Revista Naval e Oceânica, Rio de Janeiro, v.1, 2024, p. 22-36 DOI: 10.12957/rno.2024.85016

https://www.e-publicacoes.uerj.br/index.php/rno

UTILIZANDO PYTHON NO ENSINO DA CONSTRUÇÃO NAVAL: EXPERIÊNCIAS DE UM CURSO DE EXTENSÃO

Using Python in Teaching Naval Construction: Experiences from an Extension Course

Carlos Vitor de Alencar Carvalho [a]*, Marcelo Musci [b]

Recebido em: 10 jun. 2024 | Aceito em: 12 jun. 2024

RESUMO

A utilização de Python como ferramenta na Educação tem se mostrado uma abordagem inovadora e eficaz para capacitar alunos e profissionais na área. Este artigo destaca a experiência de um curso de extensão ministrado a estudantes de tecnologia da construção naval, onde foram desenvolvidos programas em Python para abordar diversos aspectos relevantes para a indústria Naval, Marítima e Portuária. No curso, foram abordados quatro exercícios práticos que demonstraram a aplicação direta da linguagem Python em problemas específicos da área. O primeiro exercício consistiu na verificação da navegabilidade de um navio em portos, utilizando como critérios o calado do navio e a profundidade do porto. Em seguida, foi realizado um cálculo de deslocamento baseado nas dimensões do casco e na densidade da água, permitindo uma análise do comportamento do navio em diferentes condições. Outro exercício envolveu a simulação de rotas de navio com base em coordenadas de partida e chegada, fornecendo uma ferramenta para planejamento de trajetórias. O último exercício envolveu o uso de biblioteca gráficas para a análise e visualização da distribuição dos tipos de cargas em um determinado porto. Esses exemplos ilustram como Python pode ser utilizado para resolver problemas interessante na indústria naval, marítima e portuária, proporcionando aos alunos uma compreensão prática dos conceitos teóricos e preparando-os para desafios do mercado de trabalho. A integração de programação em Python nos currículos educacionais nessa área pode contribuir significativamente para o desenvolvimento de profissionais qualificados e alinhados com o mercado de trabalho.

Palavras-chave: Python na Educação, Programação aplicada à Indústria Naval, Educação em Engenharia.

[a]Universidade do Estado do Rio de Janeiro (UERJ), Faculdade de Ciências Exatas e Engenharias Departamento Naval e Pesca (DEPNAPE), Rio de Janeiro– RJ, Brasil. Universidade de Vassouras, Vassouras– RJ, Brasil.

[b]Universidade do Estado do Rio de Janeiro (UERJ), Faculdade de Ciências Exatas e Engenharias Departamento Naval e Pesca (DEPNAPE), Rio de Janeiro– RJ, Brasil.

* Autor correspondente: carlos.vitor.carvalho@uerj.br

ABSTRACT

The utilization of Python as a tool within the realm of Education has emerged as an innovative and efficacious approach for empowering both students and professionals in the field. This article delineates the experiential narrative of an extension course delivered to students specializing in shipbuilding technology, wherein Python-based programs were devised to address manifold pertinent facets of the Naval, Maritime, and Port industries. The curriculum encompassed four pragmatic exercises, each serving to elucidate the direct applicability of the Python language to bespoke issues within the domain. The initial exercise entailed assessing the navigability of vessels within ports, predicated upon parameters such as vessel draft and port depth. Subsequent to this, displacement computations were conducted predicated upon hull dimensions and water density, affording an analysis of vessel behavior across diverse scenarios. Another exercise entailed the simulation of vessel routes predicated upon departure and arrival coordinates, furnishing a tool for trajectory delineation. The concluding exercise involved the utilization of graphical libraries for the analysis and visualization of cargo type distribution within a designated port. These exemplars underscore the efficacy of Python in addressing substantive challenges within the naval, maritime, and port sectors, thereby imparting to students a practical comprehension of theoretical paradigms whilst equipping them for vocational exigencies. The incorporation of Python programming into educational curricula within this sphere stands to significantly enhance the development of proficient professionals harmonized with the demands of the contemporary job market.

Keywords: Python in Education, Programming applied to Naval Industry, Engineering Education.

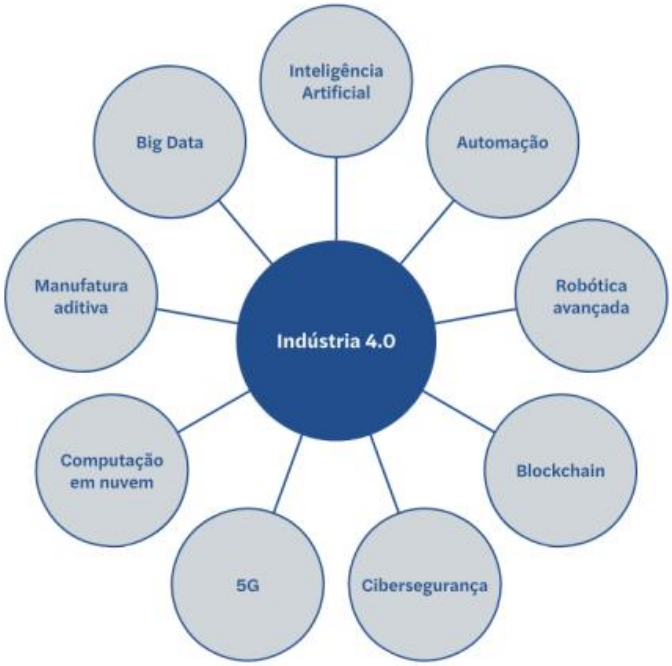
1. INTRODUÇÃO

A indústria marítima desempenha um papel fundamental na economia global, sendo responsável pelo transporte de aproximadamente 80% das mercadorias ao redor do mundo (Revista Transporte Marítimo, 2023). Nesse contexto, a educação naval desempenha um papel crucial na formação de profissionais capacitados para atuar em diversas áreas, desde a construção e manutenção de embarcações até a gestão portuária e logística marítima. A crescente complexidade das operações marítimas, aliada às demandas por eficiência e segurança, destaca a importância da educação contínua e da adoção de tecnologias na indústria naval e marítima, em destaque as tecnologias da indústria 4.0, como visto na Figura 1.

Um estudo realizado pela Mútua em 2023 (Araújo et al., 2023) evidencia claramente que a indústria 4.0, caracterizada pela integração de diversas tecnologias digitais e físicas nos processos de produção, é e continuará sendo essencial e indispensável para o ambiente de trabalho tanto presente quanto futuro. A Figura 1 ilustra as principais habilidades exigidas pela indústria 4.0. Nela, destacam-se diversas competências intimamente relacionadas ao desenvolvimento de software, incluindo a capacidade de criar programas voltados para

inteligência artificial, análise de grandes volumes de dados (big data), computação em nuvem, automação, robótica e cibersegurança.

Figura 1 – Habilidades da Indústria 4.0



Fonte: Araújo et al. (2023)

Nos últimos anos, o Python emergiu como uma das linguagens de programação de alto nível mais populares e versáteis do mundo. Sua sintaxe simples e legibilidade facilitam o aprendizado, tornando-o uma escolha ideal para iniciantes e especialistas em programação. Python está disponível de forma gratuita no site *www.python.org*. Ademais, a vasta gama de bibliotecas e frameworks disponíveis para Python o torna adequado para uma variedade de aplicações, desde análise de dados até desenvolvimento web e automação de tarefas. Alguns dos principais fundamentos da linguagem podem ser vistos no livro de Lambert (2022), no livro de Alves (2021) e no livro de Neto e Silva (2022).

O presente artigo tem como objetivo explorar o potencial da linguagem Python utilizando exemplos da indústria Naval, Marítima e Portuária e com base na experiência de um curso de extensão realizado com estudantes de tecnologia da construção naval. Serão apresentados os exercícios práticos desenvolvidos no curso para demonstrar a aplicação direta do Python em problemas específicos da indústria e academia. Ao final do artigo, serão discutidas as implicações dessas experiências e uma análise qualitativa mostrando as impressões positivas pelos participantes do curso.

2. REVISÃO DA LITERATURA

A construção naval e as atividades marítimas são áreas complexas que envolvem uma variedade de processos e operações. A programação desempenha um papel fundamental nessas áreas, permitindo a automação de tarefas, análise de dados, simulação de sistemas e desenvolvimento de soluções para problemas específicos.

Na construção naval, por exemplo, a programação é utilizada para modelagem e simulação de embarcações, análise estrutural, controle de qualidade e gestão de processos de produção. Nas atividades marítimas, a programação é empregada em sistemas de navegação, controle de tráfego marítimo, otimização de rotas e logística portuária. Portanto, o domínio da programação é essencial para profissionais que atuam nessas áreas, permitindo-lhes desenvolver soluções eficientes e inovadoras para os desafios enfrentados na indústria marítima.

Observa-se na literatura diversas iniciativas usando a linguagem Python na educação. Em Silva (2020) pode-se observar o uso da ferramenta *Google Colab* no ensino de linguagem de programação Python. O autor mostrou que por meio desse ambiente é possível a interação do estudante na prática do ensino da programação de forma simples e dedutiva. Em Marques, Costa, Silva e Rebouças (2011), os autores mostraram uma experiência interessante da introdução a programação junto à alunos do ensino médio utilizando jogos e linguagem Python.

2.1. Vantagens e Características do Python

Python tem ganhado destaque como uma linguagem de programação preferencial em uma variedade de setores, incluindo a indústria marítima. Suas vantagens incluem uma sintaxe simples e legível, o que facilita o aprendizado e a manutenção do código (Lambert, 2022). Outrossim, Python possui uma vasta biblioteca padrão e uma comunidade ativa de desenvolvedores que contribuem com uma ampla gama de pacotes e frameworks. Por exemplo, bibliotecas como *TensorFlow* e *Scikit-learn* oferecem aos desenvolvedores a capacidade de implementar modelos de Machine Learning. Isso torna Python extremamente versátil e adequado para uma variedade de aplicações, desde tarefas simples até projetos complexos. Outra vantagem do Python é sua portabilidade, pois é suportado por uma ampla gama de plataformas e sistemas operacionais. Essas características fazem do Python uma escolha ideal para o desenvolvimento de soluções na indústria naval e marítima, onde a simplicidade, eficiência e confiabilidade são essenciais.

3. METODOLOGIA

O curso de extensão abordou uma variedade de conceitos teóricos e práticos fundamentais relacionados à aplicação do Python na indústria marítima. Isso incluiu conceitos de programação básica, como variáveis, estruturas de controle, funções e classes. Assim como tópicos mais avançados, como manipulação de arquivos, tratamento de exceções, uso de bibliotecas específicas e desenvolvimento de aplicações práticas.

Os participantes do curso foram introduzidos a bibliotecas Python relevantes para a indústria marítima, como *numpy* para cálculos numéricos, *matplotlib* para visualização de dados e pandas para manipulação de dados estruturados. Ainda foram exploradas técnicas de resolução de problemas comuns na indústria marítima, incluindo análise de rotas, cálculos de deslocamento e simulação de sistemas de navegação.

A metodologia de ensino adotada no curso priorizou uma abordagem prática e orientada para projetos, visando proporcionar aos participantes uma experiência de aprendizado hands-on* e contextualizada com a indústria marítima. Foram utilizadas aulas expositivas para apresentar os conceitos fundamentais de programação Python, combinadas com atividades práticas e projetos orientados para resolver problemas específicos da indústria naval.

Durante o curso, os participantes foram desafiados a desenvolver programas em Python para abordar problemas práticos e relevantes para a indústria naval. Isso incluiu o desenvolvimento de algoritmos para cálculos de deslocamento de embarcações, simulação de rotas de navegação, análise e visualização de dados de processos portuários. Os programas desenvolvidos foram baseados em conceitos teóricos aprendidos durante as aulas e adaptados para atender às necessidades específicas da indústria naval e marítima.

4. EXPERIÊNCIA DO CURSO DE EXTENSÃO

4.1. Perfil dos Participantes

O curso de extensão foi concebido como uma iniciativa para capacitar estudantes de tecnologia da construção naval, dos demais cursos de engenharia da instituição interessados em adquirir habilidades em programação Python aplicada à indústria naval e profissionais da área. Os participantes do curso variavam desde estudantes universitários até profissionais já inseridos no mercado de trabalho, todos com interesse em aprimorar seus conhecimentos e competências na área de computação. O curso foi estruturado para atender tanto iniciantes em programação quanto aqueles com algum conhecimento prévio, garantindo uma experiência inclusiva e acessível a todos os interessados.

*Hands on: termo originário da língua inglesa que significa se envolver diretamente na execução de uma atividade.

4.2. Apresentação dos quatro Exercícios Práticos Realizados Durante o Curso

Durante o curso, foram propostos quatro exercícios práticos que serviram como casos de estudo para aplicar os conceitos aprendidos em situações reais da indústria naval. O primeiro exercício, mais simples, envolveu o cálculo de deslocamento de uma embarcação com base nas dimensões do casco e na densidade da água, permitindo uma análise do comportamento do navio em diferentes condições. No Quadro 1, pode-se verificar o código desenvolvido para esse problema.

Quadro 1 - Exercício sobre o cálculo do deslocamento de uma embarcação

```
# Função para calcular o deslocamento
def calcular_deslocamento(comprimento, boca, calado, densidade_agua):
    # Converter a densidade da água de t/m³ para kg/m³
    densidade_agua_kg_m3 = densidade_agua * 1000

    # Calcular o volume do casco abaixo da linha d'água em metros cúbicos
    volume_deslocado = comprimento * boca * calado

    # Calcular o deslocamento em toneladas (t)
    deslocamento_toneladas = volume_deslocado * densidade_agua_kg_m3 / 1000

    return deslocamento_toneladas

# Dados de entrada do usuário
comprimento = 397 #metros
boca = 56 #metros
calado = 15.5 #metros
densidade_agua_ton_m3 = 1.025 #ton/m3

# Calcular o deslocamento e exibir o resultado
resultado = calcular_deslocamento(comprimento, boca, calado,
densidade_agua_ton_m3)
print(f"O deslocamento do navio é {resultado:.2f} toneladas.")
```

Fonte: O autor

Neste código Python, uma função chamada *calcular_deslocamento* é definida para calcular o deslocamento de um navio com base no comprimento, boca, calado e densidade da água.

No início do código há definição da função *calcular_deslocamento*. Dentro dela, a densidade da água é convertida de toneladas por metro cúbico para quilogramas por metro cúbico, pois a fórmula requer a densidade da água em kg/m³. Em seguida, o volume do casco abaixo da linha d'água é calculado multiplicando o comprimento, a boca e o calado. Por fim, o

deslocamento é calculado em toneladas, utilizando a fórmula do volume deslocado multiplicado pela densidade da água e o resultado é retornado.

Em seguida, são fornecidos os dados de entrada do usuário: comprimento, boca, calado e densidade da água. Depois disso, a função *calcular_deslocamento* é chamada com os dados fornecidos e o resultado é armazenado na variável resultado. Por fim, o resultado é exibido, apresentando o deslocamento do navio em toneladas.

O segundo exercício apresenta um sistema para avaliar a adequação de navios em diferentes portos, levando em consideração o calado do navio e a profundidade do porto. No Quadro 2, pode-se verificar o código desenvolvido.

Inicialmente é definido um dicionário chamado portos, que mapeia o nome de diferentes portos no Brasil para suas respectivas profundidades em metros.

Em seguida, é declarada uma função chamada *verificar_porto_adaptado*. Essa função recebe dois parâmetros: o calado do navio e a profundidade do porto. Ela retorna o termo *True* se o calado do navio for menor ou igual à profundidade do porto, indicando que o navio pode navegar com segurança no porto e o termo *False* caso contrário.

Finalmente é definida outra função chamada *listar_portos_adaptados*.

Esta função recebe o calado do navio como entrada e itera sobre o dicionário de portos. Para cada porto, chama a função *verificar_porto_adaptado* para determinar se o porto é adequado para o calado do navio. Se for adequado, adiciona o nome do porto a uma lista chamada *portos_adaptados*. Finalmente, retorna a lista de portos adequados.

Depois disso, o código solicita ao usuário que insira o calado do navio. Em seguida, chama a função *listar_portos_adaptados* com o calado do navio fornecido para obter uma lista de portos adequados. Por fim, o código exibe os portos adequados, se houver algum, ou uma mensagem indicando que nenhum porto é adequado para o calado do navio.

O terceiro exercício consistiu na simulação de rotas de navegação com base em coordenadas de partida e chegada. No código foi realizada simulação de rota de navio entre dois pontos geográficos (Nova York; Nova York e Agadir; Marrocos). No Quadro 3, pode-se verificar o código desenvolvido.

Quadro 2: Verificação da navegabilidade de um navio em portos

```
# Dicionário de portos e suas profundidades em metros
portos = {
    "Itaguaí (RJ)": 20,
    "Itapoá (SC)": 14,
    "Paranaguá (PR)": 16,
    "Rio de Janeiro (RJ)": 14.5,
    "Rio Grande (RS)": 14,
    "Salvador (BA)": 15,
    "Santos (SP)": 15,
    "Suape (PE)": 14.5,
}

# Função para verificar se um porto é adequado com base no calado do navio
def verificar_porto_adaptado(calado_navio, profundidade_porto):
    if calado_navio <= profundidade_porto:
        return True
    else:
        return False

# Função para listar os portos adequados com base no calado do navio
def listar_portos_adaptados(calado_navio):
    portos_adaptados = []
    for porto, profundidade in portos.items():
        if verificar_porto_adaptado(calado_navio, profundidade):
            portos_adaptados.append(porto)
    return portos_adaptados

# Solicitar o calado do navio ao usuário
calado_navio = float(input("Informe o calado do navio em metros: "))

# Chamar a função para listar os portos adequados
portos_adaptados = listar_portos_adaptados(calado_navio)

# Exibir os portos adequados
if portos_adaptados:
    print("O navio pode navegar com segurança nos seguintes portos:")
    for porto in portos_adaptados:
        print(porto)
else:
    print("O navio não pode navegar com segurança em nenhum dos portos listados.")
```

Fonte: O autor

Quadro 3 - Simulação de rotas de navegação

```
import math
import folium
# Função para calcular a distância entre dois pontos de coordenadas geográficas
def calcular_distancia(lat1, lon1, lat2, lon2):
    # Converter de graus para radianos
    lat1 = math.radians(lat1)
    lon1 = math.radians(lon1)
    lat2 = math.radians(lat2)
    lon2 = math.radians(lon2)

    # Raio médio da Terra em quilômetros
    raio_terra = 6371.0

    # Diferenças entre as coordenadas
    dlat = lat2 - lat1
    dlon = lon2 - lon1

    # Fórmula de Haversine
    a = math.sin(dlat/2)**2 + math.cos(lat1) * math.cos(lat2) * math.sin(dlon/2)**2
    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1-a))
    distancia = raio_terra * c

    return distancia

# Função para plotar a rota no mapa usando Folium
def plotar_rota(lat_partida, lon_partida, lat_chegada, lon_chegada):
    # Inicializar o mapa com as coordenadas do ponto de partida
    mapa = folium.Map(location=[lat_partida, lon_partida], zoom_start=5)

    # Adicionar marcador de partida
    folium.Marker([lat_partida, lon_partida], tooltip='Partida', popup='Porto de Nova
York ').add_to(mapa)

    # Adicionar marcador de chegada
    folium.Marker([lat_chegada, lon_chegada], tooltip='Chegada', popup='Porto de
Agadir').add_to(mapa)

    # Calcular a distância entre os pontos
    distancia = calcular_distancia(lat_partida, lon_partida, lat_chegada, lon_chegada)

    # Adicionar uma linha de rota entre os pontos de partida e chegada
    folium.PolyLine([(lat_partida, lon_partida), (lat_chegada, lon_chegada)],
color='red', weight=2.5, opacity=1).add_to(mapa)

    # Exibir a distância no mapa
   folium.Marker([(lat_partida + lat_chegada) / 2, (lon_partida + lon_chegada) / 2],
tooltip=f'Distância: {distancia:.2f} km').add_to(mapa)

    # Salvar o mapa em um arquivo HTML
```

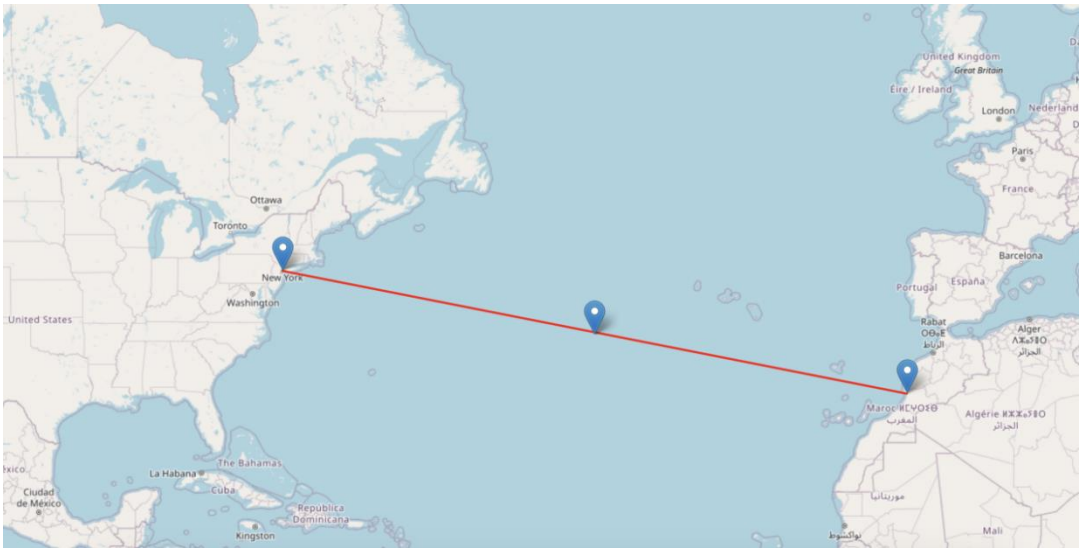
```
    mapa.save("rota_navio_curso_seanav.html")

# Função principal do programa
def main():
    print("Simulação de Rota de Navio")

    # Coordenadas dos pontos de partida (Nova York, NY) e chegada (Agadir,
Marrocos)
    lat_partida = 40.7128
    lon_partida = -74.0060
    lat_chegada = 30.4220
    lon_chegada = -9.5982

    # Calcular a distância entre os pontos
    distancia = calcular_distancia(lat_partida, lon_partida, lat_chegada, lon_chegada)

    # Exibir o resultado
    print(f"A distância entre os pontos é de {distancia:.2f} quilômetros.")

    # Plotar a rota no mapa usando Folium
    plotar_rota(lat_partida, lon_partida, lat_chegada, lon_chegada)
    print("A rota foi plotada no mapa. Você pode verificar o mapa em
'rota_naval.html'.")
if __name__ == "__main__":
    main()
```

Fonte: O autor

Neste código, foram utilizadas as bibliotecas *math* para cálculos trigonométricos e as bibliotecas *folium* para visualização de mapas. Inicialmente é definida a função *calcular_distancia*, que recebe as coordenadas geográficas de dois pontos e utiliza a fórmula de *Haversine* para calcular a distância entre eles, considerando a curvatura da Terra.

Em seguida é apresentada a função *plotar_rota*, que cria um mapa usando *Folium*. Esta função recebe as coordenadas de partida e chegada do navio, adiciona marcadores para indicar os pontos no mapa, traça uma linha de rota entre eles e exibe a distância entre os pontos no mapa. O resultado é apresentado em um arquivo html nomeado de *rota_navio_curso_seanav.html*. O resultado de todo o processo após abrir o arquivo html, pode ser visto na Figura 2.

Figura 2 – Mapa gerado indicando a rota entre os dois portos.



Fonte: O autor

O quarto exercício teve o objetivo de criar um gráfico de barras que representa a distribuição de diferentes tipos de cargas por quantidade (em toneladas) em um determinado porto. Para isso foi utilizada a biblioteca chamada *matplotlib*. O código está disponível no Quadro 4.

Quadro 4: Verificação da navegabilidade de um navio em portos

```
# Dicionário de portos e suas profundidades em metros
portos = {
    "Itaguaí (RJ)": 20,
    "Itapoá (SC)": 14,
    "Paranaguá (PR)": 16,
    "Rio de Janeiro (RJ)": 14.5,
    "Rio Grande (RS)": 14,
    "Salvador (BA)": 15,
    "Santos (SP)": 15,
    "Suape (PE)": 14.5,
}

# Função para verificar se um porto é adequado com base no calado do navio
def verificar_porto_adaptado(calado_navio, profundidade_porto):
    if calado_navio <= profundidade_porto:
        return True
    else:
        return False

import matplotlib.pyplot as plt

# Dicionário de cargas por tipo (em toneladas)
cargas = {
    "Contêineres": 15000,
    "Minério de Ferro": 5000,
```

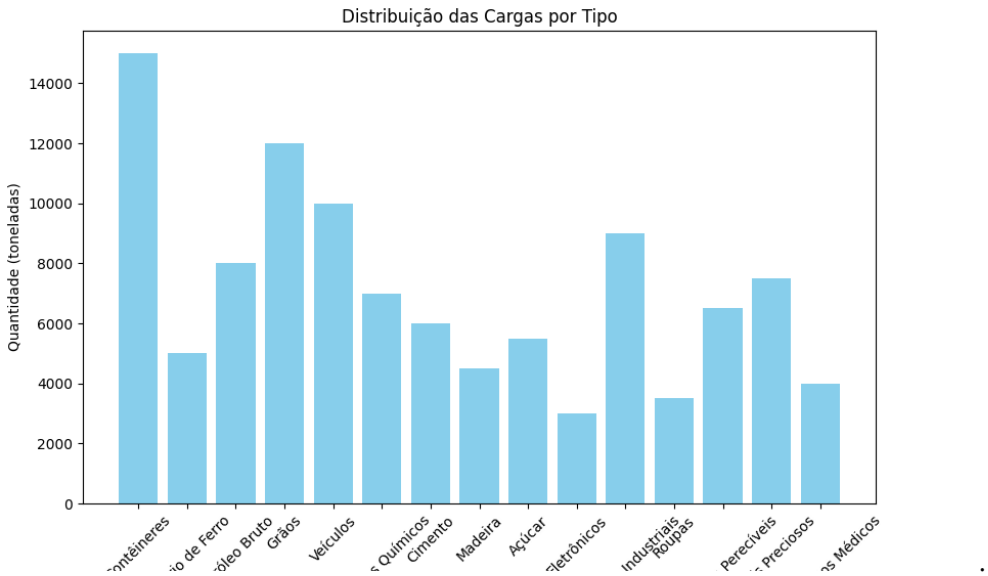
```
    "Petróleo Bruto": 8000,
    "Grãos": 12000,
    "Veículos": 10000,
    "Produtos Químicos": 7000,
    "Cimento": 6000,
    "Madeira": 4500,
    "Açúcar": 5500,
    "Eletrônicos": 3000,
    "Máquinas Industriais": 9000,
    "Roupas": 3500,
    "Alimentos Perecíveis": 6500,
    "Metais Preciosos": 7500,
    "Equipamentos Médicos": 4000,
}

# Função para calcular o total de cargas
# print(f"Key {carga} has value {quantidade}")
def calcular_total_cargas(cargas):
    total = 0
    for carga, quantidade in cargas.items():

        total += quantidade
    return total

# Função para criar um gráfico de barras das cargas
def criar_grafico_barras(cargas):
    tipos_cargas = list(cargas.keys())
    quantidades = list(cargas.values())

    plt.figure(figsize=(10, 6))
    plt.bar(tipos_cargas, quantidades, color='skyblue')
    plt.xlabel('Tipos de Cargas')
    plt.ylabel('Quantidade (toneladas)')
    plt.title('Distribuição das Cargas por Tipo')
    plt.xticks(rotation=45)
    plt.show()

# Exibir os tipos de cargas e suas quantidades
print("Tipos de Cargas e Quantidades:")
for carga, quantidade in cargas.items():
    print(f"{carga}: {quantidade} toneladas")

# Calcular o total de cargas
total_cargas = calcular_total_cargas(cargas)
# Exibir o total de cargas
print(f"\nTotal de Cargas: {total_cargas} toneladas")
# Criar um gráfico de barras das cargas
criar_grafico_barras(cargas)
```

Fonte: o autor

Inicialmente, um dicionário cargas é definido, contendo os tipos de carga como chaves e as quantidades como valores. Em seguida, duas funções são definidas: *calcular_total_cargas* para somar todas as quantidades de carga e *criar_grafico_barras* para plotar o gráfico de barras usando os tipos de carga no eixo x e as quantidades no eixo y. O código também exibe os tipos de cargas e suas quantidades. Por fim, o total de todas as cargas é calculado e exibido, seguido pela criação e exibição do gráfico de barras.

Resumindo, o código realiza de forma rápida uma análise e visualização da distribuição de cargas, fornecendo uma visão clara da quantidade de cada tipo de carga em relação as demais. A Figura 3 mostra o gráfico gerado.

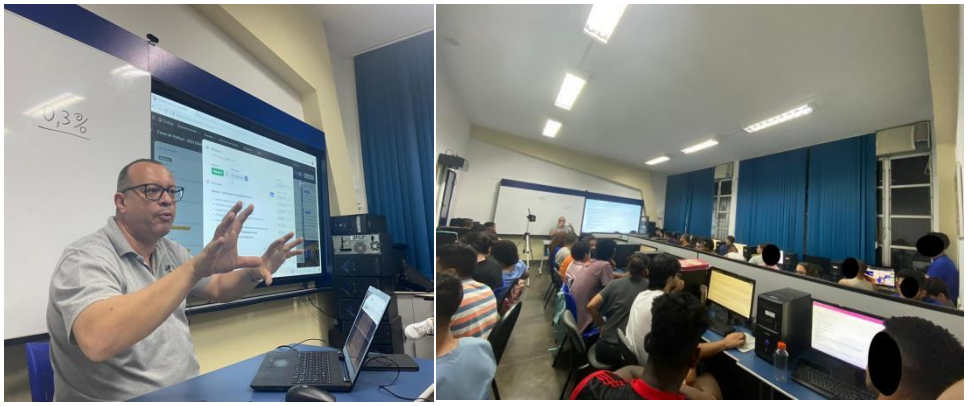
Figura 3 – Gráfico gerado com as informações dos tipos de cargas.



Fonte: O autor

Esses exercícios práticos proporcionaram aos participantes uma oportunidade de aplicar seus conhecimentos de programação Python na resolução de problemas reais da indústria naval, preparando-os para os desafios possíveis em outras disciplinas do curso, tal como o mercado de trabalho. Na Figura 4 é apresentado alguns momentos de apresentação, desenvolvimento e discussão no curso de extensão realizado.

Figura 4 – Registros da realização do curso de extensão.



Fonte: O autor

5. DISCUSSÕES E CONSIDERAÇÕES FINAIS

O sucesso do curso de extensão e a aplicação bem-sucedida de Python na indústria naval e atividades marítimas indicam um vasto potencial de expansão do uso dessa linguagem de programação nesses setores. À medida que a tecnologia continua a desempenhar um papel cada vez mais importante na indústria marítima/naval/portuária, espera-se que a linguagem Python seja cada vez mais adotada para uma variedade de aplicações. Sendo elas: automações de processos de portos, análise de dados para tomada de decisões e otimização das operações portuárias, previsão de demandas da área usando Machine Learning otimização de rotas de navegação, análise de big data, monitoramento ambiental e muito mais.

As considerações finais do curso, em uma análise qualitativa por parte dos participantes do curso, são extremamente positivas, refletindo a satisfação e o entusiasmo dos alunos em relação ao aprendizado proporcionado. Os participantes expressaram não apenas um interesse genuíno, mas também um desejo de continuar a se desenvolver na área de programação com Python. Suas opiniões reforçam a importância de iniciativas como essa, que proporcionam uma experiência prática e relevante para a formação acadêmica e profissional. Os alunos manifestaram um claro desejo por mais oportunidades de aprendizado semelhantes, demonstrando um comprometimento em expandir seus conhecimentos e habilidades na utilização do Python para resolver desafios na indústria naval e marítima. Além disso, pensando em proporcionar estudos posteriores e novos desenvolvimentos, os códigos desenvolvidos durante o curso foram disponibilizados no repositório do GitHub: *https://github.com/cvitorc/CursoPython-Seanav-2023.*

6. REFERÊNCIAS BIBLIOGRÁFICAS

Alves, W. P. (2021). *Programação Python: aprenda de forma rápida* (E-book). Editora Saraiva. ISBN 9786558110149. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9786558110149/. Acesso em: 21 abr. 2024.

Araújo, B. C. P. O. de, Lima, V. R. de Linguitte, M. A., & Pieracciani, V. (Coord.). (2023). *O Futuro das engenharias no Brasil* [livro eletrônico]. Barueri, SP: Pieracciani Desenvolvimento de Empresas; Brasília, DF: Mútua - Caixa de Assistência dos Profissionais do Crea. Disponível em: https://www.mutua.com.br/wp-content/uploads/2023/08/futuro_engenharias-compactado.pdf. Acessado em: 18 de abr. 2024.

Lambert, K. A. (2022). *Fundamentos de Python: primeiros programas* (E-book). Cengage Learning Brasil. ISBN 9786555584301. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9786555584301/. Acesso em: 21 abr. 2024.

Marques, D. L., Costa, L. F. S., Silva, M. A. de A., & Rebouças, A. D. D. S. (2011). *Atraindo Alunos do Ensino Médio para a Computação: Uma Experiência Prática de Introdução a Programação utilizando Jogos e Python*. In: WORKSHOP DE INFORMÁTICA NA ESCOLA (WIE), 17., 2011, Aracajú. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação (pp. 1138-1147). https://doi.org/10.5753/wie.2011.21724.

Neto, R. F. T., & Silva, F. M. da. (2022). *Introdução à Programação para Engenharia: Usando a Linguagem Python* (E-book). Grupo GEN. ISBN 9788521638346. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788521638346/. Acesso em: 21 abr. 2024.

Revista Transporte Marítimo. (2023). Disponível em: https://unctad.org/system/files/official-document/rmt2023_en.pdf. Acessado em: 18 de abril de 2024.

Silva, M. D. (2020). *Aplicação da Ferramenta Google Colaboratory para o Ensino da Linguagem Python.* In: Escola Regional de Engenharia de Software, 2020, Brasil. *Anais da IV Escola Regional de Engenharia de Software (ERES 2020)* (v. I, pp. 67-76).