UNIVERSIDADE DA INTEGRAÇÃO INTERNACIONAL DA LUSOFONIA AFRO-BRASILEIRA

INSTITUTO DE CIÊNCIAS EXATAS E DA NATUREZA CURSO DE LICENCIATURA EM FÍSICA

CONSTANTINO FRANCISCO VASCONCELOS

UTILIZANDO A LINGUAGEM PYTHON PARA RESOLUÇÃO DE PROBLEMAS DE FÍSICA

REDENÇÃO/CE

2022

CONSTANTINO FRANCISCO VASCONCELOS

UTILIZANDO A LINGUAGEM PYTHON PARA RESOLUÇÃO DE PROBLEMAS DE FÍSICA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Licenciatura em Física do INSTITUTO DE CIÊNCIAS EXATAS E DA NATUREZA da UNIVERSIDADE DA INTEGRAÇÃO INTERNACIONAL DA LUSOFONIA AFRO BRASILEIRA, como requisito parcial à obtenção do grau de Graduação em Licenciatura em Física.

Orientador: Prof. Dr. Levi Rodrigues Leite

REDENÇÃO/CE

2022

Universidade da Integração Internacional da Lusofonia Afro-Brasileira
Sistema de Bibliotecas da UNILAB
Catalogação de Publicação na Fonte.

Vasconcelos, Constantino Francisco.

V331u

Utilizando a linguagem Python para resolução de problemas de física / Constantino Francisco Vasconcelos. - Redenção, 2022.
43f: il.

Monografia - Curso de Física, Instituto de Ciências Exatas e da Natureza, Universidade da Integração Internacional da Lusofonia Afro-Brasileira, Redenção, 2022.

Orientador: Prof. Dr. Levi Rodrigues Leite.

1. Ensino de física. 2. Python. 3. Física computacional. I. Título

CE/UF/BSCA CDD 530.07

CONSTANTINO FRANCISCO VASCONCELOS

UTILIZANDO A LINGUAGEM PYTHON PARA RESOLUÇÃO DE PROBLEMAS DE FÍSICA

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Licenciatura em Física do INSTITUTO DE CIÊNCIAS EXATAS E DA NATUREZA da UNIVERSIDADE DA INTEGRAÇÃO INTERNACIONAL DA LUSOFONIA AFRO BRASILEIRA, como requisito parcial à obtenção do grau de Graduação em Licenciatura em Física.

Aprovado em: _____/_____/2022

BANCA EXAMINADORA

_____

Prof. Dr. Levi Rodrigues Leite (orientador)

Universidade da Integração Internacional da Lusofonia Afro-Brasileira – UNILAB

_____

Prof. Dr. Aurélio Wildson Teixeira de Noronha (Examinador)

Universidade da Integração Internacional da Lusofonia Afro-Brasileira – UNILAB

_____

Prof. Dr. Leandro Jader Pitombeira Xavier (Examinador)

Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE

Gostaria de dedicar esse trabalho de conclusão de curso a minha namorada Dja Codé, que já se foi, porém em mim continua e será para sempre. Foi uma das que contribuiu de forma muito significativa nessa conquista. Não sei quanto tempo vai passar para nos reencontramos novamente, mas aqui estão os resultados dos seus esforços, com muita gratidão.

A te Kaky!

AGRADECIMENTOS

Antes de mais nada, agradeço a Deus pela vida e saúde que me deu nessa caminhada como estudante e durante todos esses anos da minha vida.

Agradeço aos meus pais, Francisco José Vasconcelos e de uma forma muito especial a minha querida mamãe, Sato de Carvalho, pela educação e carinho que me prestou durante esses tempos da minha vida dentro e fora do país.

Agradeço à Universidade da Integração Internacional e da Lusofonia Afro-Brasileira-UNILAB, pela oportunidade que me deu para estudar e também pelo apoio financeiro durante esses cincos anos da minha estadia nessa instituição.

Aos professores, pelas aprendizagens, pela motivação, ensinamentos e por terem contribuído para o desenvolvimento reflexivo da minha pessoa. E ainda os agradeço por toda experiência vivida nesses anos.

Ao meu orientador, Levi Leite, por me aceitar como seu orientando, pela paciência e vontade demonstrado na produção deste trabalho de conclusão de curso.

Aos professores Aurélio e Leandro pela participação e contribuições deixadas ao meu TCC.

A nossa querida professora Cinthia Paschoal pelo carinho demonstrado pela nossa turma (2017.1) durante esses anos da nossa formação.

Agradeço aos meus primos e minhas primas que de alguma forma contribuíram significativamente nessa minha trajetória acadêmica, em especial a Carminda Vasconcelos, Silvestre Vasconcelos, Carlos Manga, Adriano Otindio, Anísio Barreto, Elisabete Manga, Elisabete Cudango, Narcisa de Carvalho, Dino Sambu.

Aos meus tios e tias de uma forma geral, em especial ao Djoi de Carvalho, Lai Manga, Osvaldo Manga, Seco de Carvalho, Alfusene de Carvalho, Neto Pereira, Lemas Manga, Carlitos Manga, Fatu Quine Vieira, Antcho Sanha, Filomena de Carvalho.

Meus agradecimentos de uma forma muito especial ao Ivanildo Rui Barbosa, Midana Sambu, Augustinho Cá, Elisio Nhuta pelos apoios que têm me dado durante estes anos do meu curso. Agradeço aos meus colegas da turma e de uma forma especial ao Ildo Ufala, Gaspar Luís, Domingos Fernandes, Sermos Domingos, Mauro Jorge, Antônio Romário, Peter Frotas, Maria Taís, Jonas e a Felisbela pela amizade e todo aprendizado que alcançamos ao longo desses anos, pela motivação e encorajamento de prosseguir no curso.

Aproveito essa linha de agradecimentos para agradecer a mãe do meu filho, Eva Bilatcha, por ter cuidado e cuidar do nosso filho sem a minha presença por perto.

Por fim, agradeço meus professores, manos, do ensino médio que sempre estiveram ao meu lado mesmo com essas distancias entre a gente, de uma forma especial ao professor Jardel Mario, Nixon, e a professora Gracie Semedo. Sem esquecer do meu mano, irmão, Wilquine na Betam.

Gratidão a todos vocês!

RESUMO

Devido à constante evolução das linguagens de programação, é possível sua aplicação para criação de programas que auxiliem no ramo do ensino, especificamente em física. Particularmente, a linguagem Python, que será utilizada neste trabalho, pode ser utilizada na criação de simulações, gráficos e resolução de problemas de física. Python é uma linguagem de programação interpretada e de uso geral que se destacou pela facilidade de programação e versatilidade. O presente trabalho de conclusão de curso (TCC) tem como o objetivo principal resolver cinco problemas de física em Python, sendo dois de mecânica clássica, um de termodinâmica e dois de eletromagnetismo. Para melhor compreensão do conteúdo, os problemas selecionados foram mostrados de duas formas: a primeira forma é analítica, mostrando passo a passo de como resolver o problema e a segunda é de forma numérica em Python permitindo assim que o leitor possa comparar as resoluções, facilitando sua compreensão de como utilizar a linguagem Python.

Palavras Chaves: Python, Ensino de física, Física computacional.

ABSTRACT

Due to the constant evolution of programming languages, it is possible to apply them to create programs that can be useful in the field of education, specifically in physics. Particularly, the Python language, which will be used in this work, can be used to create simulations, graphs and solve Physics problems. Python is an interpreted, general-purpose programming language that stands out for its ease of programming and versatility. The main goal of the present undergraduate thesis (TCC) is to solve five Physics problems in Python, two about classical mechanics, one about thermodynamics and two about electromagnetism. For a better understanding of the content, the selected problems were shown in two ways: the first is analytical, showing step by step how to solve the problem, and the second will be solved numerically in Python, thus allowing the reader to compare the resolutions, facilitating the understanding of how to use the Python language.

Keywords: Python, Physics teaching, Computational Physics.

LISTA DAS FIGURAS

GRÁFICOS

Sumário

Capítulo 1 – Introdução

A Física é uma área do conhecimento que, de modo geral, costuma sofrer rejeição por parte de muitos estudantes, que costumam classificar como uma área "difícil". Além disso, também não é incomum ouvirmos estudantes alegando que as aulas não são cativantes, o que compromete o interesse no assunto. Isso nos leva a se perguntar, como futuros docentes, o que poderia ser feito para tornar as aulas de física mais atraentes. É nesse momento que os softwares e outras tecnologias podem ter grande valor.

> "É de suma importância o uso das tecnologias digitais no Cursos de Licenciatura, visto que, hoje se têm evidências concretas de que as tecnologias com as potencialidades de registro, busca, recuperação e atualização constante de informações, comunicação e produção de conhecimento, abrem novas perspectivas para o desenvolvimento do currículo emancipatório, a prática pedagógica reflexiva, a formação do profissional crítico e a valorização da pesquisa científica" (ALMEIDA, 2014).

Os profissionais da educação em geral, poderiam aprender a programar em sua formação acadêmica, pois através da programação é possível tornar uma disciplina mais interessante por parte dos alunos, quando se trata de aprendizado, uma vez que outra forma de abordar o assunto se torna possível. Costuma-se dizer que, existem os usuários e os criadores de tecnologia. Se quisermos ser vistos não apenas como usuários das tecnologias, devemos pensar em nosso modo de ensinar, como despertar em nossos estudantes, o desejo de não apenas usar um celular, mas também de saber como programá-lo, de não apenas usar jogos eletrônicos, mas também como criar seu próprio jogo e de não apenas usar uma calculadora para resolver uma determinada equação, mas também como criar sua própria calculadora. A programação não é apenas para transformar estudantes em desenvolvedores, mas sim seres pensantes, criativos, inteligentes, com grande facilidade de resolução de problemas (SILVEIRA, 2018).

Python é uma linguagem de programação interpretada e de uso geral que se destacou pela facilidade de programação e versatilidade, por isso, se tornou uma das linguagens mais ensinadas nas universidades. A linguagem Python possui aplicações em muitas áreas, possuindo muitas vezes um certo destaque em comparação com outras linguagens de programação mais simples. Sendo uma linguagem interpretada, usamos o Jupyter Notebook, instalando uma distribuição das

linguagens de programação denominado Anaconda para programar e rodar um código escrito nessa linguagem.

O presente trabalho de conclusão de curso tem como objetivo principal resolver cinco problemas de Física usando a linguagem de programação Python, sendo dois problemas de mecânica clássica, um de termodinâmica e dois de eletromagnetismo.

Capítulo 2 – Sobre a Linguagem Python

2.1 O que é o Python?

Python é uma linguagem de programação interpretada, orientada a objetos, de alto nível e com semântica dinâmica. Foi criado por Guido Van Rossum na década de oitenta, sua simplicidade reduz o grau de dificuldade de criar um programa. Esta linguagem suporta módulos e pacotes, que encoraja a programação modularizada e reuso de códigos. É uma das linguagens que mais tem crescido nos últimos anos devido a sua compatibilidade (funciona na maioria dos sistemas operacionais) e sua capacidade de auxiliar outras linguagens. Python é a linguagem mais popular para análise de dados e devido sua versatilidade conquistou também a comunidade científica (CAELUM, 2020).

2.2 Aspectos históricos

Em 1989 o Holandês Guido Van Rossum iniciou o desenvolvimento do Python com intuito de procurar uma sucessora da linguagem ABC. Logo depois, em 1991 foi lançada a primeira versão pública dessa linguagem (Versão 0.9.0), essa nova versão já incluía classes com heranças, tratamento de exceções e funções, sua característica fundamental é funcionamento modular. Isto permitiu que fosse uma linguagem muito simples e acessível para pessoas com poucos conhecimentos na área de programação, uma característica que permanece até então. O desenvolvimento desta linguagem de programação era liderado e coordenado pessoalmente por Rossum até 2018, a partir de 2019, a empresa possui cinco pessoas que decidem como evolui e se desenvolve a linguagem Python, um conselho que é renovado anualmente. Quando foi lançada a primeira versão definitiva do Python, a popularidade desta linguagem era tal que foi criado um fórum de discussão Python, comp.lang.python, que multiplicou ainda mais o seu número de utilizadores com o tempo.

Devido as demandas e a constante evolução da tecnológica, o Python como qualquer software necessita das atualizações à medida que o tempo passa, para isso, de 1989 até a data presente, 2022, existe várias atualizações das versões dessa linguagem.

A versão 1.0, criado por Guido Van Rossum enquanto trabalhava no Centro de Pesquisa Holandês (CWI), e foi este centro de pesquisa que em 1995 lançou a versão 1.2 do Python. A versão 1.6 do Python teve alguns problemas com o seu tipo de licença até que a Free Software Foundation (FSF) conseguiu mudar Python para uma licença de Software Livre, o que o passou a tornar compatível com GPL.

Em maio de 2000, Guido e o time principal de Python se mudaram para a BeOpen.com para formar o time BeOpen PythonLabs. Em outubro do mesmo ano, o time da PythonLabs se moveu para a Digital Creations onde foi publicada a **Versão 2.0** Python. Uma nova versão que incluía a geração de listas, uma das características mais importantes desta linguagem.

Umas das grandes atualizações na história do Python ocorreu em 2008 com o lançamento da **versão 3.0**, que veio solucionar as principais falhas no design desta linguagem de programação. Embora Python mantenha a sua filosofia, nesta última versão, esta linguagem de programação acumulou formas novas e redundantes para programar o mesmo elemento. Daí a necessidade de novas versões que eliminem esses construtores duplicados. A versão 3.0, quebra a compatibilidade com as versões anteriores da linguagem, uma vez que o código do Python 2.x não deve ser executado necessariamente sem modificações no Python 3.0.

Portanto, a versão que iremos utilizar nesse trabalho será 3.9.5. Essa versão foi lançada em Maio de 2021. Mostraremos passo a passo no apêndice como baixar, o Python e a interpretadora Anaconda Tokio (2021) como Caelum (2020).

2.3 Aplicações

A linguagem de programação Python possui aplicações em várias áreas do conhecimento. “Você pode fazer arte ou engenharia em Python, navegar na web ou calcular seus impostos, escrever palavras ou escrever música, fazer um filme ou fazer a próxima start-up bilionária da Internet” (NEWMANN, 2013). A Fig. 1 mostra algumas empresas que usam o Python para desenvolvimento dos seus negócios.

Figura 1: algumas empresas que usam o Python.

Fonte: google. Disponível em: https://conaenge.com.br/curso-python-para-iniciantes/.

Aplicativos de GUI da área de trabalho: A maioria das distribuições binárias do Python vem com Tk, uma biblioteca GUI padrão. Ele permite que você esboce uma interface de usuário para um aplicativo. Além disso, alguns kits de ferramentas estão disponíveis:

Kivy – para escrever aplicativos multitoque Qt via pyqt ou pyside;

Aplicações científicas e numéricas.

SciPy – Uma coleção de pacotes para matemática, ciências e engenharia;

Pandas – Uma biblioteca de modelagem e análise de dados;

IPython – Um shell poderoso para fácil edição e gravação de sessões de trabalho. Ele também suporta visualizações e computação paralela. Além disso, o NumPy nos permite lidar com cálculos numéricos complexos.

Educação: A valorização e aplicação prática dos conhecimentos adquiridos para um ensino significativo, pegando a realidade como ponto de partida, leva os profissionais da educação a querer aprender cada vez mais. As aplicações da programação Python na educação têm um escopo enorme, pois é uma ótima linguagem para ensinar nas escolas ou até mesmo aprender por conta própria. Graças à sua simplicidade e brevidade pode deixar o aluno mais motivado a aprender, o Python é uma ótima linguagem de programação introdutória.

2.4 Programação na física

Nessa seção vamos nos concentrar em aplicações voltadas ao curso de Licenciatura em Física, enfatizando sua utilidade em uma aula de Física para a resolução de problemas clássicos desta área do conhecimento, tomando como base os conceitos abordados dentro da sala de aula.

A utilização de simulações computacionais na aula de ciência, é defendida por proporcionar um ambiente interativo entre o aluno e seus colegas ou professores. Também permite um processo de ensino e aprendizagem mais dinâmico no qual o aluno pode ser ativo, testar suas hipóteses, obter um feedback rápido, avançar no processo de acordo com suas capacidades e desenvolver habilidades e competências que são exigidas para um bom entendimento do conteúdo (COSTA, 2017).

Tradicionalmente, a física utiliza abordagens experimentais e teóricas para descobrir as causas por trás de um determinado fenômeno. Ser capaz de transformar uma teoria em um algoritmo requer uma visão teórica significativa, compreensão física e matemática detalhada e domínio da arte da programação (RUBIN, 2015). Porém a simplicidade do Python ajuda, instiga, os que nunca têm contato com programação a programar, é fácil de baixar, instalar e de trabalhar.

No Python, existem alguns recursos poderosos que podem facilitar a vida de um Físico, recursos que permitem construção dos vetores e gráficos; montagem e cálculo de matriz; cálculo de funções básicas (função de primeiro grau e segundo grau), cálculo de derivadas, cálculo de integrais e outros. Ao representar o modelo que descreve um determinado fenômeno físico no formato multimídia através de uma simulação computacional, utilizamos um conjunto de símbolos e ícones na identificação visual dos conceitos a serem abordados. Esse esquema visual deve corresponder ou ser equivalente, com suas simplificações inerentes ao ambiente virtual, ao modelo do fenômeno que queremos representar de forma idealizada (SOUSA, 2010).

Portanto é importante que a compreensão do aplicativo criado seja precisa e rápida, sem que seja necessário investir tempo demasiado na explicação a respeito do funcionamento do simulador. Nesse sentido, é muito útil que o programa seja intuitivo, com ícones, figuras e ilustrações que, em conjunto, levem o usuário a perceber claramente o ambiente no qual o fenômeno é apresentado.

2.5 Interpretador

Para programar e rodar um sistema escrito nessa linguagem é necessária a utilização de um interpretador.

Python é uma linguagem interpretada e por outro lado uma linguagem compilada. Existe um processo, ou seja, o compilador traduz a linguagem Python em linguagem de máquina - o código Python é traduzido em um código intermediário que deve ser executado por uma máquina. O interpretador faz esta tradução em tempo real para código de máquina, já o compilador traduz o programa inteiro em código de máquina de uma só vez e então o executa, criando um arquivo que pode ser rodado.

Existem vários intérpretes dessa linguagem de programação, porém neste trabalho usamos Jupyter Notebook instalando uma distribuição das linguagens de programação para computação científica denominado Anaconda (CAELUM, 2020).

Capítulo 3 – Resolução dos Problemas

A Física é uma ciência que possui teorias bonitas, riquíssimas e muito bem elaboradas por diversos ícones dessa ciência. Tentar resolver um exercício ou um problema de Física sem conhecer a teoria que explica sobre o fenômeno em questão pode ser o caminho certo para o erro (JÚNIOR, 2022). Entretanto, neste capítulo faremos resolução dos problemas escolhidos de duas formas diferentes: Primeiramente, iremos resolver de uma forma analítica, mostrando passo a passo e em seguida, segunda forma, iremos apresentar imagem dos códigos que resolvem o mesmo problema no Python. A maioria dos problemas são de nível médio (dois pontinhos) do livro Fundamento de Física, nona edição, Hallidy.

Na segunda forma de resolução dos problemas, no Python, tentamos trazer para cada resolução uma nova função, comandos e variáveis a fim de demonstrar quão é amplo trabalhar com essa linguagem de programação (Python). No entanto, existe várias formas de resolver um determinado problema no Python, a forma escolhida neste trabalho para resolução dos problemas de substituição simples, para o cálculo de derivada, integral definida, construções dos gráficos, formatação dos números decimal e cálculo com os vetores não seriam as únicas formas, você pode encontrar num outro trabalho uma outra forma para resolução de uma das funções acima. Dessa maneira, nós escolhemos a forma mais simples para que um leitor que não sabe nada da linguagem (Python) possa entender o cálculo e a função de comandos utilizados.

Problema 01- Movimento Uniformemente Variado (MUV).

Um avião a jato de grande porte precisa atingir uma velocidade de 500 km/h para decolar, e tem uma aceleração de 4 m/s^2. Quanto tempo ele leva para decolar e que distância percorre na pista até a decolagem?

- **Solução analítica:**

Temos valor de aceleração (a = 4m/s^2), e o valor da velocidade (v = 500km/h = 139m/s), lembrando que a aceleração é diretamente proporcional a variação da velocidade e inversamente proporcional a variação de tempo, ou seja,

$$a_{media} = \frac{\Delta v}{\Delta t} \qquad (1)$$

explicitando o t na equação (1) obtendo,

$$t = \frac{v}{a} \qquad (2)$$

Substituindo os valores de velocidade e da aceleração na equação (2), temos que,

$$t = \frac{v}{a} = \frac{139}{4} = 34{,}75s$$

Portanto, o tempo necessário para que o avião se decolar é $34{,}75s$.

A distância percorrida pelo avião até decolar será dada pela equação horaria da posição no movimento uniforme variado (UMV),

$$s = s_0 + v_0 t + \frac{at^2}{2} \qquad (3)$$

Como no instante inicial a posição inicial e a velocidade inicial são zero ($s_0 = 0\ e\ v_0 = 0$).

Logo,

$$s = 0 + (0)(34{,}75) + \frac{(4)(34{,}75)^2}{2} = 2{,}415m$$

Portanto, a distância percorrida até o ponto de decolagem é, $2{,}415m$.

- **Solução numérica:**

Para resolução dos nossos problemas em Python vamos importar algumas bibliotecas e alguns comandos como:

“From sympy import*” – é uma função que nos permite importa a biblioteca “Sympy”.

SymPy é uma biblioteca para computação simbólica. Possui ferramentas que variam do cálculo de aritmética simbólica básica, álgebra, funções matemática, física quântica e entre outros.

“Import matplotlib.pyplot as plt” – é uma função que permite importar a biblioteca “Matplotlib” que por sua vez contém uma grande coleção de módulos para criar representações gráficas de dados de alta qualidade em vários formatos e visualização de dados em Python.

plt.rcParams['figure.figsize'] = (10,6) – é a função que nos permite colocar tamanho (largura, altura) do gráfico numa forma desejada.

vf – recebe o valor de velocidade final.

a – recebe o valor de aceleração.

print() – é a função que imprime a resposta de um determinado cálculo.

Observação: os comentários serão feitos nas figuras depois de um cardinal (#).

Figura 2: mostra os códigos no Python para o cálculo de tempo no problema 01.

```
#PROBLEMA 01:
from sympy import*
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = (10,6)

#DADOS PARA CÁLCULO DE TEMPO DE DECOLAGEM DE AVIÃO:
t = Symbol("t")          #DEFINIMOS O T COMO UM SIMBLO
vf = 139                 #EM METROS POR SEGUNDOS
a = 4                    #EM METROS POR SEGUNDOS AO QUADRADO

#SOLUÇAO:
eq1 = Eq(vf/t, a)        #MONTAGEM DA EQUAÇÃO 01
k = solve(eq1, t)        #PEDINDO QUE K RESOLVE eq1 PARA t.

print("O tempo necessário é:", k[0],"s.") #PEDINDO QUE ELE IMPRIME A RESPOSTA
```

```
O tempo necessário é: 139/4 s.
```

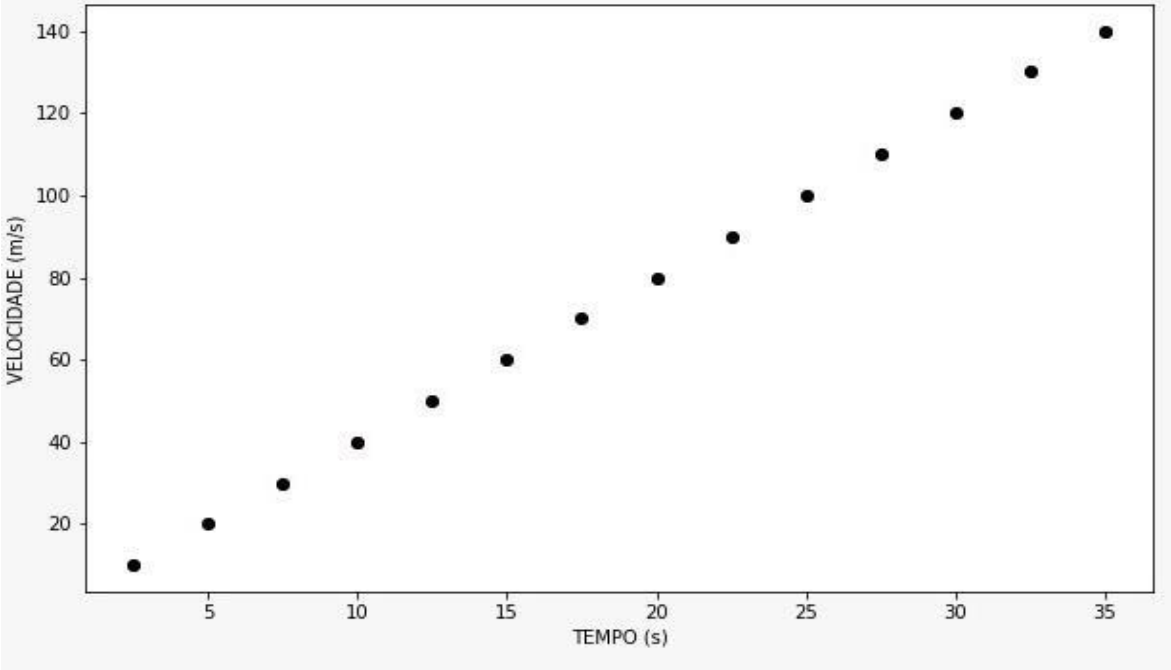
Fonte: autor

Figura 3: mostra os códigos para construção de gráfico 01.

```
#CONSTRUÇÃO DE GRÁFICO:
V = 0                       #EM METROS
dv = 10                     #PEDINDO QUE dv VARIA DE 10 EM 10+
while V < vf:               #PEDINDO QUE O PROCESSO CONTINUA ENQUANTO V<vf
    V = V + dv              #CRIAMOS UMA NOVA FUNÇÃO PARA V
    T = (V/a)               #MANTEMOS A eq1 NA FORMA PADRÃO
    plt.scatter(T, V)       #PEDINDO QUE PLOTAR O GRÁFICO DE T x V
    plt.scatter(T, V, color="black")  #ATRIBUIMOS A CÔR PRETA PARA O GRAFICO
    plt.xlabel('TEMPO (s)')           #DEFÍNIMOS O EIXO TEMPO EM SENGOS (s)
    plt.ylabel('VELOCIDADE (m/s)')    #DEFÍNIMOS O EIXO VELOCIDADE EM m/s
    plt.savefig('imagem 01 python')   #PEDINDO QUE SALVE A IMAGEM DO GRÁFICO
```

Fonte: autor

Gráfico 01: mostra variação de velocidade com o tempo.



Fonte: autor

Os laços de repetições mais populares e usados em Python são: ***for*** e ***while***. Na Fig. 4, usamos uma dessas estrutura, ***for()***, na construção do gráfico para executar uma tarefa repetidamente enquanto a condição definida é atendida.

Figura 4: mostra os códigos para o cálculo de espaço percorrido e construção de gráfico 02.

```
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = (10,6)

#DADOS PARA CÁLCULO DA DISTÂNCIA DE DECOLAGEM.
S0 = 0           #EM METROS
t = 139/4        #EM SEGUNDOS
v0 = 0           #EM METROS POR SEGUNDOS
a = 4            #EM METROS POR SEGUNDOS AO QUADRADO

#SOLUÇÃO:
Sf = (S0 + v0*t + a*(t**2)/2)  #EQUAÇÃO PARA CÁLCULO DE ESPAÇO.

print("O espaço percorrido é de:", Sf,"m.") #IMPRIME A RESPOSTA
```

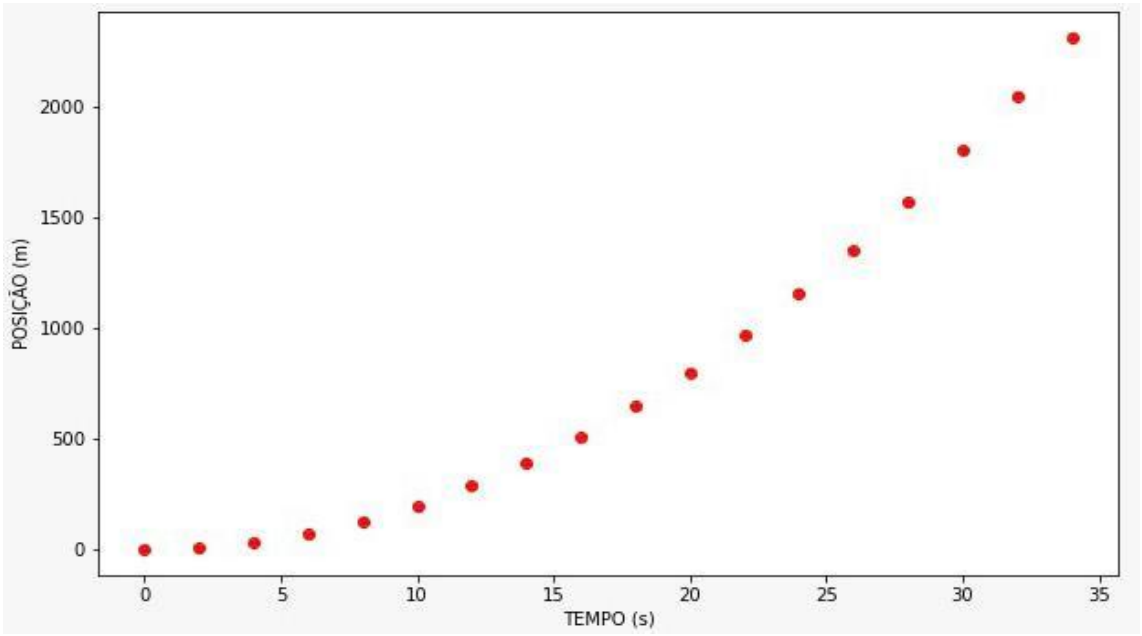
```
O espaço percorrido é de: 2415.125 m.
```

```
#CONSTRUÇÃO DE GRÁFICO
for dT in range(0, 36, 2):  #ATRIBUINDO dt OS VALORES DE 0 À 36 PULANDO 2 EM 2.
    T = dT                  #ATRIUINDO OS VALORES PARA TEMPO
    S = (S0 + v0*T + a*(T**2)/2) #VALORES DE POSIÇÃO P/ CADA INTANTE T.

    plt.scatter(T, S, color="red")    #PEDINDO QUE PLOTE O GRÁFICO VERMELHO
    plt.xlabel('TEMPO (s)')           #DEFÍNIMOS O EIXO TEMPO EM SEGUNDOS
    plt.ylabel('POSIÇÃO (m)')         #DEFÍNIMOS O EIXO posição EM metros
    plt.savefig('imagem 02 python')   #PEDINDO QUE SALVE A IMAGEM COMO
```

Fonte: autor.

Gráfico 02: mostra variação de posição com o tempo.



Fonte: autor

Problema 02: Trata sobre velocidade instantânea.

A posição de uma partícula que se move ao longo do eixo *x* é dada por $\boldsymbol{x} = 9{,}75 + 1{,}50t^3$, onde *x* está em centímetros (cm) e *t* em segundos. Calcule (a) a velocidade média durante o intervalo de tempo de $\boldsymbol{t}$ = 2,00 s a $\boldsymbol{t}$ = 3,00 s; **(b)** a velocidade instantânea em $\boldsymbol{t}$ = 2,00 s; (c) a velocidade instantânea em *t* = 3,00 s; (d) a velocidade instantânea em $\boldsymbol{t}$ = 2,50 s; (e) a velocidade instantânea quando a partícula está na metade da distância entre as posições em *t =2,00 s e t =3,00s*. **(f)** Plote o gráfico de $\boldsymbol{x}$ em função de t e indique suas respostas graficamente.

- **Solução analítica:**

Segundo o enunciado, a função posição é dada por,

$$x(t) = 9{,}75 + 1{,}50t^3 \qquad (4)$$

Velocidade média é igual à variação de posição (Δx) sobre a variação de tempo (Δt).

$$v_{méd} = \frac{\Delta x}{\Delta t} \qquad (5)$$

A velocidade instantânea (v_{inst}) da partícula é a derivada da função posição (equação 4),

$$v_{inst} = \frac{dx}{dt} = 4{,}50t^2$$

Portanto,

$$v_{inst} = 4{,}50t^2 \qquad (6)$$

(a) Substituindo t = 2,0s e t = 3,0s na equação 4, obtemos que:

$$para\ t_1 = 2{,}0s \quad => \quad x_1 = 21{,}75cm$$

$$para\ t_2 = 3{,}0s \quad => \quad x_2 = 50{,}25cm$$

Logo, a velocidade média no instante $2{,}0s \leq t \leq 3{,}0s$ é,

$$v_{méd} = \frac{\Delta x}{\Delta t} = \frac{x_2 - x_1}{t_2 - t_1} = \frac{50{,}25cm - 21{,}75cm}{3{,}0s - 2{,}0s} = 28{,}25cm.$$

(b) Usando a equação 6 temos que,

$$para\ t = 2{,}0s \quad => \quad v_{inst} = 4{,}50(2{,}0)^2 = 18{,}0cm/s$$

(c) $para\ t = 3{,}0s \quad => \quad v_{inst} = 4{,}50(3{,}0)^2 = 40{,}50cm/s$

(d) $para\ t = 2{,}5s \quad => \quad v_{inst} = 4{,}50(2{,}50)^2 = 28{,}1cm/s$

(e) O instante em que a partícula está a meio caminho (t_m) entre $x_2\ e\ x_1$, será calculado de seguinte forma,

$$x_m = \frac{x_2 + x_1}{2} = \frac{50{,}25 + 21{,}75}{2} = 36cm$$

Fazendo $x_m = x(t)\ e\ t = t_m$na equação (1), em seguida explicitando t_m obtém-se,

$$x_m = 9{,}75 + 1{,}50{t_m}^3 => t_m = 2{,}596s.$$

Portanto, a velocidade instantânea nesse instante (t_m) é,

$$v_{inst} = 4{,}50(2{,}596)^2 = 30{,}3cm/s$$

- **Solução Numérico:**

NumPy - é uma biblioteca para a linguagem Python com funções para se trabalhar com computação numérica.

diff() – é uma função dentro da biblioteca SymPy que pode ser usada para calcular a derivada de uma função.

t, tm – atribuímos a eles função de símbolo, ou seja, são variáveis.

Figura 5: mostra os códigos e os dados para resolução de problema 02.

```
#PROBLEMA 02
from sympy import*
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = (10,6)
import numpy as np
t = Symbol("t")
tm = Symbol("tm")
```

```
#DADOS
#x(t) = 9.75 + 1.50t**3   #Função possição da partícula
x1 = 21.75                #Em Centimetros
x2 = 50.25                #Em Centimetros
t1 = 2.0                  #Em Segundos
t2 = 3.0                  #Em Segundos
t3 = 2.50                 #Em Segundos
```

Fonte: Autor

Na Fig. 6, o ***sol*** recebe uma função que permite calcular a derivada e ***V*** recebe a equação para o cálculo da velocidade média no item (a).

Figura 6: mostra cálculo dos itens (a), (b) e (c) de problema 02.

```
#A Velocidade instantania (V) será dada por,
#V=dx/dt. Logo,

sol = diff(9.75 + 1.50*t**3, t)
print("Derivando a função X(t), temos que V é iguala a", sol)
```

```
Derivando a função X(t), temos que V é iguala a 4.5*t**2
```

```
#SOLUÇÃO:
#(a) Velocidade Média é dada por,
#V = Δx/Δt

V = (x2 - x1)/(t2 - t1)

print("A velocidade média no instante t1 e t2 é,", V,"cm/s.")
```

```
A velocidade média no instante t1 e t2 é, 28.5 cm/s.
```

```
#(b)A velocidade instantânea para

t = 2.0           #em segundos
V1 = 4.5*t**2     #velocidade em função de t.
print("Para t = 2.0s, temos que V =",V1,"cm/s.")
```

```
Para t = 2.0s, temos que V = 18.0 cm/s.
```

```
#(c)A velocidade instantânea para

t = 3.0           #em segundos
V2 = 4.5*t**2     #velocidade em função de t.
print("Para t = 3.0s, temos que V =",V2,"cm/s.")
```

```
Para t = 3.0s, temos que V = 40.5 cm/s.
```

Fonte: Autor

Figura 7: mostra cálculo dos itens (d), (e) e dados para construção de gráfico.

```
#(d)A velocidade instantânea para

t = 2.50          #em segundos
V3 = 4.5*t**2     #velocidade em função de t.
print("Para t = 2.50s, temos que V =",V3,"cm/s.")
```

```
Para t = 2.50s, temos que V = 28.125 cm/s.
```

```
#(e)O instante (tm) em que a partícula está a meio caminha é,
#Xm=(x1+x2)/2=36. Para Xm=9.75+1.5*tm**3 implica que,
#tm=(Xm - 9.75)/1.5)**0.333   logo,

tm = ((36 - 9.75)/1.5)**0.333   #atribuimos tm uma equação
V4 = 4.5*tm**2                #velocidade instantania em função de tm
print("Portanto, a velocidade instantâne nesse instante é",V4,"cm/s.")
```

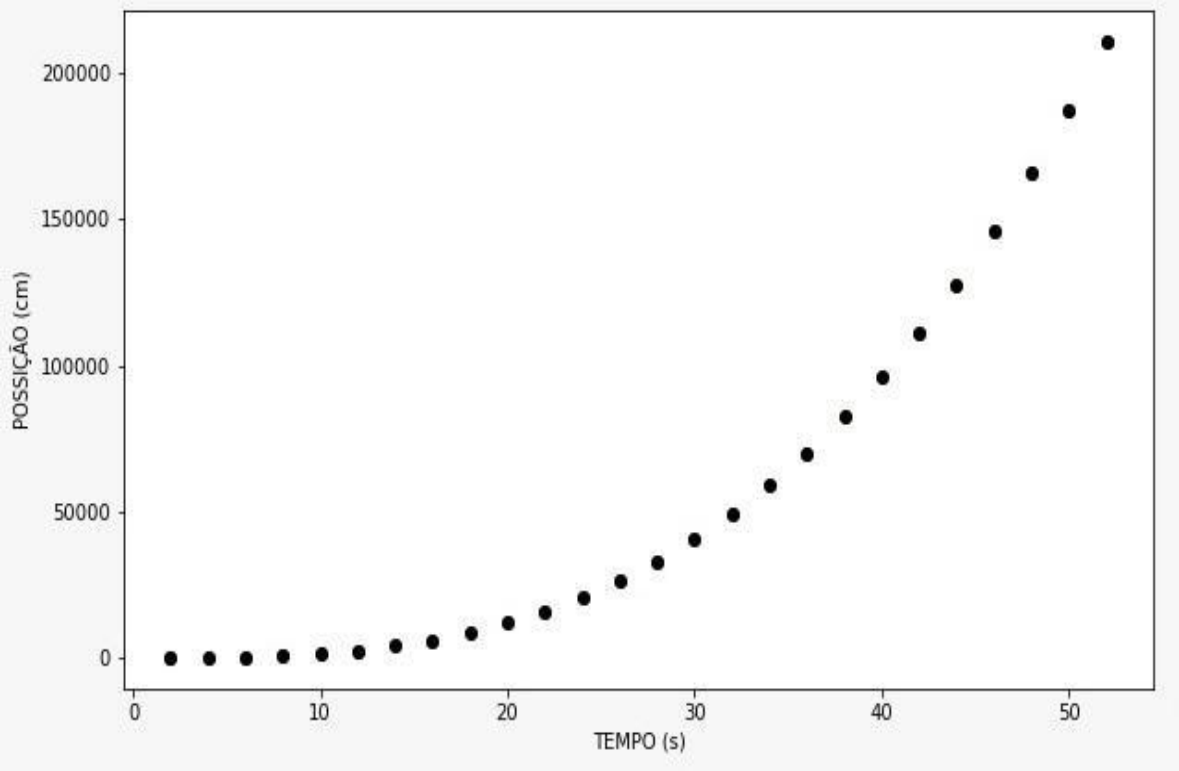
```
Portanto, a velocidade instantâne nesse instante é 30.274421572469986 cm/s.
```

```
#DADOS PARA CONSTRUÇÃO DE GRÁFICO
t0 = 0                          #tempo zero para no inicio
dt = 2                         #dt variando de 2s em 2s
while t0 < x2:                #pendito que repita enquanto t0 < x2
    t0 = t0 + dt                 #definimos a nova função p/ t0
    x = 9.75 + 1.50*t0**3     #recebendo valores de x p/ cada t0
    plt.scatter(t0, x)      #pedindo para plotar o gráfico (t0,x)
    plt.scatter(t0, x, color="black") #dando côr preta p/ gráfico
    plt.xlabel('TEMPO (s)')       #dando nome p/ eixo horizontal
    plt.ylabel('POSSIÇÃO (cm)')  #dando nome p/ eixo vertical
    plt.savefig('imagem 02 python') #pedindo p/ salvar o gráfico
```

Fonte: Autor

while **–** é uma estrutura de repetição que pode ser usada para tarefas repetitivas em um loop contínuo, o loop continua até que satisfaz a condição definida.

Gráfico 03: mostra variação de posição com o tempo.



Fonte: Autor

Problema 03: Cálculo de calor específico dependente da temperatura.

O calor específico de uma substância varia com a temperatura de acordo com a equação, $c = 0{,}20 + 0{,}14T + 0{,}023T^2$ com T em℃ e c em cal/g.K. Determine a energia necessária para aumentar a temperatura de 2,0g desta substância de 5,0℃ para 15,0℃.

- **Solução analítica:**

O calor necessário, energia, para aumentar a temperatura da substância pode ser calculado integrando o calor específico,

$$\int_{T_i}^{T_f} mc\ dT \qquad (7)$$

Para,

$T_i = 5{,}0℃$; $T_f = 15℃$; $m = 2{,}0g$

Substituindo os valores na nossa equação 7, obtemos

$$Q = m\int_{T_i}^{T_f} c\ dT$$

$$= 2\int_{5}^{15} 0{,}20 + 0{,}14T + 0{,}023T^2\ dT$$

$$= 2\left[(0{,}20T + 0{,}070T^2 + 0{,}00767T^3)\right]_5^{15} = 82cal$$

Portanto, a energia necessário para aumentar a temperatura de 2,0g da nossa substancia é 82 cal.

- **Solução Numérico:**

Para resolução deste problema em Python, usamos algumas funções para o cálculo de integral como também para construção de gráfico.

***(Integral(c)).doit()*-** é uma das função que nos permite calcular a integral no Python.

evalf(). **–** é uma função que avalia uma determinada expressão numérica até uma determinada precisão de ponto flutuante.

nunpy **-** é uma biblioteca auxiliar que nos permite a definir intervalos de integrais definidos.

plt.fill_between(T, f(T), where=[(T > 5) and (T < 15) for T in T])- esse função permite a marcação da área azul no gráfico abaixo

Figura 8: mostra cálculo de calor especifico através de integral numérica.

```
#PROBLEMA 03:
from sympy import*
import matplotlib.pyplot as plt
import numpy as np
T = Symbol("T")
```

```
#DADOS
c = (0.2 + 0.14*T + 0.023*T**2)      #CALOR ESPECIFICO(c)
m = 2.0                              #EM GRAMAS(g)
Ti = 5.0                             #EM oC
Tf = 15                              #EM oC
```

```
#SOLUÇÃO:
Q = (m*(Integral(c, (T, 5, 15)))).doit().evalf(3) #cálculo de integral

print("O calor necessário para aumentar a temperatura é:", Q,"cal.")
```

```
O calor necessário para aumentar a temperatura é: 81.8 cal.
```

Fonte: Autor

Figura 9: mostra os códigos usado para construção do gráfico 04.

```
import matplotlib.pyplot as plt
import numpy as np
plt.rcParams['figure.figsize'] = (10,6)
```

```
def f(T):                  #definimos uma função f(T)
    return (0.2 + 0.14*T + 0.023*T**2)   #definição de f(T)

T = np.linspace(5, 15, 1000) #cria uma seqüência de números espaçado.
plt.plot(T, f(T))            #plote o gráfico
plt.axhline(color="black")   #atribuindo a côr preta p/ gráf.
plt.fill_between(T, f(T), where=[(T > 5) and (T < 15) for T in T])
plt.xlabel('TEMPERATURA (oC)') #dando o nome p/ eixo horizontal
plt.ylabel('CALOR ESPECIFÍCO (cal/g.oC)') #dando nome p/ eixo vertical
plt.savefig('imagem 04 python')    #pedindo p/ que salve o gráf.
```

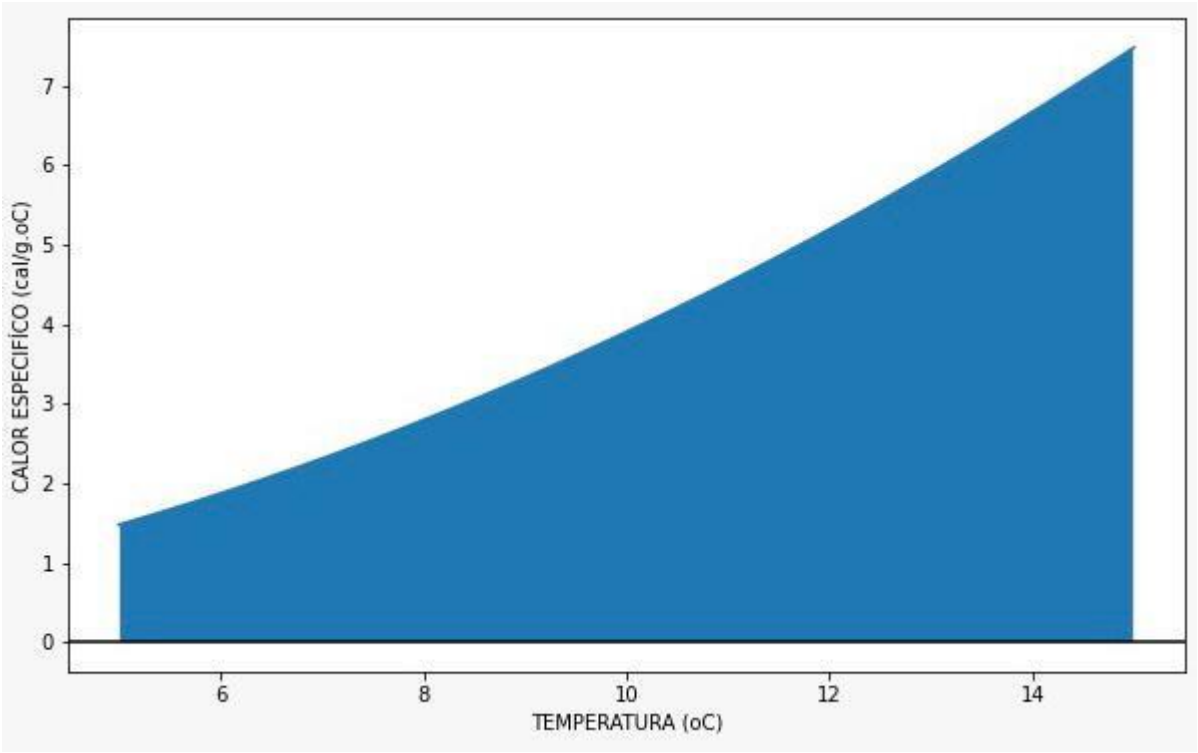
Fonte: Autor

linspace **–** essa função na Fig. 9 nos retorna um número de amostra espaçado uniforme em um intervalo especificado.

plt.fill_between **–** essa função preencha a área do gráfico a baixo.

where **–** delimita o espaço de preenchimento de acordo com intervalo criado.

Gráfico 04: mostra aumento de calor específico de acordo com a temperatura.



Fonte: Autor

Problema 04: Dipolo elétrico sub a influência de um campo elétrico.

Um dipolo elétrico de momento dipolar $\vec{\rho} = (3{,}00\hat{\imath} + 4{,}00\hat{\jmath})(1{,}24\text{x}10^{-30}\text{C.m})$ é submetido a um campo elétrico $\vec{E} = (4000\text{N/C})\hat{\imath}$. (a) Qual é a energia potencial do dipolo elétrico? (b) Qual é o torque que age sobre o dipolo? (c) Se um agente externo faz girar o dipolo até que o momento dipolar seja $\vec{\rho} = (-4{,}00\hat{\imath} + 3{,}00\hat{\jmath}])(1{,}24\text{x}10^{-30}\text{C.m})$, qual é o trabalho realizado pelo agente externo?

- **Solução analítica:**

(a) A energia potencial é dada pela,

$$U = -\vec{\rho}.\vec{E} \qquad (8)$$

Fazendo as seguintes relações $\hat{\imath}.\hat{\imath} = 1 \;\; e \;\; \hat{\jmath}.\hat{\imath} = 0$ em seguida, substituindo os valores numéricos na equação (9) obtém-se,

$$U = -\vec{\rho}.\vec{E}$$

$$= -[(3{,}00\hat{\imath} + 4{,}00\hat{\jmath})(1{,}24\text{x}10^{-30})].(4000)\hat{\imath}$$

$$= -[(12000)(1{,}24\text{x}10^{-30})] = 1{,}488\text{x}10^{-26}J$$

Portanto,

$$U = 1{,}488\text{x}10^{-26}J.$$

(b) O torque é dado pelo produto vetorial entre o vetor momento e o vetor campo elétrico,

$$\vec{\tau} = \vec{\rho}x\vec{E} \qquad (9)$$

Fazendo,

$$\hat{\imath}x\hat{\imath} = 0 \;\; e \;\; \hat{\jmath}x\hat{\imath} = -\hat{k}$$

Logo,

$$\vec{\tau} = \vec{\rho}x\vec{E}$$

$$= [(3{,}00\hat{\imath} + 4{,}00\hat{\jmath})(1{,}24\text{x}10^{-30})]\text{x}(4000)\hat{\imath}$$

$$= \left(-16000\hat{k}\right)(1{,}24\text{x}10^{-30})$$

Portanto,

$$\vec{\tau} = (1{,}98\text{x}10^{-26}N.m)\hat{k}$$

(c) O trabalho realizado pelo agente externo quando faz girar o dipolo até que o momento dipolar seja,

$$\overrightarrow{\rho_f} = (-4{,}00\hat{\imath} + 3{,}00\hat{\jmath})(1{,}24\text{x}10^{-30}\text{C.m})$$

É dado por,

$$W = \Delta U = \Delta\left(-\vec{\rho}.\vec{E}\right)$$

$$W = \left(\overrightarrow{\rho_i} - \overrightarrow{\rho_f}\right).\vec{E} \qquad (10)$$

Fazendo a diferença entre o momento dipolar inicial e o momento final em seguida multiplicar por vetor campo elétrico, obtemos,

$$W = [[(3{,}00\hat{\imath} + 4{,}00\hat{\jmath}) - (-4{,}00\hat{\imath} + 3{,}00\hat{\jmath})](1{,}24\text{x}10^{-30})].\vec{E}$$

$$= (7{,}00\hat{\imath} + 1{,}00\hat{\jmath}).(4.96\text{x}10^{-28}\hat{\imath})$$

$$W = 3.472\text{x}10^{-27}J$$

- **Solução Numérico:**

Figura 10: *mostra os dados e os códigos para o cálculo de item (a).*

```
#PROBLEMA 04:
from sympy import*
import numpy as np
```

```
#DADOS:

p = [3, 4]                  #vetor momento em C*m
E = [4000, 0]              #vetor campo elétrico em N/C
```

```
#SOLUÇÃO
#(a) Energia Potencial:

r = np.dot(p, E)*-1.24*10**-30 #calculo de produto escalar

print("(a) A energia potêncial é igual a", r,"J.")
```

```
(a) A energia potêncial é igual a -1.488e-26 J.
```

Fonte: Autor

array **-** é uma estrutura de dados fundamental e muito importante na maioria das linguagens de programação. Através dela podemos converter lista, criar matrizes e muito mais.

Na Fig. 11, usamos a função ***np.array()*** para converter a lista de números em uma matriz a fim de poder fazer a diferença entre ***pi*** e ***pf***.

Figura 11: mostra os dados e os códigos para o cálculo de item (b) e (c).

```
#(b)Torque:
sol = np.cross(p, E)*-1.24*10**-30 #calculo de produto vetorial

print("O torque é,(",sol,"N.m)k̂")
```

```
O torque é,( 1.9840000000000002e-26 N.m)k̂
```

```
#(c)Trabalho pelo agente externo será dada por,
pi = np.array([3, 4])              #vetor momento em C*m
pf = np.array([-4, 3])             #vetor momento em C*m
E1 = np.array([4000, 0])           #Campo Elétrico em N/C

Δp = pi - pf                       #diferença entre pi e pf
w = np.dot(Δp, E1)*(1.24*10**-30)  #calculo de produto escalar

print("Trabalho realizado pelo agente externo é,",w,"J.")
```

```
Trabalho realizado pelo agente externo é, 3.472e-26 J.
```

Fonte: Autor

Problema 05: Leis de Kirchhoff.

Uma célula solar produz uma diferença de potencial de 0,10V quando um resistor de 500Ω é ligado a seus terminais e uma diferença de potencial de 0, 15 V quando o valor do resistor é 1000Ω. Determine (a) a resistência interna e (b) a força eletromotriz da célula solar. (c) A área da célula é 5,0 cm^2 e a potência luminosa recebida é 2,0 mW/cm^2. Qual é a eficiência da célula ao converter energia luminosa em energia térmica fornecida ao resistor de 1000Ω?

- **Solução analítica:**

(a) Seja V_1 a diferença de potencial da célula solar quando foi ligada a resistência R_1de 500Ω e V_2a diferença de potencial da célula quando foi ligada a resistência R_2 de 1000Ω.

Logo,

$$V = \in - \text{ir} \ , \qquad i = V/R$$

Implica que,

$$V = \in - \frac{V}{R} r \qquad (11)$$

Onde ∈ é a força eletromotriz da célula solar e r é a resistência interna. Logo, criando sistema para V_1 e V_2, obtém-se

$$\begin{cases} V_1 = \in - \left(\frac{V_1}{R_1}\right) r \\ V_2 = \in - \left(\frac{V_2}{R_2}\right) r \end{cases} \qquad (12)$$

Resolvendo o nosso sistema para *r* na equação acima (12) temos que,

$$r = \frac{R_2 R_1 (V_2 - V_1)}{(R_2 V_1 - R_1 V_2)} \qquad (13)$$

Substituindo os valores numérico na equação 13, obtemos

$$r = \frac{R_2 R_1 (V_2 - V_1}{(R_2 V_1 - R_1 V_2)} = \frac{2500}{25} = 1000\Omega = 1\text{x}10^3\Omega.$$

(b) A força eletromotriz da célula pode ser calculada usando qualquer uma das equações no sistema (9). Ou seja,

$$V_1 = \in - \left(\frac{V_1}{R_1}\right) r$$

$$0{,}10 = \in - \left(\frac{0{,}10}{500}\right) 1000$$

Portanto,

$$\in = 0{,}10 + 0{,}2 = 0{,}30V.$$

(c) A eficiência é dada por,

$$n = \frac{P_{for}100\%}{P_{total}} \qquad (14)$$

Onde P_{for} é a potência que o resistor recebi e P_{total} é a potência total da célula solar.

Logo,

$$P_{for} = RI^2 \ , \qquad i = V/R$$

$$=> \qquad P_{for} = \frac{V^2}{R} = \frac{(0{,}15)^2}{1000} = 2{,}25x10^2W$$

E por outro lado, temos que

$$P_{lum} = \frac{P_{total}}{A} \qquad (15)$$

Na qual P_{lum} é a potência luminosa dada no problema e A área da célula em centímetros ao quadrado. Substituindo os valores na equação (12) em seguida calcular potência total,

$$P_{total} = A.P_{lum} = (5)(2x10^{-3}) = 10x10^{-3}W = 10mW$$

Tendo os valores numérico de potência total e a potência que o resistor recebe, portanto, substituindo na equação (11), temos que,

$$n = \frac{(2{,}25x10^2)100\%}{10x10^{-3}} = 0{,}225\%$$

- **Solução Numérica:**

Para resolução desse problema no Python, usamos três modos diferentes para formatar a quantidade de casas decimais de um número decimal através das seguintes funções:

"round()" – é função que retorna um número de ponto flutuante que é uma versão arredondada do número especificado, com o número especificado de decimais.

Figura 12: mostra os dados e os códigos para o cálculo de item (a).

```
#PROBLEMA 05:
from sympy import*
```

```
#DADOS
V1 = 0.10     #potencial 01 em (V)
R1 = 500      #resistor 01 em (Ω)
V2 = 0.15     #potencial 02 em (V)
R2 = 1000     #resistor 02 em (Ω)
A = 5.0       #área em cm**2
Pf = 2.0      #potência fornecida em (W/cm**2)
```

```
#(a) RESISTENCIA INTERNA (r)

#V1 = e - (V1/R1)r   equação 01
#v2 = e - (V2/R2)r   equação 02
#IGUALANDO A NOSSA eq. (01) COM (02)
#E EXPLICITANDO r. TEMOSOS,
#r = V1 - V2/(V2/R2 - V1/R1)   equação 03

t = V1 - V2           #numerador de eq. 03
t2 = (V2/R2 - V1/R1)  #denominador de eq, 03

r = t/t2              #calculo de resistencia interna
valor = round(r,3)    #pedindo para que redonte o resltado
print("(a) A resistencia interna é igual a:",valor,"Ω!")
```

```
(a) A resistencia interna é igual a: 1000.0 Ω!
```

Fonte: Autor

Figura 13: mostra os códigos para o cálculo de item (b) e (c).

```
#(b) FORÇA ELETROMOTRIZ(e):
e = V1 + (V1/R1)*r

print(f"(b) A força eletromotriz da celula é:{e:.2f}V.")
```

```
(b) A força eletromotriz da celula é:0.30V.
```

```
#(c) A EFICIÊNCIA(n) SERÁ DADA PELA SEGUINTE EQUAÇÃO:
Pt = Pf*A*R2          #potência total (Pt)
P = (V2**2)*(100)     #potência que o resistor recebi
n = P/Pt              #cálculo de eficiência

print("(c) A eficiência é {:.6f}%".format(n))
```

```
(c) A eficiência é 0.000225%
```

Fonte: Autor

Note que, nos itens (b) e (c) usamos formas diferentes para a formatação dos números decimal.

No item (b) usamos “**print(f"(b) A força eletromotriz da celula é:{e:.2f}V.")**” , o uso do 'f' antes da mensagem que virá e também a variável 'e' que vai diretamente dentro das chaves junto com a formatação de 2 casas decimais depois da vírgula.

Para o item (c) temos “**print("(c) A eficiência é {:.6f}%".format(n))**”. Ou seja, usamos o format fora da mensagem e dizemos que nos retorna um valor com apenas 6 casas decimais após a vírgula.

4 CONCLUSÃO

Devido a versatilidade e simplicidade que a linguagem Python possui, não seria um exagero sugerir que fosse uma ferramenta obrigatória para aprender em um curso de Licenciatura em Física visto que, cada vez mais a programação e a educação estão caminhando conjuntamente. Seu uso para resolução dos problemas, exercícios e criação das simulações dentro da sala de aulas ou dentro de um laboratório pode vir a tornar-se mais uma ferramenta com o intuito de motivar e ajudar os alunos a compreenderem os conteúdos de uma forma mais didático e simples, pois exige um cálculo no momento da criação. Esses e os demais cálculos farão com que o aluno tenha uma ideia melhor do que lhe passava de alguma forma despercebido dentro da sala de aulas. Assim sendo, conseguimos demonstrar a versatilidade e o uso desta linguagem de programação (Python) na educação, de uma forma particular, no curso de Licenciatura em Física resolvendo cincos problemas de algumas áreas da Física (mecânica, termodinâmica e eletromagnetismo), com ajuda de alguns comandos importados da biblioteca Sympy e Nunpy. Demonstramos também, uma forma para o cálculo das derivadas, uma forma para o cálculo de integral definidas, algumas formas para construção dos gráficos, algumas formas para formatação dos números decimal e algumas formas para os cálculos dos problemas de substituições.

REFERÊNCIAS

SILVEIRA, J.L.P. IMPORTÂNCIA DE PROGRAMAÇÃO PARA O ENSINO. Disponivel em: https://www.plataformaeducacional.net/blog/2018/10/31/a-importancia-da-programacao-para-ensino/. Acessado no dia 21/04/2022.

ALMEIDA, Maria Elizabeth Bianconcini de. Inclusão digital do professor: formação e prática pedagógica. São Paulo: Editora Articulação, 2004.

MOYSÉS, H. N. Curso de física básica - vol. 1, ed. 4 – São Paulo, 2002.

CAELUN. Python e orientação a objetos. Disponível em: https://www.caelum.com.br/apostila/apostila-python-orientacao-a-objetos.pdf. Acessado no dia 23/04/2022.

TOKIO, School. Disponível em: https://tokioschool.pt/noticias/historia-python/. Acessado no dia 23/04/2022.

Data Flair. Disponível em: https://data-flair.training/blogs/python-applications/. Acessado no dia 23/04/2022.

NEWMANN, Marck. Computational physics with python. Disponível em: http://www-personal.umich.edu/~mejn/computational-physics/. Acessado no dia 25/04/2022.

SOUBHIA L. Ana. Python 101.COSTA, M. Simulações computacionais no ensino de física: Revisão sistemática de publicações da área do ensino. Disponível em: https://educere.bruc.com.br/arquivo/pdf2017/24200_12224.pdf. Acessado no dia 03/05/2022.

SOUSA, G. F. F. Simuladores computacional para o ensino de física básica: uma discussão sobre produção e uso. 12/2010. Disponível em: https://www.if.ufrj.br/~pef/producao_academica/dissertacoes/2010_Geraldo_Felipe/dissertacao_Geraldo_Felipe.pdf. Acessado no dia 05/05/2022.

RUBIN H. L, Manuel J. Páez, Cristian C. Bordeianu. Computational Physics: Problem Solving with Python. Ed. 3, illustrated, reprint. Publ. John Wiley & Sons, 2015.

JÚNIOR, Joab Silas da Silva. "Cinco dicas para resolver exercícios de Física"; *Brasil Escola*. Disponível em: https://brasilescola.uol.com.br/fisica/cinco-dicas-para-resolver-exercicios-fisica.htm. Acesso em 08 de maio de 2022.

STOODI. Física mecânica: saiba tudo sobre o movimento dos corpos. Disponível em: https://www.stoodi.com.br/blog/fisica/fisica-mecanica/. Acessado no dia 09/05/2022.

MOTISUK R. Mecanica básica- Resumo teórico. Disponível em: https://www.politecnicos.com.br/disciplinas/4323101-fisica-i-poli-usp/pdf/4323101/002.pdf. Acessado no dia 09/05/2022.

MOYSÉS. H. N. Mecânica I. Física básica. Vol. 1. Ed. 04. Ano 2002.

HALLIDAY, D.; RESNICK, R.; WALKER, J. Fundamentos de física. Mecânica. Cap. 02. Questão 17. 9.ed. Rio de Janeiro: LTC, 2012. v.l.

HALLIDAY, D.; RESNICK, R.; WALKER, J. Fundamentos de física. Gravitação, Ondas e Termodinâmica. Cap. 18. Questão 32. 9.ed. Rio de Janeiro: LTC, 2012. v.2.

HALLIDAY, D.; RESNICK, R.; WALKER, J. Fundamentos de física. Eletromagnetismo. Cap. 22. Questão 83. 9.ed. Rio de Janeiro: LTC, 2012. v.3.

HALLIDAY, D.; RESNICK, R.; WALKER, J. Fundamentos de física. Eletromagnetismo. Cap. 27. Questão 16. 9.ed. Rio de Janeiro: LTC, 2012. v.3.