de programação Python para a análise dos dados, por meio da avaliação do código criado pelos alunos.

Outra forma de avaliação pode ser a apresentação dos resultados da análise, em que os alunos terão a oportunidade de explicar como estudar, justificar as soluções e responder a possíveis questionamentos da turma. A avaliação pode considerar a organização e clareza das informações aprendidas, bem como a habilidade dos alunos em se comunicar e expressar suas ideias de forma clara e concisa.

A atividade proposta pode ser usada para promover o pensamento crítico e o uso de problemas contextualizados usando programação em Python, bem como desenvolver habilidades práticas e teóricas em estatística e programação, ajudando os alunos e desenvolver habilidades interpessoais.

6 CONSIDERAÇOES FINAIS

A programação é uma habilidade valiosa que se torna cada vez mais importante na educação atual. Aprender a programar ajuda a desenvolver habilidades importantes como a resolução de problemas, pensamento crítico e lógico, além de incentivar a colaboração. O ensino da programação pode aumentar a diversidade na área de tecnologia, permitindo que mais pessoas de diferentes origens e perspectivas tenham a oportunidade de se envolver.

O ensino da programação Python é especialmente adequado para a resolução de problemas matemáticos e científicos, é uma linguagem de programação fácil de aprender, possui uma sintaxe clara e legível e possui bibliotecas que permitem realizar cálculos e visualizações matemáticas de maneira eficiente, além disso, pode contribuir para tornar o aprendizado da matemática mais personalizado e concreto, permitindo que os alunos visualizem conceitos abstratos de uma forma mais tangível.

Considerando o conteúdo exposto ao longo deste trabalho podemos ressaltar que a utilização da programação em Python para o ensino da estatística descritiva na educação básica representa uma nova perspectiva que pode trazer grandes benefícios para o aprendizado dos alunos, proporcionando uma maior interação dos alunos com o conteúdo, desenvolvimento de habilidades práticas e resolução de problemas reais. O uso de ferramentas computacionais pode tornar a compreensão dos conceitos matemáticos mais acessível e estimulante, além de propiciar um ensino mais dinâmico e interativo.

Os alunos podem se beneficiar da aplicação da programação em Python na estatística descritiva de diversas maneiras. Primeiramente, essa abordagem que seria a aplicação da programação em Python na matemática estatística, permite que os alunos visualizem os dados e os resultados de forma mais clara e compreensível. Além disso, o uso de algoritmos e códigos pode ajudar a desenvolver a lógica e o pensamento matemático dos alunos, preparando-os para futuras atividades e desafios que envolvem a análise de dados.

Outro aspecto positivo é a possibilidade de criar uma ponte entre o ensino teórico e a aplicação prática. Com a utilização de exemplos e exercícios reais, os alunos podem compreender como a matemática aplicada em diferentes áreas, como na análise de dados de mercado ou no desenvolvimento de pesquisas científicas.

Entretanto, é importante destacar que essa abordagem requer uma formação adequada dos professores, assim como a disponibilidade de recursos e infraestrutura adequada, visto que tem que estar em consonância com as tendências educacionais atuais,

que valorizam a interdisciplinaridade, a tecnologia e a prática como formas de aprendizagem significativas.

Além disso, é preciso garantir que a utilização da programação em Python não substitua a aprendizagem dos conceitos fundamentais da matemática estatística, mas sim complemente e aprofunde o conhecimento dos alunos através de uma metodologia de ensino

Uma proposta futura consiste na implementação da metodologia sugerida em sala de aula, seguida pela análise dos resultados obtidos. A programação é uma habilidade cada vez mais valiosa e relevante na educação atual. Ao aprender a programar, os alunos desenvolvem habilidades importantes, como resolução de problemas, pensamento crítico e lógico, além de incentivar a colaboração. O ensino da programação pode contribuir para aumentar a diversidade na área de tecnologia, permitindo que pessoas de diferentes origens e perspectivas tenham a oportunidade de se envolver.

A utilização da programação Python para o ensino de estatística descritiva na educação básica representa uma nova perspectiva com benefícios potenciais para o aprendizado dos alunos. Isso proporcionaria uma maior interação dos alunos com o conteúdo, o desenvolvimento de habilidades práticas e a resolução de problemas reais. O uso de ferramentas computacionais pode tornar a compreensão dos conceitos matemáticos mais acessível e estimulante, além de fornecer um ensino mais dinâmico e interativo

Em resumo, a utilização da programação em Python para o ensino da matemática estatística no Ensino Médio pode proporcionar grandes benefícios para os alunos, tornando o aprendizado mais acessível, dinâmico e interativo. Pode ser cada vez mais considerado uma habilidade importante para as gerações futuras e deve ser incentivado desde cedo na educação.

Em decorrência desse fato o presente trabalho abordou uma metodologia prática para professores e educadores que desejam aprimorar seus métodos de ensino na promoção de uma aprendizagem mais ativa e significativa para seus alunos. Este trabalho pode contribuir para pesquisas futuras sobre o uso da programação em Python no ensino dessas áreas. No entanto, é preciso garantir a formação adequada dos professores e que dominem alguma metodologia de ensino para validar o uso da ferramenta e a disponibilidade de recursos necessários para que essa abordagem seja aplicada com sucesso.

REFERÊNCIAS

BANIN, S. L. **Python 3 Conceitos e Aplicações: Uma Abordagem Didática**.1. ed. São Paulo, Saraiva Educação SA, 2018. 264 p.

BATISTA, R. M. **Ensino de lógica de programação na educação básica e seus impactos**. 2019. 69 p. Dissertação (Mestrado Profissional em Educação) – Programa de Pós-Graduação em Educação, Universidade Federal dos Vales do Jequitinhonha e Mucuri, Diamantina, 2019. Disponível em: http://acervo.ufvjm.edu.br/jspui/handle/1/2400. Acesso em: 14 de fev. 2023.

BERTOLINI, C.; CUNHA, G. B.; FORTES, P. R. **Lógica matemática.** 1. ed. Santa Maria, UFSM, NTE, UAB, 2017. 85 p. Disponível em: http://repositorio.bc.ufg.br/tede/handle/tede/10090. Acesso em: 14 mar. 2023.

BOBSIN, R. S. *et al*. **O Pensamento Computacional presente na Resolução de Problemas Investigativos de Matemática na Escola Básica**. *In*: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 31., 2020, Online. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2020. p. 1473-1482. Disponível em: https://doi.org/10.5753/cbie.sbie.2020.1473. Acesso em: 16 mar. 2023

BRASIL. Lei de Diretrizes e Bases da Educação Nacional, **LDB**. 9394/1996. BRASIL. 2017. Disponível em: http://www.planalto.gov.br/ccivil_03/Leis/L9394.htm. Acesso em: 11 mar 2023.

BRASIL. **Ministério da Educação. Base Nacional Comum Curricular**. Brasília, 2018.

CAMARGO, M. *et al*. **Introdução à Álgebra Linear: utilizando a linguagem de programação Python e o recurso de descrição de imagens para deficientes visuais**. 1. ed. Santa Maria, RS: UFSM, CTE, UAB, 2022. Disponível em: http://repositorio.ufsm.br/handle/1/28222. Acesso em: 13 fev. 2023

CARNEIRO, S. S. **Uso da linguagem de programação Python na formação de professor para o ensino de matemática**. 2022. 20f. TCC (Graduação em Matemática) - Universidade do Estado do Amazonas, Parintins, 2022. Disponível em: http://repositorioinstitucional.uea.edu.br//handle/riuea/4136. Acesso em: 18 fev. 2023

CARNIELLO, A.; ZANOTELLO, M. **Desenvolvimento de habilidades digitais na escola por meio da integração de jogos digitais, programação e robótica educacional virtual**. Revista de Ensino de Ciências e Matemática, *[S. l.]*, v. 11, n. 3, p. 176–198, 2020. DOI: 10.26843/rencima.v11i3.2268. Disponível em: https://revistapos.cruzeirodosul.edu.br/index.php/rencima/article/view/2268. Acesso em: 12 mar. 2023.

CASTRO, J. B. **A utilização de objetos de aprendizagem para a compreensão e construção de gráficos estatísticos**. 2012. 218f. – Dissertação (Mestrado) – Universidade Federal do Ceará, Programa de Pós-graduação em Educação Brasileira, Fortaleza (CE), 2012. Disponível em: http://www.repositorio.ufc.br/handle/riufc/7341. Acesso em 22 mar. 2023.

CORDEIRO, A. C. *et al*. Dyfocus**: Desenvolvimento do Back-End de um Aplicativo Mobile para Smartphone**. 2016. 71 f. TCC (graduação em Engenharia de Controle e Automação) - Universidade Federal de Santa Catarina. Centro Tecnológico. Florianópolis.

2016. Disponível em: https://repositorio.ufsc.br/xmlui/handle/123456789/169957. Acesso em: 3 mar. 2023.

COSTA, A. C. M. *et al*. **Python: Será que é possível numa escola pública de Ensino Médio?.** *In*: WORKSHOP DE INFORMÁTICA NA ESCOLA, 23. 2017, Recife. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2017. p. 255-264.
DOI: https://doi.org/10.5753/cbie.wie.2017.255. Acesso em: :27 jan. 2023.

DANIEL, L. O.; VARRICCHIO, S. L. **Ferramentas de Prototipação Aplicadas a Sistemas de Potência: Matlab versus Python**. Simpósio Brasileiro de Sistemas Elétricos-SBSE, Niteroi. **Anais [...].** Rio de Janeiro. v. 1, n. 1, p.7, fev. 2020.
DOI: https://doi.org/10.48011/sbse.v1i1.2333. Acesso em: 8 jan. 2023.

MORAIS, D. C. *et al*. **A linguagem de programação python para o ensino da matemática**. 2021. 48f. Trabalho de Conclusão de Curso apresentado ao Curso de Matemática – Licenciatura da Universidade do Sul de Santa Catarina, Florianópolis, 2021. Disponível em: https://repositorio.animaeducacao.com.br/handle/ANIMA/17689. Acesso em: 8 jan. 2023.

FAJARDO, R. A. S. **Lógica Matemática**. São Paulo. Editora da Universidade de São Paulo, 2017. 198 p.

ERSE, A. V. **Desenvolvimento e análise do backend do projeto CuidaIdoso.** 2021. 51 f. Monografia (Graduação em Ciência da Computação) - Instituto de Ciências Exatas e Biológicas, Universidade Federal de Ouro Preto, Ouro Preto, 2021.Disponivel em: http://monografias.ufop.br/handle/35400000/3274. Acesso em: 7 de abr. 2023.

FELISBINO, C. H. D. **A influência do desenvolvimento colaborativo de software na cultura digital.** 2013. 133 f. Dissertação (Mestrado em Tecnologias da Inteligência e Design Digital) - Programa de Estudos Pós-Graduados em Tecnologias da Inteligência e Design Digital, Pontifícia Universidade Católica de São Paulo, São Paulo, 2013. Disponível em: https://tede2.pucsp.br/handle/handle/18136. Acesso em: 9 de jan. 2023.

GALVÃO, M. C. C. **Ensino e aprendizagem da matemática na educação básica utilizando tecnologias e desenvolvendo pensamento computacional: abordagem com Scratch, Portugol, Python e Geogebra**. 2021. 164f. Dissertação (Mestrado Profissional em Matemática em Rede Nacional) - Centro de Ciências Exatas e da Terra, Universidade Federal do Rio Grande do Norte, Natal, 2021. Disponível em: https://repositorio.ufrn.br/handle/123456789/45644. Acesso em: 14 fev. 2023.

GONÇALVES, M. F**. Uso de animação digital como ferramenta auxiliar no ensino-aprendizagem de algoritmo os e lógica de programação (estudo de caso utilizando a linguagem logo).** 2017. 54 f. Trabalho de Conclusão de Curso - Faculdade de Ciência da Computação das Faculdades Integradas de Caratinga, Minas Gerais. 2017. Disponível em: http://hdl.handle.net/123456789/403. Acesso em: 3 de mar. 2023.

GOTARDO, R. **Linguagem de programação**. 1. Ed. Rio de Janeiro: **Seses**, 2015. 202 p.

GUEDES, D. B. **Linguagem de programação Python e Arduino como ferramenta para motivar estudantes iniciantes em programação**. 2018. 79 f. Trabalho de Conclusão de Curso (Tecnologia em Sistemas de Telecomunicações) - Universidade Tecnológica Federal do Paraná, Curitiba, 2018. Disponível em: http://repositorio.utfpr.edu.br/jspui/handle/1/9709. Acesso em: 14 fev. 2023.

JESUS, G. S. **Uma abordagem para auxiliar a correção de erros de programadores iniciantes**. 2018. 156 f. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Sergipe, São Cristóvão, SE, 2018. Disponível em: http://ri.ufs.br/jspui/handle/riufs/10922. Acesso em: 22 abr. de 2022.

LACERDA, Á. L. A**nálise técnica e visualização de dados do mercado de ações utilizando Python.** 2021. Trabalho de Conclusão de Curso (Ciências da Computação) - Escola de Ciências Exatas e da Computação, Pontifícia Universidade Católica de Goiás 2021 Disponível em: https://repositorio.pucgoias.edu.br/jspui/handle/123456789/1759. Acesso em: 25 mar. 2023

LEAL, C. N. N. **O uso das tecnologias da informação e comunicação (TIC): uma análise sobre os limites/possibilidades de sua implementação em uma escola pública de Santa Bárbara – PA**. 2018. Trabalho de Conclusão de Curso (Licenciatura em Pedagogia) - Campus Universitário de Castanhal, Universidade Federal do Pará, Castanhal, 2018. Disponível em: https://bdm.ufpa.br/jspui/handle/prefix/2494. Acesso em: 03 mar. 2023

LEITÃO, M. E. C. **Álgebra linear e linguagem de programação Python numa perspectiva interdisciplinar**. 2021.101 f. TCC (Trabalho de conclusão de curso do Curso de Graduação em Interdisciplinar em Ciência e Tecnologia) - Centro de Ciências Exatas e Naturais, Universidade Federal do Semi-Árido, Mossoró, 2021. Disponível em: https://repositorio.ufersa.edu.br/handle/prefix/7608. Acesso em 14 março. 2023

LIMA, P. H. F. **Data science: um glossário para profissionais da área de química.** 2022. Trabalho de Conclusão de Curso (Graduação em Química) – Universidade Federal de São Carlos, São Carlos, 2022. Disponível em: https://repositorio.ufscar.br/handle/ufscar/16843. Acesso em: 12. fev 2023.

LOPES, G. R. *et al*. **Introdução à análise exploratória de dados com Python.** Minicursos ERCAS ENUCMPI, v. 2019, p. 160-176, 2019.

LOSEKAN, G. **Jovens (e) programadores(as) : uma abordagem geracional sobre a relação entre juventude e trabalho no contexto digital**. 2022. 103 f. Dissertação (Mestrado em Sociologia) – Universidade Federal de Sergipe, São Cristóvão, 2022. Disponivel em: http://ri.ufs.br/jspui/handle/riufs/16603. Acesso em: 02 fev. 2023.

MANZANO, J. Augusto, N. G.; OLIVEIRA, J. F. **Algoritmos: lógica para desenvolvimento de programação de computadores**. 28. ed. São Paulo: Érica, 2017. E-book. Documento não paginado. Disponível em: https://app.saraivadigital.com.br/leitor/ebook:621544. Acesso em: 19 jun. 2020

MARINHO, L. H. *et al*. 2019. **Conceitos, Implementação e Dados Privados de Algoritmos de Recomendação.** In: VI Escola Regional de Sistemas de Informação do Rio de Janeiro (ERSI). Rio de Janeiro. SBC, p 32.

MATOS, R. L. O.; FILHO, O. S.; KIOURANIS, N. M. M. A "linha de abastecimento": reflexões sobre a educação das meninas na área das Ciências Exatas e da Computação. **Revista de Ensino de Ciências e Matemática**, *[S. l.]*, v. 10, n. 3, p. 18–36, 2019. DOI: 10.26843/rencima.v10i3.1999. Disponível em: https://revistapos.cruzeirodosul.edu.br/index.php/rencima/article/view/1999. Acesso em: 12 mar. 2023

MELO, A. M. **Acessibilidade e Inclusão Digital em Contexto Educacional,** Anais da 3ª Jornada de Atualização em Informática na Educação. M. A. S. N. Nunes, E. M. Rocha, Brasil, Sociedade Brasileira de Computação, p. 1-41, 2014.

MENEZES, N. N. C. **Introdução a programação com Python**. **São Paulo: Novatec**, 2010.

MIRANDA, B. D. G; MEGGIOLARO, T. N. **Tecnologias de investimentos para iniciantes**. 2022. 18 p. Trabalho de Conclusão de Curso (Faculdade de Computação e Informática) – Universidade Presbiteriana Mackenzie, São Paulo. 2022. Disponível em: https://dspace.mackenzie.br/handle/10899/31279. Acesso em: 30 jan. 2023.

MORAIS, A. D.; Basso, M. V. A.; Fagundes, L. C. **Educação Matemática & Ciência da Computação na escola: aprender a programar fomenta a aprendizagem de matemática?** Ciência & Educação (Bauru), Jun 2017, Vol. 23, Nº 2, pp. 455 – 473. Disponível em: https://doi.org/10.1590/1516-731320170020011 Acesso em: Jan 2023.

NASCIMENTO, C. A.; SANTOS, D. A. S; NETO, A. T. **Pensamento computacional e interdisciplinaridade na educação básica: um mapeamento sistemático**. In: CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 7., 29 out.-01 nov. 2018, Fortaleza (CE). Anais. Fortaleza (CE): SBC, 2018. p. 709-718.

NETO, J. V. et al. Boxplot: um recurso gráfico para a análise e interpretação de dados quantitativos. **Revista Odontológica do Brasil Central**, Goiânia, v. 26, n. 76, p. 1-6, Mar./Abr.2017. Disponível em: https://doi.org/10.36065/robrac.v26i76.1132. Acesso em: 20 mai 2023

NUNES, A. R. **Estudo numérico dos regimes de escoamento no interior de tambores rotatórios.** 2022. 68 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia (Química) – Centro de Tecnologia, Universidade Federal de Alagoas, Maceió, 2022. Disponivel em: http://www.repositorio.ufal.br/jspui/handle/123456789/10546. Acesso em: 12 fev. 2023.

PENEDO, L. S. et al. Utilização das ferramentas da qualidade nos processos de manutenção, visando o desperdício de tempo e a produtividade. **Revista Eletrônica TECCEN**, Rio de Janeiro, v. 13, n. 1, p. 16-24, Jan./Jun.2020. Disponível em: https://doi.org/10.21727/teccen.v13i1.2262. Acesso em: 20 mai 2023

PEROSA, P. **Programação em Python no Ensino Médio: uma proposta em Educação Financeira**. 2021. 121 f. Trabalho de Conclusão de Curso Curso de Matemática: Licenciatura) - Universidade Federal do Rio Grande do Sul. Instituto de Matemática e Estatística. Porto Alegre. 2021. Disponivel em: http://hdl.handle.net/10183/236525. Acesso em: 23 fev. 2023.

PESENTE, G. M. **O ensino de matemática por meio da linguagem de programação Python**. 2019. 139 p. Dissertação (Mestrado em Ensino de Ciência e Tecnologia) - Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2019. Disponível em: http://repositorio.utfpr.edu.br/jspui/handle/1/5020. Acesso em: 23 fev. 2023

RAMOS, J. S; NASCIMENTO, L. G. S. B.; BISCHOFF, R. A. **Ferramenta para monitoramento de dispositivos de rede implementada utilizando a linguagem de programação Python.** 2022. Trabalho de Conclusão de Curso (Tecnologia em Sistemas de

Telecomunicações) – Universidade Tecnológica Federal do Paraná, Curitiba, 2022. Disponível em: http://repositorio.utfpr.edu.br/jspui/handle/1/29729. Acesso em: 22 jan. 2023.

RIBEIRO, M. A. **Contribuição ao estudo do Software R como ferramenta didático-pedagógica para o desenvolvimento de Estatística Descritiva no Ensino Médio**. 2022. Dissertação (Mestrado em Mestrado Profissional em Matemática em Rede Nacional) - Instituto de Ciências Matemáticas e de Computação, University of São Paulo, São Carlos, 2022. doi:10.11606/D.55.2022.tde-17012023-152348. Acesso em: 01 dez. 2023.

ROCHA, B. O. **Uso das Tecnologias Digitais de Informação e Comunicação (TDIC) na Ação Docente no Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte – IFRN**. 2018. 380 f. Tese (Doutorado) – Curso Ciências da Educação Especialidade em Tecnologia Educativa. Universidade do Minho – Instituto Educação. Portugal. 2018.n. Disponível em: https://memoria.ifrn.edu.br/bitstream/handle/1044/1646/Usodetecnologias_rocha_2018.pdf?sequence=3. Acesso em: 10 mar. 2023.

ROCHA, E. **Segmentação do perfil de clientes inadimplentes utilizando ferramentas computacionais**. 79 p. 2022. Trabalho de Conclusão de Curso (Graduação em Estatística) - Universidade Estadual Paulista (Unesp), Presidente Prudente, 2022. Disponível em: http://hdl.handle.net/11449/217472. Acesso em: 09 mar. 2023.

ROLIM, E. L. S. S. **Uma unidade de ensino potencialmente significativa intermediada pela linguagem Python para o ensino de álgebra**. 49 p. 2021. Trabalho de Conclusão de Curso (Graduação em Licenciatura em Matemática) – Universidade Federal do Pampa, Campus Bagé, Bagé, 2021. Disponível em: http://dspace.unipampa.edu.br:8080/jspui/handle/riu/5804. Acesso em: 22 fev. 2023.

ROQUE, M. M. **Inserção de lógica de programação no ensino básico usando linguagem PYTHON e Biblioteca PYGAME**. 2017. 124 f. Monografia (Graduação em Engenharia Elétrica) - Centro de Tecnologia, Universidade Federal do Ceará, 2017. Disponível em: http://www.repositorio.ufc.br/handle/riufc/35030. Acesso em: 22 fev. 2023.

SCAICO, P. D. *et al.* **Ensino de Programação no Ensino Médio: Uma Abordagem Orientada ao Design com a linguagem Scratch**. Revista Brasileira de Informática na Educação, [S.l.], v. 21, n. 02, p. 92, ago. 2013. ISSN 2317-6121. Disponível em: http://ojs.sector3.com.br/index.php/rbie/article/view/2364. Acesso em: 12 mar. 2023. doi:http://dx.doi.org/10.5753/rbie.2013.21.02.92.

SEBESTA, R. W. **Conceitos de linguagens de programação**. Tradução: Tortello, J. E. N. T. Porto Alegre: Bookman, 2011. 11. ed. 758 p.

SOUZA, J. E. **O uso da linguagem de programação python na resolução de problemas matemáticos do ensino médio**. 2023. 124 f. Dissertação (Mestrado em Matemática em Rede PROFMAT) – Programa de Pós-Graduação em Matemática, Centro de Ciências e Tecnologia, Universidade Federal de Campina Grande, Paraíba, Brasil, 2023. Disponível em: http://dspace.sti.ufcg.edu.br:8080/jspui/handle/riufcg/29479. Acesso em: 12 abr. 2023

SOUZA, T. T. **Uma breve introdução à linguagem Phyton para professores de matemática**. 2020. 33 f. Trabalho de Conclusão de Curso (Graduação em Matemática -

Licenciatura) - Instituto UFC Virtual, Universidade Federal do Ceará, Caucaia, 2020. Disponível em: http://www.repositorio.ufc.br/handle/riufc/68976. Acesso em: 22 abr. 2023.

SILVA, M. D. **Aplicação da Ferramenta Google Colaboratory para o Ensino da Linguagem Python.** In: ESCOLA REGIONAL DE ENGENHARIA DE SOFTWARE (ERES), 4., 2020, Evento Online. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2020. p. 67-76. DOI: https://doi.org/10.5753/eres.2020.13717. Acesso em: 27 de abr. 2023.