UNIVERSIDADE ESTADUAL DO PIAUÍ - UESPI
CAMPUS POETA TORQUATO NETO
CURSO DE LICENCIATURA EM MATEMÁTICA

JOÃO GUALBERTO NETO

UMA BREVE INTRODUÇÃO A LINGUAGEM DE PROGRAMAÇÃO EM PYTHON PARA PROFESSORES DE MATEMÁTICA

TERESINA
2023

JOÃO GUALBERTO NETO

UMA BREVE INTRODUÇÃO A LINGUAGEM DE PROGRAMAÇÃO EM PYTHON PARA PROFESSORES DE MATEMÁTICA

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Matemática da Universidade Estadual do Piauí, campus Poeta Torquato neto, como requisito parcial para a obtenção do título de Licenciatura em Matemática, sob a orientação do Prof. Ms. José de Jesus Uchôa

Teresina

2023

G896b Gualberto Neto, João.
Uma breve introdução a linguagem de programação em Python para professores de matemática / João Gualberto Neto. – 2023.
32 f. : il.

Monografia (graduação) – Universidade Estadual do Piauí – UESPI, Licenciatura em Matemática, *Campus* Poeta Torquato Neto, Teresina-PI, 2023.
"Orientador Prof. Ms. José de Jesus Uchôa."

1. Matemática. 2. Linguagem de Programação Python. 3. Professor de Matemática. I. Título.

CDD: 510.07

Ficha elaborada pelo Serviço de Catalogação da Biblioteca Central da UESPINayla Kedma de Carvalho Santos (Bibliotecária) CRB-3ª/1188

JOÃO GUALBERTO NETO

INTRODUÇÃO A LINGUAGEM DE PROGRAMAÇÃO EM PYTHON PARA PROFESSORES DE MATEMÁTICA

BANCA EXAMINADORA

José de Jesus Uchôa

(Presidente da banca)

Raimundo Nonato Rodrigues

(Membro)

Avaliador(a) – Filiação acadêmica

NOTA: _____

DATA: _____/_____/_____

Teresina
2023

AGRADECIMENTOS

Em primeiro lugar, quero agradecer a Deus por me conceder força, sabedoria e perseverança ao longo dessa jornada acadêmica.

A minha família, em especial aos meus pais Antônio Luís e Antonieta Gualberto, a minha irmã Elaine Gualberto pelo apoio incondicional, o incentivo constante e o amor que vocês demonstraram foram fundamentais para minha conquista. Sem vocês, eu não teria chegado tão longe.

Também gostaria de agradecer ao meu orientador, suas orientações e disponibilidade em ajudar a concluir essa etapa

RESUMO

Este trabalho tem como objetivo apresentar uma introdução a linguagem de programação Python, direcionada especificamente para professores de matemática. Mostrar a facilidade na programação em Python que mesmo um professor sem conhecimento prévio de programação é capaz de utilizar essa linguagem como ferramenta no uso de assuntos abordados em sala. Durante essa dissertação o computador será utilizado como ferramenta para resolução de quatro problemas sendo eles: manipulação de números, resolução de equações, manipulação de matrizes e uso de algoritmos identificação de quadrados perfeitos. Esses exemplos foram escolhidos para apresentação inicial a conceitos básicos da linguagem de programação Python aplicada a matemática mostrando a utilidade de suas ferramentas no auxilio das aulas.

ABSTRACT

The aim of this work is to present an introduction to the Python programming language, specifically targeted at mathematics teachers. It aims to showcase the ease of programming in Python, demonstrating that even a teacher without prior programming knowledge can utilize this language as a tool to explore topics covered in the classroom. Throughout this dissertation, the computer will be used as a tool for solving four problems: number manipulation, equation solving, matrix manipulation, and the use of algorithms identification of perfect squares. This will demonstrate the applicability and functionality of programming in mathematics education.

LISTA DE FIGURAS

SUMARIO

1. INTRODUÇÃO

Nos últimos anos, a tecnologia tem desempenhado um papel cada vez mais importante na área da educação, o Censo Escolar de 2020 mostrou que na rede estadual, que tem a maior participação na oferta do ensino médio em todo o território nacional, 80,4% das unidades têm internet banda larga e o percentual de computadores de mesa para alunos é de 79,3% (INEP,2023)

O Python oferece uma abordagem poderosa e flexível para ensinar conceitos matemáticos complexos, permitindo que os alunos explorem e visualizem as ideias de maneira interativa. (Resnick, M., & Rosenbaum, E. (2013).

A programação em Python pode ajudar os professores de matemática a desenvolver atividades envolventes que incentivam os alunos a resolver problemas e a explorar conceitos matemáticos de forma mais prática. (Blikstein, P. (2013).

O Python oferece uma ampla gama de bibliotecas e ferramentas matemáticas que podem ser integradas ao currículo escolar, permitindo que os professores explorem tópicos avançados de matemática e promovam o engajamento dos alunos por meio de projetos práticos. (Lavy, I., & Kali, Y. (2015)

O Python oferece uma sintaxe clara e legível que permite aos professores de matemática introduzir conceitos de programação de forma gradual, facilitando o aprendizado dos alunos e sua aplicação em problemas matemáticos. (Guzdial, M. (2015).

Ao utilizar Python como uma ferramenta de programação, os professores de matemática podem estimular a criatividade e a imaginação dos alunos, permitindo que eles explorem e resolvam problemas matemáticos de maneiras inovadoras e não convencionais. (DiSessa, A. A., & Wagner, J. F. (2005)

Compreender, utilizar e criar tecnologias digitais de informação e comunicação de forma crítica, significativa, reflexiva e ética nas diversas práticas sociais (incluindo as escolares) para se comunicar, acessar e disseminar informações, produzir conhecimentos, resolver problemas e exercer protagonismo e autoria na vida pessoal e coletiva (MEC, 1996).

A programação tem se destacado como uma ferramenta poderosa para potencializar o ensino e aprendizagem em diferentes disciplinas. No campo da matemática, em particular, a linguagem de programação Python tem conquistado espaço como uma opção versátil e acessível para professores e estudantes explorarem conceitos matemáticos de maneira interativa e dinâmica.

Nessa perspectiva, diante da atual disponibilidade de maios e facilidade de acesso à tecnologia nas escolas faz-se necessário investigar e apresentar os benefícios e aplicações de Python no contexto do ensino de Matemática. Ao utilizar essa linguagem de programação, é possível desenvolver programas e projetos que proporcionam uma abordagem pratica e envolvente para explorar conceitos matemáticos complexos, desde cálculos básicos até tópicos mais avançados.

A escolha do Python como foco deste trabalho se dá em virtude de suas características favoráveis para a educação matemática. Python é uma linguagem de programação de alto nível, de fácil compreensão e sintaxe simples, o que torna seu aprendizado acessível mesmo para aqueles que não possuem experiencia prévia em programação. além disso, Python oferece uma ampla gama de bibliotecas e recursos específicos para cálculos numéricos, álgebra simbólica, visualização de dados e resolução de problemas matemáticos.

Ao longo deste trabalho, serão apresentados exemplos práticos de como o Python pode ser utilizado para explorar diferentes áreas da Matemática. Serão abordados tópicos como a manipulação de números, resolução de equações, manipulação de matrizes e até mesmo o uso de algoritmos de aprendizado de máquina em problemas matemáticos.

Espera-se que este TCC forneça um panorama abrangente das possibilidades oferecidas pelo Python como ferramenta auxiliar no ensino de matemática. Ao explorar os benefícios e recursos dessa linguagem de programação, professores de matemática poderão enriquecer suas aulas, tornando-as mais interativas, práticas e relevantes para os alunos, estimulando o interesse pela disciplina e facilitando a compreensão dos conceitos matemáticos.

No próximo capítulo aprenderemos uma noção geral sobre algoritmo e uma noção específica sobre algoritmo matemático. Essa noção básica faz-se necessária pois a base de qualquer linguem de programação é através do uso de algoritmos. O nosso foco será em resolver quatro problemas matemáticos explicando algumas ideias e características da linguagem que existem por trás de cada um desses

problemas, sendo eles: manipulação de números, resolução de equações, calcular matriz inversa e determinar quadrados perfeitos

2. ALGORITIMO

2.1. DEFINIÇÃO

Os algoritmos estão presentes em diversas áreas do conhecimento, como ciência da computação, matemática engenharia e até mesmo nas atividades diárias. Eles desempenham um papel fundamental na resolução de problemas, fornecendo uma estrutura lógica e organizada para a execução de tarefas.

Um bom algoritmo deve ser preciso, eficiente, claro e capaz de resolver o problema proposto. Ele deve ser compreensível o suficiente para ser implementado por um ser humano ou por uma máquina. Além disso, os algoritmos podem ser representados de diferentes formas, como fluxogramas, pseudocódigos ou em uma linguagem de programação especifica.

No campo da ciência da computação, os algoritmos são a base para o desenvolvimento de programas e sistemas. Eles são usados para realizar uma ampla variedade de tarefas, desde ordenar uma lista de números até realizar complexas simulações ou algoritmos de aprendizado de máquina. Os algoritmos são a essência da programação, pois fornece as instruções necessárias para que um computador execute uma determinada tarefa.

Em resumo, os algoritmos são sequencias de passos que descrevem como resolver um problema de forma sistemática e organizada. Eles são fundamentais em várias áreas do conhecimento, incluindo a matemática e ciências da computação, e desempenham um papel crucial na resolução de problemas e na implementação de soluções eficientes.

2.2. ALGORITMO MATEMÁTICO

Em Matemática, um algoritmo é um conjunto de instruções logicas e precisas que descreve a solução de um problema ou a realização de um cálculo matemático

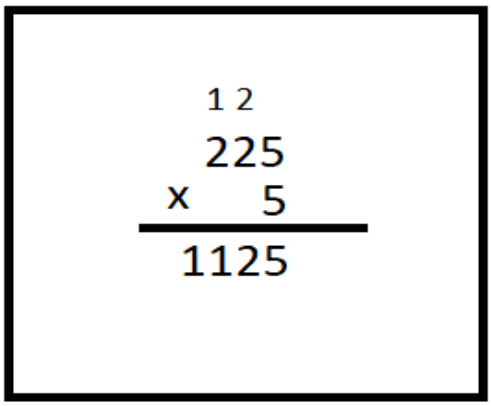
especifico. Os algoritmos matemáticos podem envolver diferentes áreas, como álgebra, geometria, calculo, estatística, entre outros.

Por exemplo, na álgebra, um algoritmo pode descrever a resolução de uma equação, seguindo uma sequência de passos para encontrar o valor desconhecido. Esse processo pode envolver operações como simplificação, isolamento de variáveis, aplicação de propriedades algébricas e substituições.

Fica fácil perceber que o professor no seu dia a dia em sala usa vários algoritmos matemáticos na imagem a seguir, vamos ilustrar o uso do algoritmo da multiplicação que o professor escreve no quadro quando mostra para os alunos os passos desse algoritmo:

Exemplo: Determine o valor da multiplicação de 5 por 225

Figura 1 algoritmo multiplicação



Fonte o autor

Neste exemplo temos os números 5 e 225 os valores de entrada os quais serão multiplicados. O algoritmo começa a ser aplicado partir do momento que organizamos os valores de entrada em linhas com um dos valores na parti superior, o segundo valor na parte central com o símbolo da multiplicação representado por x, uma barra horizontal e embaixo da mesma é representado o resultado da operação. Então começamos a fazer os cálculos. Primeiramente, multiplicamos 5x5=25.

Como o valo encontrado possui 2 dígitos usaremos a regra conhecida como “sobe um” que colocamos o 5 na parte inferior embaixo da barra horizontal e colocamos 2 em cima do próximo número a ser multiplicado na primeira entrada, agora

vamos multiplicar a próxima coluna, ou seja, 5x2=10 e somar o número na parte superior da coluna, 10+2=12 novamente encontramos dois dígitos e repetimos o processo do “sobe um” deixamos o 2 no resultado e colocamos 1 em cima da coluna seguinte, multiplicamos essa coluna , ou seja 5x2=10 e somar o número na parte superior da coluna, 10+1=11, como não temos mais colunas o resultado 11 será colocado na linha inferior em baixo da linha horizontal. Portanto, temos a solução do exemplo, ou seja, 5x225=1125. É fácil vermos que acabamos de representar um algoritmo matemático como foi descrito anteriormente

Com esse exemplo podemos observar que o professor usa no seu dia a dia em sala de aula, mesmo sem perceber, diversos algoritmos matemáticos em várias áreas com resolução de equações do segundo grau, cálculo de áreas de figuras planas, cálculo de matrizes, etc. De maneira geral, os professores já possuem o conhecimento básico de algoritmos e suas utilizações faltando somente o conhecimento de programação para que seja feita uma aplicação dessa base já aprendida.

3. PYTHON

3.1. CONCEITO

Python é uma linguagem de programação de alto nível, interpretada e de propósito geral. Ela foi desenvolvida por Guido Van Rossum nos anos 1990 e se tornou uma das linguagens mais populares do mundo devido à sua simplicidade, legibilidade e grande comunidade de desenvolvedores (WIKIPEDIA, 2023).

Uma das vantagens do Python é sua facilidade de uso, que permite que até mesmo iniciantes em programação possa escrever códigos rapidamente. Além disso, possui uma sintaxe limpa e clara, o que torna fácil entender e depurar os programas.

Python permite realizar operações matemáticas comuns, como adição, subtração, multiplicação e divisão. Pode se utilizar os operadores aritméticos como (+, -, *, /) para realizar essas operações em números inteiros e de ponto flutuantes. Que serão representados respectivamente por “*int*” e “*float*”

Em Python, assim como em outras linguagens de programação, os pontos flutuantes são modos equivalentes de dados utilizados para representar números reais não inteiros, ou seja, números que podem ter casa decimais.

Os números de ponto flutuante em Python são representados pelo tipo de dado "*float*". Eles são usados para armazenar números reais, incluindo números inteiros e fracionários. Por exemplo, a representação de pi como *float* (3.14) e número de Euler representado por *float* (2.71828) são números de ponto flutuante.

É importante ter em mente que devido à natureza do armazenamento em ponto flutuante, algumas operações matemáticas podem ter resultados imprecisos devido a arredondamentos e limitação de precisão. Isso ocorre porque nem todos os números reais podem ser exatamente representados em ponto flutuante

Na programação em Python, a diferença de uso do ponto e da virgula em números é importante e está relacionada à representação decimal e ao formato utilizado para separar a parte inteira da parte fracionaria de um número.

Em Python, o ponto (.) é utilizado como separador decimal padrão. Isso significa que ao escrever um número com parte fracionaria, como 3.14, o ponto é utilizado para indicar a separação entre a parte inteira (3) e a parte decimal (0.14). É importante observar que em Python (assim como em muitas outras linguagens de programação), o ponto é utilizado como separador decimal, independente do padrão usado em alguns países o qual a virgula desempenha esse papel.

Por outro lado, a virgula (,) é utilizada em Python para outros propósitos, como separar itens em listas ou *tuplas* (lista imutável). Por exemplo, em uma lista de números [1, 2, 3], a virgula é utilizada para separar cada elemento da lista. Portanto, se usarmos a virgula como separador decimal em um número, isso resultara em um erro de sintaxe

3.2. BIBLIOTECAS MATEMATICAS

Em Python, bibliotecas são conjuntos de módulos e pacotes contendo funções, classes e variáveis que facilitam o desenvolvimento de software. Elas fornecem funcionalidades especificas para tarefas comuns, permitindo que os programadores reutilizem o código já implementado em vez de escrever tudo do zero

Existem várias bibliotecas de matemática disponíveis em Python que facilitam o trabalho com cálculos numéricos, simbólicos, estatísticos e visualizações. A seguir vou apresentar algumas das bibliotecas mais populares:

- **NumPy**: NumPy é uma biblioteca fundamental para computação numérica em Python. Ela fornece suporte para arrays multidimensionais eficientes e uma ampla gama de funções matemáticas para realizar operações numéricas. Com

o NumPy, é possível executar cálculos matemáticos complexos de forma eficiente e otimizada.

- **SciPy**: SciPy é uma biblioteca que se baseia no NumPy e oferece recursos adicionais para computação científica. Ela inclui módulos para otimização, álgebra linear, integração numérica, interpolação, processamento de sinais, estatísticas e muito mais. O SciPy é amplamente utilizado em áreas como ciência de dados, física, engenharia e outras disciplinas científicas.
- **SymPy**: SymPy é uma biblioteca de matemática simbólica em Python. Ela permite realizar cálculos simbólicos, manipular expressões algébricas, resolver equações, diferenciar e integrar funções simbolicamente. Com o SymPy, é possível executar operações matemáticas exatas em vez de aproximadas, o que é especialmente útil em áreas como matemática pura, física teórica e engenharia de controle.
- **Matplotlib**: Matplotlib é uma biblioteca de visualização de dados em Python. Ela oferece recursos para criar gráficos 2D e 3D de alta qualidade. Com o Matplotlib, você pode criar gráficos de linhas, gráficos de dispersão, histogramas, gráficos de barras, gráficos de contorno e muito mais. Essa biblioteca é amplamente utilizada para visualizar dados e comunicar resultados de forma clara e eficaz.
- **pandas**: Embora não seja exclusivamente uma biblioteca de matemática, o pandas é uma ferramenta essencial para análise e manipulação de dados em Python. Ele fornece estruturas de dados poderosas, como DataFrames, que facilitam o trabalho com dados tabulares. O pandas inclui funções para filtrar, ordenar, agrupar e realizar cálculos estatísticos em dados, tornando-o uma opção popular para análise de dados e pesquisa científica.

3.3. INTERFACE INICIAL

A programação em Python pode ser realizada com a instalação de programas específicos para utilização no computador ou pode ser feita direto no navegador da web o qual usaremos por sua facilidade de acesso e manuseio afim de facilitar o entendimento e aprendizado da linguagem de programação. O site referido é feito um cadastro prévio através com um e-mail o que gera um ambiente virtual que proporciona para o professor uma maior facilidade de acesso aos arquivos já preparados para aulas expositivas, proporcionando acesso ao material aos alunos e ao professor. Ao clicar no link qualquer usuário terá acesso a interface de programação de forma gratuita. Para iniciarmos precisamos acessar https://replit.com/new/python3

Acessando o link abrimos uma página inicial com a interface principal de programação que é dividida em 2 partes distintas.

figura 2 área de edição

Fonte o autor

A parte 1 é onde digitamos os algoritmos na linguagem Python, chamada de área de edição, a parte 2 é onde fica o botão “RUN” que iniciara o processamento do algoritmo, e área onde mostra o funcionamento da interface de interação e resultado do complemento dos dados mostrando de forma direta o resultado obtido através do algoritmo digitado na parte 1.

Ao primeiro acesso ao site na área de digitação encontramos um exemplo inicial fornecido pelo mesmo o qual representa uma soma de dois números naturais.

figura 3 algoritmo 1

Fonte o autor

Ao apertamos no botão RUN iniciamos a interação com a interface na parte 2 e digitando o primeiro número 25 e apertando a tecla ENTER no teclado, indicamos para o código que esse valor representa a primeira entrada na variável “a” do algoritmo em seguida colocamos o valor 15 representando a segunda variável “b” e apertamos a tecla ENTER e obtemos o resultado 40 que representa a solução

Podemos observar que de maneira pratica o exemplo inicial fornecido pelo site cria um programa interativo que ao colocar 2 números inteiros quais quer o algoritmo vai realizar essa operação e encontrar o valor dessa soma. Temos então um programa simples em Python, com apenas oito linhas de código que exemplifica o processo de somar dois números inteiros de forma interativa.

4. EXEMPLOS

4.1. MANIPULAÇÃO DE NUMEROS

A manipulação de números é um aspecto fundamental da matemática e desempenha um papel essencial em várias áreas, como álgebra, calculo, estatística e muitas outras disciplinas matemáticas. A capacidade de manipular números de maneira eficaz é crucial para resolver problemas matemáticos complexos e aplicar conceitos matemáticos em situações do mundo real.

Utilizaremos as 4 operações básicas adição, subtração, multiplicação e divisão, que serão exemplificadas pelas mesmas variáveis ‘a=10’ e ‘b=15’ nas linhas 1 e 2 que são os valores de entrada no algoritmo, nas linhas 4, 5, 6, e 7 são as operações chamados de argumentos que são:

soma = a + b

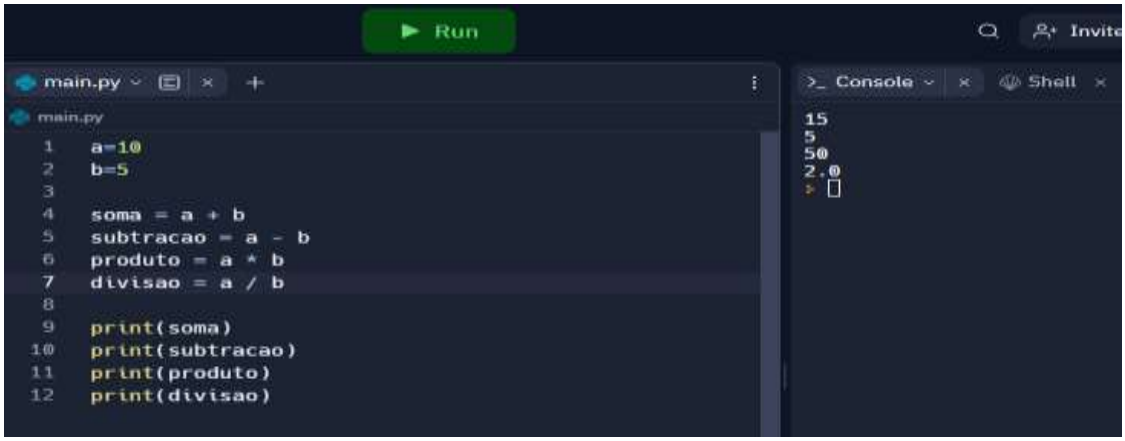
subtracao = a - b

multiplicacao = a * b

divisao = a / b

A soma é representada pelo sinal de mais (+) e a subtração por menos (-) já a multiplicação e a subtração são representadas respectivamente por asterisco (*) e barra (/). Em Python a utilização de sinais de pontual (circunflexo, cedilha, tio, etc..), não são utilizados nas linhas de digitação do algoritmo, os mesmos desempenham outras funções na programação. As linhas 9, 10, 11 e 12 representam a função de saída que é o print () essa sintaxe tem como função principal representa o resultado do argumento na tela. Observe a seguir como ficara o código.

figura 4 algoritmo 2



Fonte o autor

Ao clicar em RUN na parte 2 da interface processa os dados e mostra o resultado das 4 operações respectivamente.

O programa foi executado com perfeição é preciso agora aperfeiçoa-lo para que os valores de entrada sejam dinâmicos variando de acordo com a interação do usuário, as entradas das variáveis ‘a’ e ‘b’ são valores fixos no exemplo acima para que sejam variáveis é necessário utilizar o comando “input”, que permite uma

interação entre o usuário e a interface, o mesmo código do exemplo anterior apenas acrescentando o comando “input” as variáveis e “float” tornando as variáveis pontos flutuantes

figura 5 algoritmo 3

Fonte o autor

Com adição do comando float, o algoritmo é capaz de realizar qualquer uma das 4 operações básica com dois números inteiros de forma interativa com o usuário digitando os valores das duas variáveis.

4.2. RESOLUÇÃO DE EQUAÇÕES

A resolução de equações matemáticas é uma habilidade fundamental no estudo da matemática e tem aplicações em uma ampla variedade de campos, desde ciência e engenharia ate finanças e estatística. A capacidade de resolver equações permite-nos encontrar valores desconhecidos e compreender as relações entre diferentes variáveis.

Uma equação matemática é uma afirmação que estabelece que duas expressões são iguais. Essas expressões podem conter variáveis desconhecidas, constantes e operações matemáticas. O objetivo ao resolver uma equação é encontrar o valor ou valores que tornam a equação verdadeira, ou seja, satisfazem a igualdade.

Existem diferentes tipos de equações, como linear, quadrática, polinomiais, exponenciais, logarítmicas, etc... A abordagem para resolver cada tipo de equação pode variar, mas existem algumas estratégias gerais que podem ser aplicadas com o auxílio do Python e o uso de algoritmo para resolver essas equações.

Para isso aplicaremos apenas um algoritmo e com poucas modificações, que possibilite a resolução de equação linear, equação quadrática e sistema de equações.

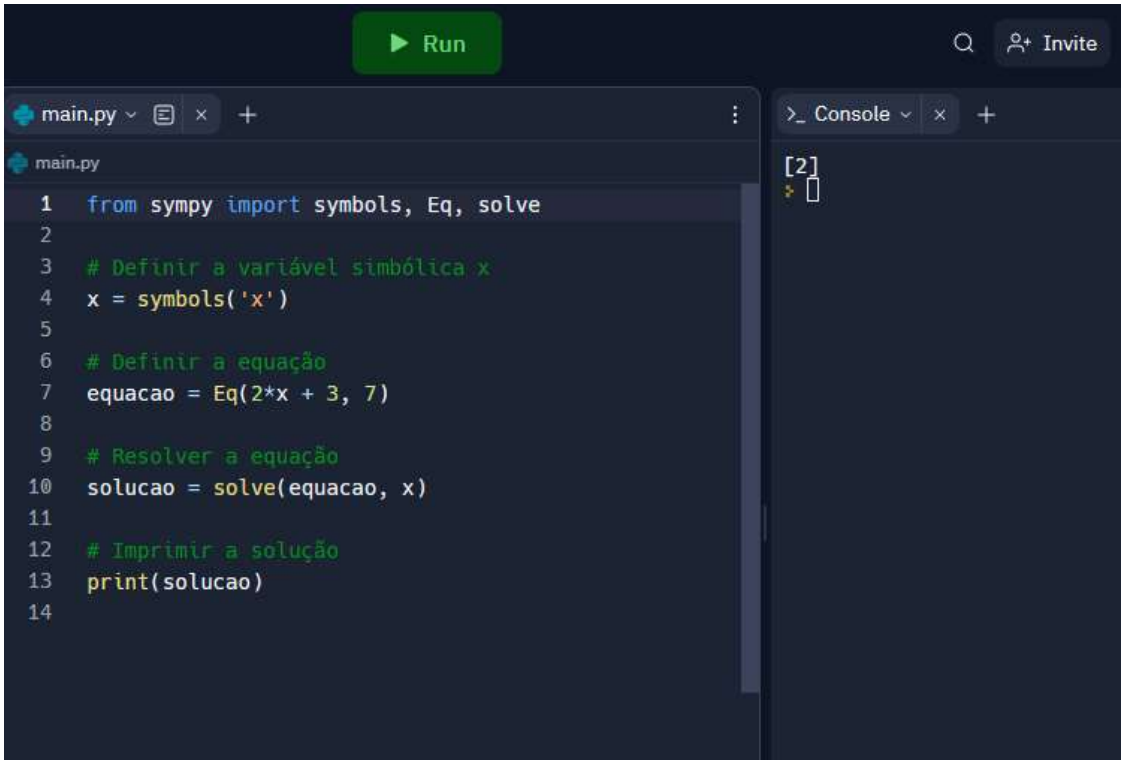
4.2.1. EQUAÇÃO LINEAR

A resolução de equações matemáticas é uma habilidade fundamental no estudo da matemática e tem aplicações em uma ampla variedade de campos, desde ciência e engenharia até finanças e estatística. A capacidade de resolver equações permite-nos encontrar valores desconhecidos e compreender as relações entre diferentes variáveis.

Uma equação matemática é uma afirmação que estabelece que duas expressões são iguais. Essas expressões podem conter variáveis desconhecidas, constantes e operações matemáticas. O objetivo ao resolver uma equação é encontrar o valor ou valores que tornam a equação verdadeira, ou seja, satisfazem a igualdade.

Como exemplo para criação do algoritmo de resolução de equações lineares e sistemas de equações será utilizado a equação: $2x+3=7$ vamos descrever um algoritmo que determine uma solução para essa equação dada.

figura 6 algoritmo 4



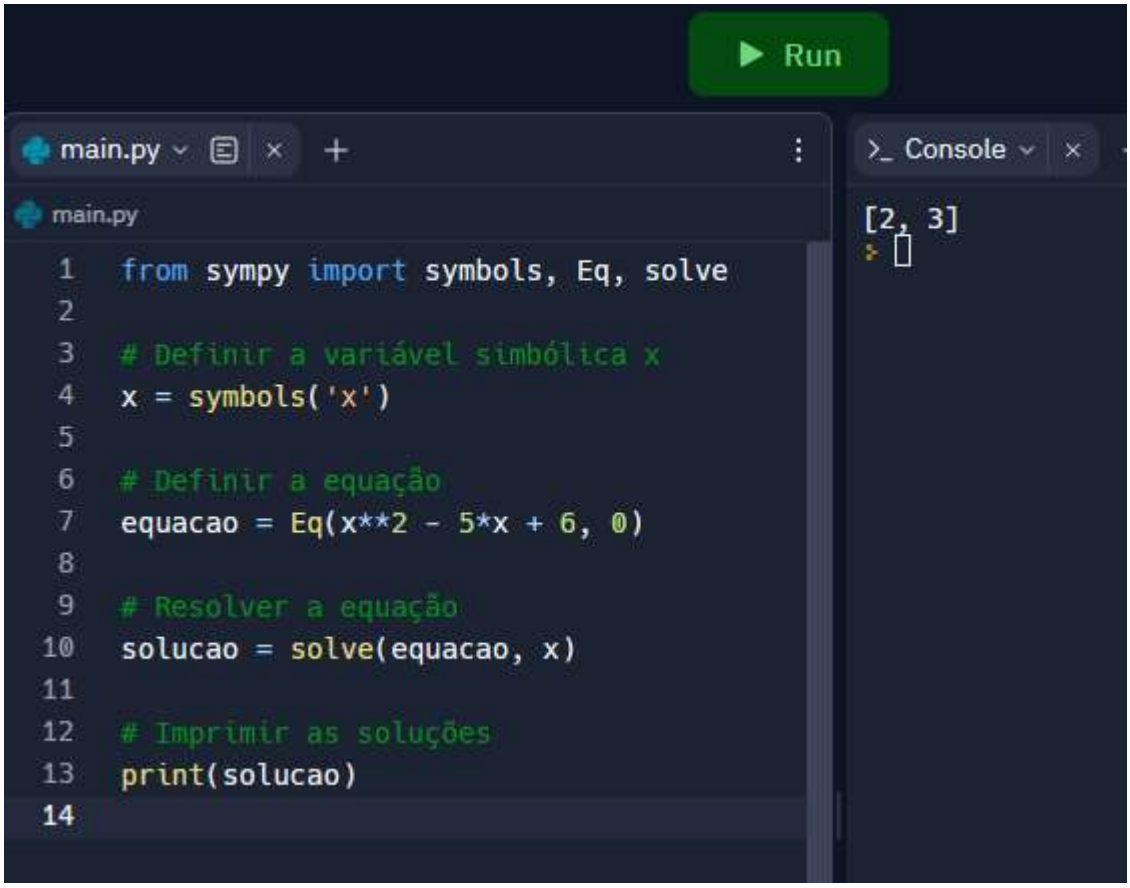
Fonte o autor

Na linha 1 determinamos qual biblioteca que está sendo utilizada, para equações lineares é utilizada sympy após colocamos com Import os comandos matemáticos symbols, Eq, solve. Na linha 4 determinamos a variável a ser utilizada e na linha 7 está descrita a equação dada. Observe que em comparação com a equação o sinal de igualdade é representa por uma virgula (,) e por fim definimos ‘solve’ o modo de resolução.

4.2.2. EQUAÇÃO QUADRATICA

Agora utilizaremos a seguinte equação quadrática: x^2 - 5x + 6 = 0. Assim como no exemplo anterior temos uma equação linear, então usaremos o mesmo código apenas modificando as linhas que representam a equação quadrática no algoritmo e encontraremos as raízes dessa equação.

figura 7 algoritmo 5



Fonte o autor

Observe que na representação matemática usual escrevemos a equação quadrática como a do exemplo sendo x^2-5x+6=0 fazendo uma comparação com a linha 7 do algoritmo podemos notar que os operadores matemáticos tem algumas representações próprias da linguagem de programação. Primeira representação é a de potenciação que está representada pela sintaxe dois asteriscos (**). Próxima relação é a multiplicação de sintaxe asterisco (*) e a igualdade de sintaxe representada por (,) outra forma de sintaxe para igualdade é (==).

4.2.3. SISTEMAS DE EQUAÇÕES

Um sistema de equações matemáticas é um conjunto de duas ou mais equações que estão relacionadas entre si. Essas equações envolvem variáveis desconhecidas, também chamadas de incógnitas. A solução de um sistema consiste

em encontrar os valores das incógnitas que satisfazem todas as equações simultaneamente.

Os sistemas de equações são amplamente utilizados em diversas áreas da matemática e em problemas práticos do dia a dia. Eles fornecem uma maneira de descrever relações entre diferentes quantidades ou variáveis, permitindo que sejam resolvidos para encontrar soluções específicas.

Usaremos o mesmo algoritmo já utilizados nas 2 equações anteriores, apenas com algumas modificações para satisfazer as diferenças das equações anteriores.

O exemplo será: determine a solução de um sistema formado pelas equações.

$$\begin{cases} 2x + y = 5 \\ x - y = 1 \end{cases}$$

A primeira modificação será as variáveis, anteriormente somente com uma denominada de x. No sistema que segue usaremos 2 variáveis x e y para isso precisamos modificar a linha 4 que está escrito: x=symbols (‘x’) representando apenas a variável x para representar duas variáveis x e y substituiremos por x, y=symbols (‘x y’), observe que a virgula funciona como um separador de termos e não como determinação de casas decimais.

A segunda modificação será as equações que formam o sistema possuindo duas precisamos classificar essas equações para que o algoritmo reconheça as mesmas nas linhas 7 e 8 chamaremos de equacao1 (2x+y=5) e de equacao2 (x-y=1)

As últimas modificações será o comando ‘solucao’ na linha 11 acrescentando as duas equações classificadas e as variáveis ao comando ‘solve’ após as modificações obtemos um programa capaz de resolver sistemas de equações

figura 8 algoritmo 6

Fonte o autor

Com apenas 1 algoritmo e pequenas modificações nas linhas de comando possibilitou a criação de um programa capaz de resolver os 3 exemplos anteriores, que o professor poderá utilizar em sala para aulas expositivas sobre equações lineares.

4.3. CALCULAR MATRIZ INVERSA

Vamos expressar um exemplo de cálculo de matriz 2x2 que será resolvida em passos representando um algoritmo matemático para que possibilite a transformação desse algoritmo na linguagem de programação Python

Vamos considerar a matriz A:

$$\begin{pmatrix} 2 & 3 \\ 4 & 5 \end{pmatrix}$$

Passo 1: Calcule o determinante da matriz A.

$$\det(A) = (2 \times 5) - (3 \times 4) = 10 - 12 = -2$$

Passo 2: Verifique se o determinante é diferente de zero (det(A) ≠ 0).

Neste caso, -2 é diferente de zero, portanto, a matriz A possui inversa.

Passo 3: Calcule a matriz adjunta de A.

$$\text{adj(A)} = \begin{pmatrix} 5 & -3 \\ -4 & 2 \end{pmatrix}$$

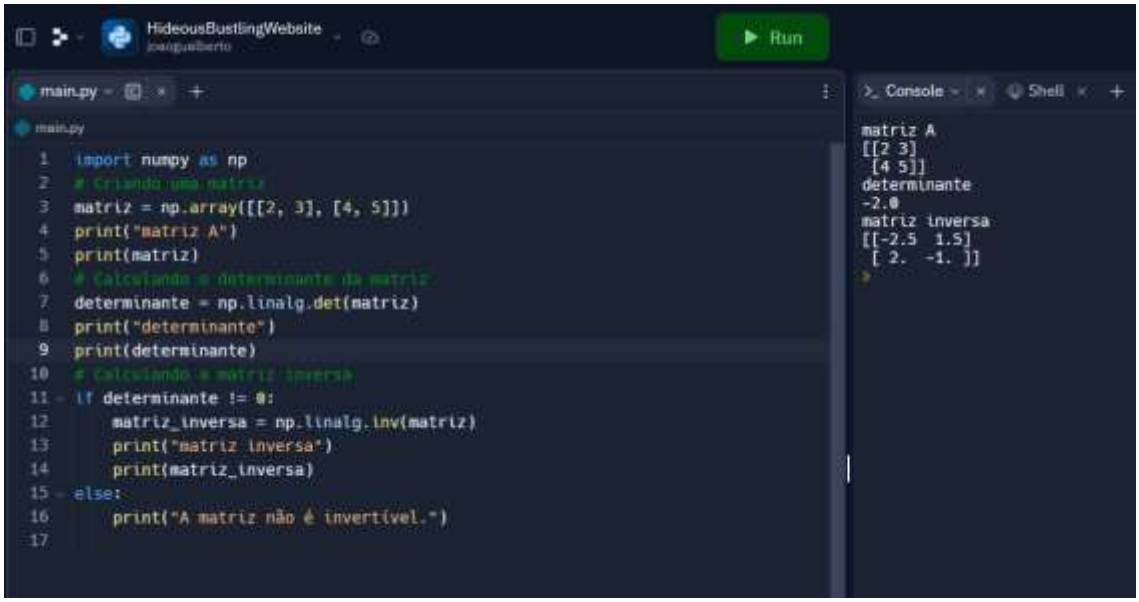
Passo 4: Calcule a matriz inversa de A dividindo cada elemento da matriz adjunta de A pelo determinante det(A).

$$\text{Inversa de A} = \begin{pmatrix} \frac{5}{-2} & \frac{-3}{-2} \\ \frac{-4}{-2} & \frac{2}{-2} \end{pmatrix} = \begin{pmatrix} -2{,}5 & 1{,}5 \\ 2 & -1 \end{pmatrix}$$

Os 4 passos representam o processo para calcular o inverso de uma matriz 2x2 como a do exemplo. Até uma matriz sendo de ordem dois mesmo sem os cálculos ficou evidenciado a dificuldade desse processo, que só aumenta com o acréscimo de colunas e linhas as matrizes. Como já temos os passos do processo podemos escrever um algoritmo em Python que resolva essas operações com matrizes.

O “*import*” é utilizado como comando de identificação de qual biblioteca estamos utilizando, além disso temos o “np” refere-se a "numpy", que é uma biblioteca popular para computação numérica em Python. Ao importar **numpy** da seguinte forma: ***import numpy as np***, estamos atribuindo um apelido "np" para a biblioteca *numpy*. Isso é comumente feito para tornar mais conveniente o uso de funções e métodos dessa biblioteca ao longo do código, sem ter que digitar o nome completo "*numpy*" toda vez que precisamos usá-la.

figura 9 algoritmo 7



Fonte o autor

A imagem acima representa o algoritmo em Python do exemplo. Observe que o algoritmo está dividido em 3 partes. A primeira é a entrada da matriz a ser calculada na linha 3, linha 4 e linha 5, foi utilizado o comando *array* na primeira linha e terminando om o comando *print.* Em Python *array* se refere a uma estrutura de dados do tipo *"array"* ou *"matriz".* Um *array* é uma coleção ordenada de elementos, em que cada elemento pode ser acessado por meio de um índice numérico no exemplo a entrada das matrizes esta separada por linhas cada uma representada pelos valores entre os colchetes [], a primeira parte finalizando pelo comando *print* para que seja representado a matriz no console de saída

A segunda parte é o cálculo do determinante representado pelas linhas 7,8 e 9 do algoritmo. Para que a matriz seja invertível, o determinante da matriz precisa ser diferente de zero. Caso contrário, a matriz não tem inversa. Para determinar o valor do mesmo usamos o comando *linalg.det* é uma função da biblioteca *NumPy* que permite calcular o determinante de matrizes quadradas em Python. Apenas com esse comando é suficiente para sabermos se a matriz e invertível.

A terceira parte representada entre as linhas 11 a 16. Como para que uma matriz seja invertível temos um condicional, que se o determinante for diferente de zero a matriz é invertível caso contrário não é invertível usaremos dois comandos *if* é usado para executar um bloco de código se uma condição for avaliada como verdadeira. O comando *else* pode ser usado em conjunto com o *if* para executar um

bloco de código alternativo caso a condição seja avaliada como falsa. No exemplo casa determinante seja diferente de zero será executado o código após o comando *if* caso seja igual a zero será executado o comando *else* e aparecera a mensagem A matriz não é invertível.

Com esse algoritmo para que seja feito o cálculo de matrizes acima da ordem 2 basta acrescentar os valores na primeira parte que o algoritmo continuará fornecendo resultados.

4.4. QUADRADO PERFEITO

Um quadrado perfeito de um número é o resultado da multiplicação desse número por ele mesmo. Em outras palavras, é um número inteiro que pode ser expresso como o produto de dois iguais. Por exemplo, o quadrado perfeito de 4 é 16, pois 4 multiplicado por 4 é igual a 16.

Existem várias maneiras de determinar se um número é um quadrado perfeito. Alguns métodos incluem a verificação se a raiz quadrada do número é um número inteiro ou se o número pode ser dividido exatamente por todos os seus fatores primos em pares.

Por exemplo, para determinar se 49 é um quadrado perfeito, podemos calcular sua raiz quadrada, que é 7. Como a raiz quadrada é um número inteiro, podemos concluir que 49 é um quadrado perfeito.

Da mesma forma, se tivermos o número 27, sua raiz quadrada é aproximadamente 5,196. Como a raiz quadrada não é um número inteiro, podemos concluir que 27 não é um quadrado perfeito.

Para o exemplo a seguir precisamos de um algoritmo que de forma interativa ao colocar um numero o algoritmo possibilite o retorno se esse número é um quadrado perfeito, para isso faremos como na imagem a seguir atreves de condicionadores.

figura 10 algoritmo 8

Fonte o autor

No algoritmo usamos a função *eh_quadrado_perfeito*. O método utilizado está nas linhas 2 e 3 que ao colocar um número o algoritmo irá calcular a raiz desse número e multiplica a raiz encontrada por ela mesmo e determinar se esse número é um quadrado perfeito.

Nas linhas 6 a linha 10 encontramos um condicional para a determinação se esse número do valor de entrada é ou não um quadrado perfeito para isso foi utilizado os condicionadores ***if*** e ***else***, observe que ao digitar um valor de entrada como o 25 na imagem acima, se a função for verdadeira o condicionador ***if*** será utilizado e retornara um resultado que o número é um quadrado perfeito caso não seja o condicionador ***else*** será utilizado retornando um resultado que o número não é um quadrado perfeito.

5. CONCLUSÃO

A resolução dos quatro problemas propostos estabeleceu uma base sólida para a compreensão dos conceitos básicos da linguagem e ofereceu uma introdução ao mundo da linguagem Python para os professores de matemática interessados em aprofundar seus conhecimentos. Ficou evidente que Python é uma linguagem extremamente rica, com funções inovadoras e a capacidade de criar novas bibliotecas, o que requer um esforço contínuo por parte dos usuários que desejam explorar melhor a linguagem.

Acreditamos que, embora pareça algo distante, o Python poderia ser amplamente utilizado nas escolas, especialmente pelos professores de matemática, devido à sua proximidade com conceitos, teorias e ideias matemáticas. Este pequeno trabalho pode servir como um ponto de partida para que os professores aprimorem seu conhecimento da linguagem, mergulhem nesse novo universo de conhecimento e compartilhem com seus alunos as amplas possibilidades que o Python oferece.

Ao incorporar Python em suas aulas, os professores de matemática podem enriquecer o ensino, trazendo exemplos práticos, resolução de problemas e visualizações gráficas interativas que tornam os conceitos matemáticos mais tangíveis e interessantes. Além disso, os alunos se beneficiarão ao aprender uma habilidade valiosa e altamente requisitada no mundo atual, onde a programação e a análise de dados desempenham um papel cada vez mais importante.

Encorajamos os professores de matemática a se dedicarem a explorar a linguagem Python, aproveitando seu potencial para promover um ensino mais dinâmico e envolvente. Com esforço e dedicação contínuos, eles poderão adquirir um conhecimento mais profundo da linguagem e, assim, trazer aos seus alunos uma educação mais abrangente e atualizada.

6. Referencias

ALGORITMO. **Wikipédia**. [S.l], 2020. Disponível em: https://pt.wikipedia.org/wiki/Algoritmo. Acesso em: 06 jun. 2023

EDUCAÇÃO é a base. MEC [S.n.t]. Disponível em: http://basenacionalcomum.mec.gov.br/. Acesso em: 04 jun. 2023

Blikstein, P. (2013). Gears of our childhood: constructionist toolkits, robotics, and the rise of computational thinking. In FabLabs: Of Machines, Makers and Inventors, 49-78.)

Lavy, I., & Kali, Y. (2015). Bridging science education and computational thinking: exploring computational thinking in a high school biology unit. Journal of Science Education and Technology, 24(6), 861-877.)

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. ABNT NBR 10520: apresentação de citações em documentos. Rio de Janeiro, 2002a.

_____. ABNT NBR 12225: títulos de lombada. Rio de Janeiro, 2004a. _____. ABNT NBR 14724: trabalhos acadêmicos: apresentação. Rio de Janeiro, 2011a.

_____. ABNT NBR 15287: projeto de pesquisa: apresentação. Rio de Janeiro, 2011b.

_____. ABNT NBR 6023: referências: elaboração. Rio de Janeiro, 2002b. _____. ABNT NBR 6024: numeração progressiva das seções de um documento. Rio de Janeiro, 2012a.

_____. ABNT NBR 6027: sumário. Rio de Janeiro, 2012b.

_____. ABNT NBR 6028: resumos. Rio de Janeiro, 2003.

_____. ABNT NBR 6034: índice. Rio de Janeiro, 2004b.

Guzdial, M. (2015). Learner-centered design of computing education: Research on computing for everyone. Synthesis Lectures on Human-Centered Informatics, 8(1), 1-149.)

DiSessa, A. A., & Wagner, J. F. (2005). What coordination has to say about learning. Journal of the Learning Sciences, 14(3), 365-397.)

MARCONDES, Guilherme A. Barucke. Matemática com Python: um guia prático. São Paulo: Novatec, 2018.

MATPLOTLIB: Visualization with Python. MATPLOTLIB, 2012. Disponível em: https://matplotlib.org/. Acesso em: 06 jun. 2023.

MINISTÉRIO DA EDUCAÇÃO, Base nacional comum curricular: educação é a base. Brasília, 1996. Disponível em: http://basenacionalcomum.mec.gov.br/images/BNCC_EI_EF_110518_versaofinal_site.pdf. Acesso em: 25 mai. 2023.

OLIVEIRA, Raul Rodrigues de. FATORIAL. [S.l], 2020. Disponível em: https://brasilescola.uol.com.br/matematica/fatorial.htm. Acesso em: 8 jun. 2023

PYTHON library for symbolic mathematics. SYMPY, 2020. Disponível em: https://www.sympy.org/en/index.html. Acesso em: 01 jun. 2023.

Resnick, M., & Rosenbaum, E. (2013). Designing for tinkerability. Design, make, play: Growing the next generation of STEM innovators, 163-181.

THE fundamental package for scientific computing with Python. NUMPY, 2019. Disponível em: https://numpy.org/. Acesso em: 25 mai. 2023.