

# Aula 1 – Conceitos Básicos e IDE de Programação

## Objetivo da Aula

Compreender o conceito de paradigma e os principais paradigmas de Programação. Conhecer a linguagem de programação Python, as suas características e os seus comandos básicos. Conhecer os ambientes de desenvolvimento.

## Apresentação

O desenvolvimento de aplicações para áreas como jogos digitais, inteligência artificial e gestão de negócios só são possíveis pelo uso de linguagens de programação. Por isso, conhecer e utilizar programação tem sido cada vez mais importante.

Para programar, é preciso se familiarizar com as abordagens utilizadas para resolver problemas usando programação. Também é necessário conhecer uma linguagem de programação e suas características.

A linguagem de programação Python, criada, em 1991, por Guido van Rossum, destaca-se por ser fácil de aprender e possuir recursos que fazem dela uma linguagem muito utilizada para o desenvolvimento de soluções em diferentes áreas.

Vamos conhecer algumas características básicas da linguagem Python e alguns dos ambientes que podemos utilizar para desenvolver programas com essa linguagem.

## 1. Linguagem de Programação e seus Paradigmas

Olá, seja bem-vindo ao mundo da Programação Orientada a Objetos! Mas, antes de qualquer coisa, vamos falar um pouco sobre conceitos importantes relacionados à linguagem de programação e aos paradigmas de programação.

Linguagem de programação é uma forma de se comunicar com o computador, permitindo que sejam escritas instruções que o computador possa entender e executar. Elas são usadas para desenvolver aplicativos, jogos e muitas outras coisas. Existem muitas linguagens de programação diferentes, cada uma com suas próprias características e finalidades.

Os paradigmas da programação se referem a diferentes abordagens para a solução de problemas de programação. Há vários paradigmas diferentes, e cada um tem suas próprias características e maneiras de pensar sobre a programação.

Os principais paradigmas da programação são:

- Programação Procedural: é um paradigma que se concentra em dividir um programa em pequenos blocos de código que realizam tarefas específicas. O código é organizado em funções e procedimentos, que podem ser chamados e executados quando necessário;
- Programação Orientada a Objetos (POO): é um paradigma que se concentra na criação de objetos que contêm dados e métodos, os quais podem ser usados para manipular esses dados. Os objetos são usados para representar as entidades do mundo real e as interações entre elas;
- Programação Funcional: é um paradigma que se concentra no uso de funções para resolver problemas. As funções são tratadas como valores, podendo ser passadas como argumentos e retornadas como resultado de outras funções;
- Programação Declarativa: é um paradigma que se concentra em descrever o que o programa deve fazer, em vez de como deve fazer. Em vez de escrever passo a passo o que o programa deve fazer, o programador descreve o resultado desejado;
- Programação Reativa: é um paradigma que se concentra na programação de sistemas que reagem a mudanças no ambiente em tempo real. Ele usa fluxos de dados para representar eventos e reage a eles conforme necessário.

Cada paradigma de programação tem suas próprias vantagens e desvantagens, e o programador pode escolher o paradigma mais adequado para a tarefa em questão. Muitas vezes, é possível usar vários paradigmas em um único programa.

Nosso objetivo será conhecer e utilizar a Programação Orientada a Objetos, e, para isso, vamos utilizar a linguagem de programação Python.

## 2. A Linguagem de Programação Python: Características

Python é uma linguagem de programação de alto nível, interpretada, dinamicamente tipada e orientada a objetos. Isso permite que ela ofereça recursos avançados de abstração e expressividade, tornando mais fácil para os programadores escreverem códigos mais complexos com menos esforço.

O termo “alto nível” significa que a linguagem de programação é projetada para ser mais próxima da linguagem humana do que da linguagem de máquina. Isso significa que a

programação, em uma linguagem de alto nível, é geralmente mais fácil e menos propensa a erros do que a programação em uma linguagem de baixo nível.

A expressão “interpretada” significa que a linguagem é executada por um programa de software chamado “interpretador”, em vez de ser compilada em código de máquina executável antes da execução. Isso quer dizer que o código-fonte é executado diretamente pelo interpretador, tornando o processo de desenvolvimento e depuração mais fácil e rápido, mas, em contrapartida, pode tornar a execução mais lenta.

Por fim, a terminologia “dinamicamente tipada” significa que as variáveis, em uma linguagem de programação, não precisam ser declaradas com um tipo específico e podem mudar de tipo durante a execução do programa. Isso permite que os programadores escrevam um código mais flexível e reduz a quantidade de códigos que precisa ser escrita. Porém, esse tipo de linguagem pode ser mais propenso a erros durante a execução, já que problemas de tipagem só serão detectados em tempo de execução.

A linguagem Python foi desenvolvida no final da década de 1980 por Guido van Rossum, na Holanda, e recebeu esse nome em homenagem ao grupo britânico de comédia Monty Python.

Inicialmente, Python foi criada como uma linguagem de scripting, para automatizar tarefas de sistema e desenvolvimento web. Com o tempo, ela foi evoluindo, e hoje é usada em diversas áreas, como ciência de dados, inteligência artificial, desenvolvimento web, jogos, automação de tarefas, entre outras.

Uma das principais características de Python é a sua simplicidade e legibilidade. O código escrito em Python é fácil de ser lido e compreendido, o que torna a linguagem ideal para iniciantes. Além disso, Python possui uma vasta biblioteca-padrão e uma grande comunidade de desenvolvedores, o que faz com que seja possível desenvolver projetos complexos de forma rápida e eficiente.

Python também é uma linguagem multiplataforma, ou seja, ela pode ser executada em diversos sistemas operacionais, como Windows, Linux e MacOS. Além disso, a linguagem é gratuita e de código aberto, o que significa que qualquer pessoa pode contribuir para o seu desenvolvimento.

### **3. A Linguagem de Programação Python: Comandos Básicos**

É possível programar em Python utilizando um ambiente on-line. Existem diversas opções de ambientes integrados de desenvolvimento (IDEs) on-line que permitem que os desenvolvedores escrevam, executem e compartilhem código Python diretamente no navegador web, sem a necessidade de instalar um software em sua máquina local.

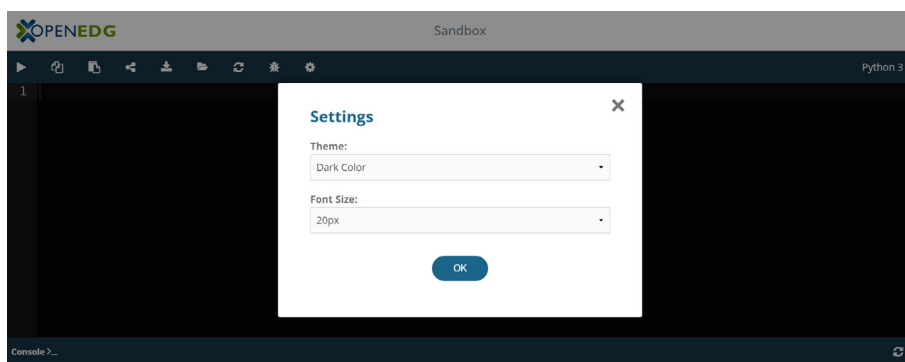
Alguns exemplos de ambientes on-line para programar em Python incluem:

- Edube Sandbox: plataforma que permite programar com diversas linguagens de programação, incluindo Python. É um ambiente bem simples que permite escrever, testar, baixar e compartilhar programas;
- Repl.it: plataforma que oferece suporte a diversas linguagens de programação. É possível escrever e executar código Python diretamente no navegador, além de compartilhar projetos com outras pessoas.
- Jupyter Notebook: é uma plataforma web para análise de dados interativa, que permite a criação de notebooks com código Python, gráficos e texto explicativo. É possível executar o Jupyter Notebook localmente ou em nuvem, por meio de serviços como o Google Colab e o Microsoft Azure;
- PythonAnywhere: plataforma que oferece um ambiente completo de desenvolvimento para Python, incluindo editor de código, terminal, banco de dados e servidor web. É possível usar o PythonAnywhere gratuitamente, mas com algumas limitações.

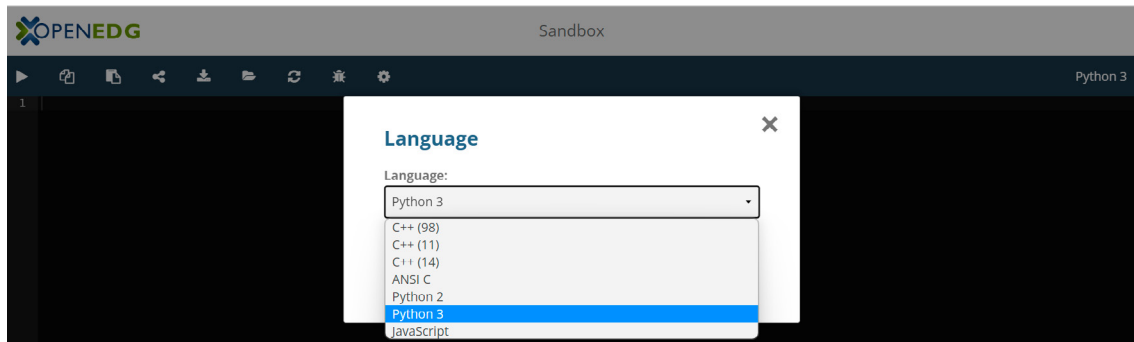
Esses ambientes on-line são especialmente úteis para quem está começando a programar em Python, pois permitem que os desenvolvedores experimentem e testem o código, sem a necessidade de instalar um ambiente de desenvolvimento em sua máquina local. Além disso, eles também são úteis para quem precisa trabalhar em projetos de forma colaborativa ou para quem precisa acessar o código de qualquer lugar, sem a necessidade de transferir arquivos.

Vamos praticar um pouco usando uma dessas alternativas! É hora de começar a escrever um código Python real e funcional. Vai ser muito simples, por enquanto. Como vamos mostrar alguns conceitos e termos fundamentais, usaremos esse trecho de código como ponto de partida.

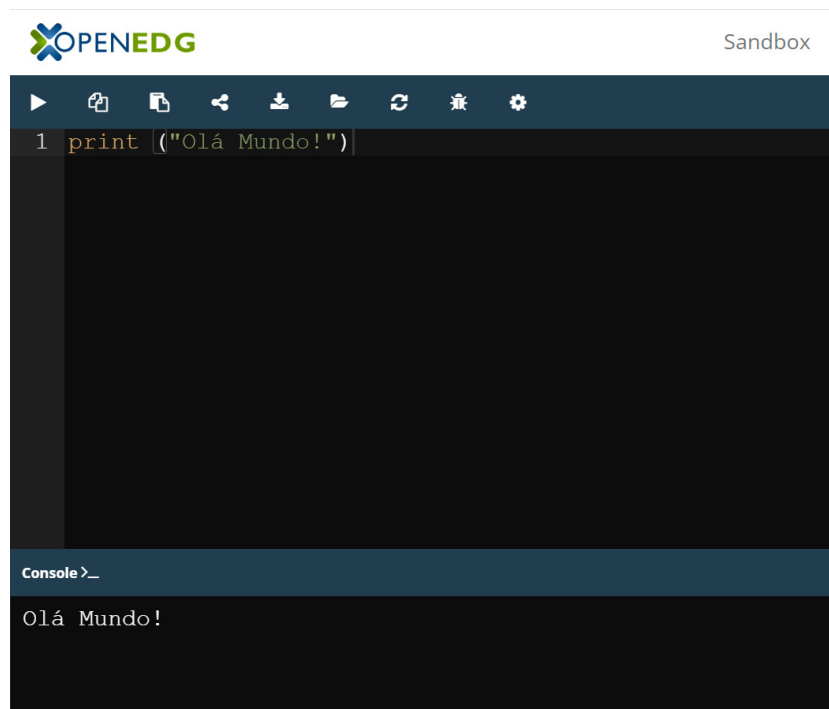
Inicie o Edube Sandbox, abrindo o navegador e digitando o endereço <https://edube.org/sandbox>. Em seguida, acesse o último botão (em forma de engrenagem) para escolher o tamanho de fonte e o tema de sua preferência.



Na parte superior à direita, vai aparecer uma linguagem de programação já definida. É possível escolher uma linguagem de programação para trabalhar, clicando no nome da linguagem inicialmente definida para fazer a troca. Vamos utilizar o Python 3.X (em que X pode ser a última numeração disponível).



Na linha 1, digite o seguinte comando: `print ("Olá Mundo!")`. Em seguida, é só acessar a primeira opção (executar) e, se tudo estiver de acordo, o resultado irá aparecer na área de console.

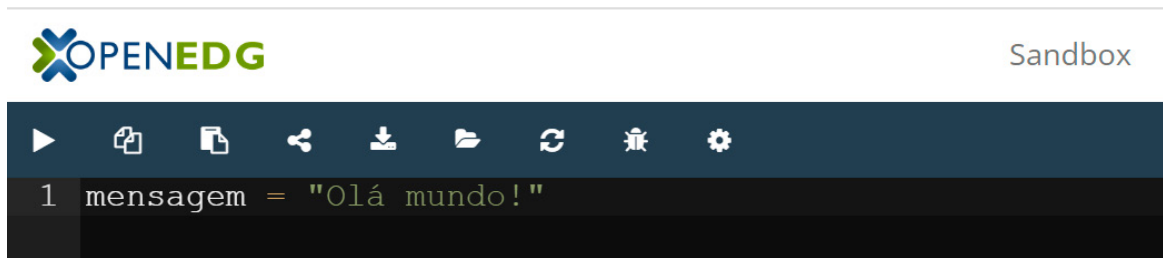


Esse pequeno exemplo apresenta alguns pontos importantes. A palavra “print” é o nome de uma função do Python capaz de exibir um resultado que pode ser visualizado pelo usuário. Para que a função funcionasse de forma adequada, foi necessário digitar o que se desejava exibir (a frase) e alguns requisitos, como o uso de aspas e parênteses.

Agora, vamos considerar alguns conceitos sobre como programar em Python, usando variáveis, operadores e comandos de entrada e saída.

### 3.1. Uso de Variáveis

Uma variável é uma forma de armazenar um valor em um programa Python. Você pode atribuir um valor a uma variável usando o sinal de igual (=). Por exemplo, para atribuir o valor "Olá Mundo!" à variável mensagem, você pode escrever:

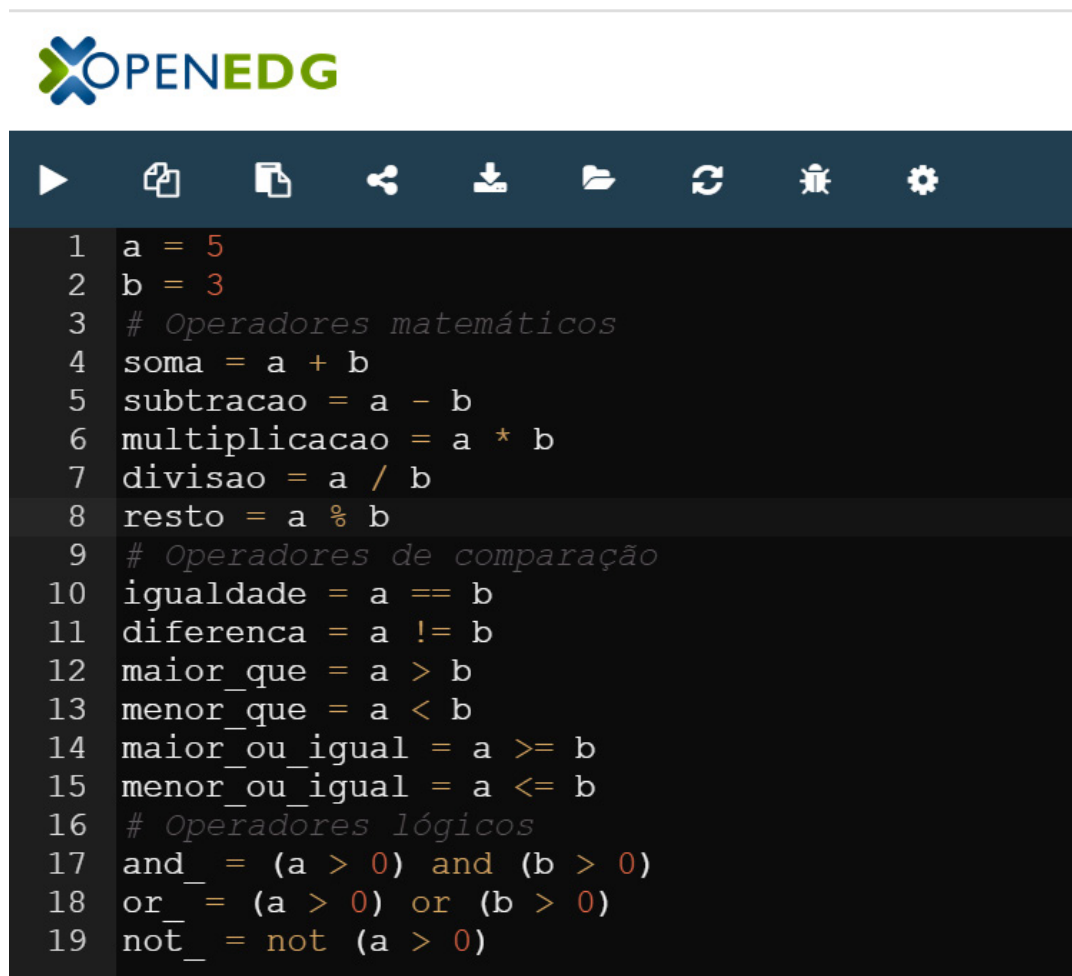


The screenshot shows the OpenEDG Python Sandbox interface. At the top, there is a toolbar with icons for running, copying, saving, sharing, downloading, opening, undo, redo, and settings. Below the toolbar, the code editor contains the following Python code:

```
1 mensagem = "Olá mundo!"
```

### 3.2. Uso de Operadores

Python suporta uma variedade de operadores que você pode usar para realizar operações em variáveis e outros valores. Por exemplo, você pode usar o operador de adição (+) para somar dois valores. Veja alguns exemplos:



The screenshot shows the OpenEDG Python Sandbox interface with a toolbar and a code editor. The code editor contains the following Python code demonstrating various operators:

```
1 a = 5
2 b = 3
3 # Operadores matemáticos
4 soma = a + b
5 subtracao = a - b
6 multiplicacao = a * b
7 divisao = a / b
8 resto = a % b
9 # Operadores de comparação
10 igualdade = a == b
11 diferenca = a != b
12 maior_que = a > b
13 menor_que = a < b
14 maior_ou_igual = a >= b
15 menor_ou_igual = a <= b
16 # Operadores lógicos
17 and_ = (a > 0) and (b > 0)
18 or_ = (a > 0) or (b > 0)
19 not_ = not (a > 0)
```

Uma observação importante. As linhas 3, 9 e 16 iniciam com o símbolo da cerquilha. No Python, esse símbolo representa uma linha de comentário e serve para orientar quem estiver lendo o código sobre aquele trecho. Um comentário pode aparecer no início da linha ou após um espaço em branco ou código.

### 3.3. Entrada e Saída de Dados

Para obter dados do usuário, você pode usar a função `input()`. A função `input()` permite que o usuário insira um valor e o armazenará em uma variável. Aqui está um exemplo:



Sandbox

```

1 nome = input("Qual é o seu nome? ")
2 print("Olá,", nome)

```

O nome digitado pelo usuário será armazenado como uma string (cadeia de caracteres), mesmo que o usuário digite números. Em alguns casos, se a entrada não for avaliada corretamente, pode acontecer um erro na execução do programa, por isso é importante realizar conversões de tipo de forma adequada antes de usar os valores capturados. Veja o exemplo a seguir:



Sandbox

```

1 produto = input("Digite o nome do produto:")
2 quantidade = int(input("Digite a quantidade:"))
3 valor_unitário = float(input("Digite o valor unitário:"))
4 total = quantidade * valor_unitário
5 print("Total a pagar:")
6 print(total)

```

**Console >\_**

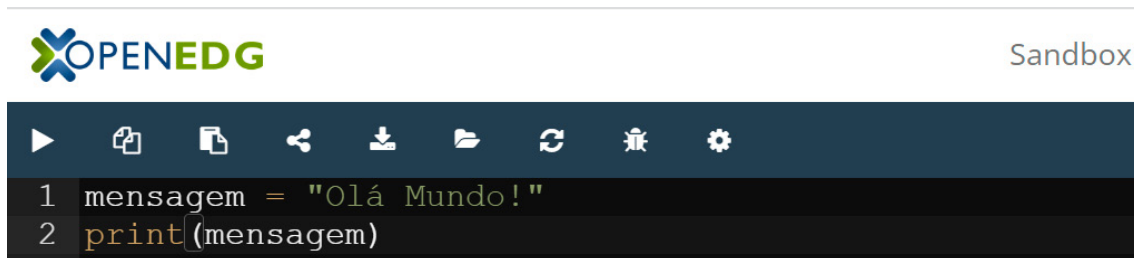
```

Digite o nome do produto:caneta
Digite a quantidade:10
Digite o valor unitário:2.5
Total a pagar:
25.0

```

No exemplo, o usuário irá digitar o nome de um produto, a sua quantidade e o seu valor unitário. Para armazenar corretamente a quantidade, foi utilizada a função `int()` para converter o valor digitado em um número inteiro. No caso do valor unitário, a função `float()` foi utilizada, visto que o valor digitado pode ser um número que contenha casas decimais.

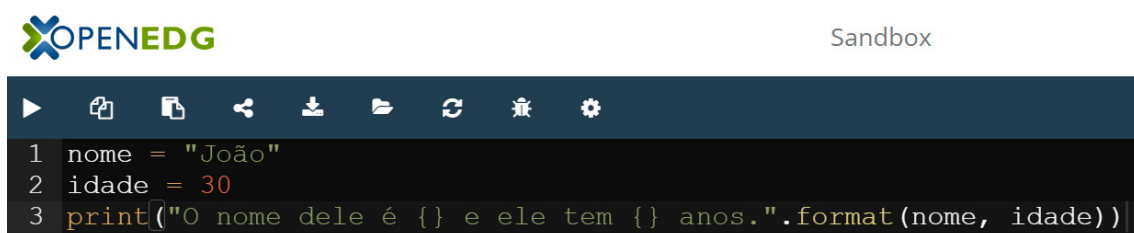
Para imprimir dados na tela, você pode usar a função `print()`, que pode imprimir texto e valores de variáveis na tela. Por exemplo:



The screenshot shows the OpenEDG Sandbox interface. At the top, there is a logo for 'XOPENEDG' and the word 'Sandbox' on the right. Below the header is a toolbar with icons for running, copying, saving, sharing, downloading, opening, refreshing, and settings. The main area contains two lines of Python code:

```
1 mensagem = "Olá Mundo!"
2 print(mensagem)
```

A linguagem Python permite formatação de saída, a qual permite que você especifique como deseja que seus dados sejam formatados antes de serem impressos na tela. Isso é especialmente útil quando você deseja imprimir valores de variáveis com um formato específico. Python suporta a formatação de saída de várias maneiras, mas aqui está um exemplo básico:

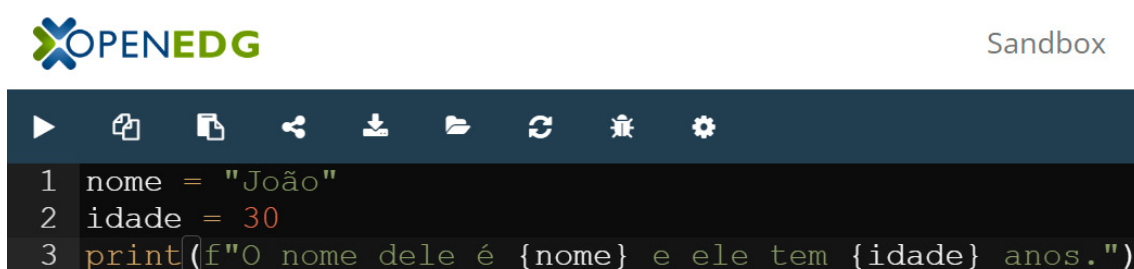


The screenshot shows the OpenEDG Sandbox interface. At the top, there is a logo for 'XOPENEDG' and the word 'Sandbox' on the right. Below the header is a toolbar with icons for running, copying, saving, sharing, downloading, opening, refreshing, and settings. The main area contains three lines of Python code:

```
1 nome = "João"
2 idade = 30
3 print("O nome dele é {} e ele tem {} anos.".format(nome, idade))
```

Nesse exemplo, "{}" é um espaço reservado que será substituído pelos valores de nome e idade. A função `format()` é usada para substituir os espaços reservados pelos valores de variáveis especificados.

Existe uma alternativa para a função `format()`, que é usar f-strings (também conhecido como literais de string formatada). Essas f-strings permitem que você insira variáveis e expressões diretamente em uma string, e o Python avaliará essas expressões e substituirá os valores correspondentes na string. Aqui está um exemplo:



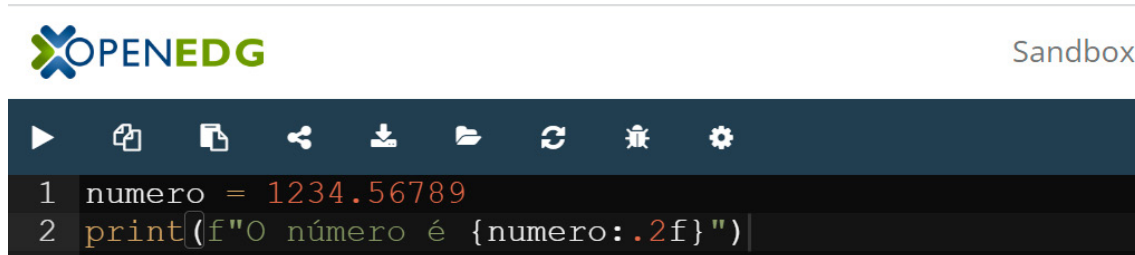
The screenshot shows the OpenEDG Sandbox interface. At the top, there is a logo for 'XOPENEDG' and the word 'Sandbox' on the right. Below the header is a toolbar with icons for running, copying, saving, sharing, downloading, opening, refreshing, and settings. The main area contains three lines of Python code:

```
1 nome = "João"
2 idade = 30
3 print(f"O nome dele é {nome} e ele tem {idade} anos.")
```



Nesse exemplo, a string é precedida pelo prefixo “f”, indicando que é uma f-string. As variáveis são incluídas dentro da string, dentro de chaves {}. Quando a string é impressa, o Python avalia as expressões dentro das chaves e as substitui pelos valores correspondentes.

Você também pode usar expressões dentro das chaves {} para formatar os valores de variáveis. Por exemplo:



The screenshot shows a web-based Python IDE interface. At the top left is the 'OPENEDG' logo, and at the top right is the word 'Sandbox'. Below the header is a toolbar with icons for running, copying, saving, sharing, downloading, opening files, refreshing, and settings. The code editor contains two lines of Python code:

```
1 numero = 1234.56789
2 print(f"O número é {numero:.2f}")
```

Nesse caso, a expressão “.2f” é usada para formatar o valor de número como um número de ponto flutuante com duas casas decimais.

As f-strings são uma adição relativamente recente ao Python e estão disponíveis a partir da versão 3.6. Se você estiver usando uma versão anterior do Python, a função format() pode ser a melhor opção para formatar suas strings de saída.

## 4. Ambiente Integrado de Desenvolvimento (IDE)

Para programar em Python, existem diversos ambientes integrados de desenvolvimento (IDEs) disponíveis no mercado. Entre os mais populares estão:

- PyCharm: é um IDE desenvolvido pela JetBrains e é considerado um dos melhores ambientes para programar em Python. Ele possui uma interface intuitiva e completa, com suporte à depuração, aos testes e ao controle de versão;
- Visual Studio Code: é um editor de código aberto desenvolvido pela Microsoft, que oferece suporte para diversas linguagens de programação, incluindo Python. Ele possui uma grande quantidade de extensões e é bastante customizável;
- IDLE: é o ambiente integrado padrão que acompanha a instalação do Python. Ele é simples e fácil de usar, mas possui recursos limitados.

Para fazer o download do Python, acesse o site oficial do Python: <https://www.python.org/downloads/>. Na página de downloads, você verá a opção de download para diferentes sistemas operacionais. Selecione o instalador apropriado para o seu sistema operacional (Windows, MacOS, Linux etc.). Por fim, escolha a versão do Python que você deseja baixar.

Para baixar o Visual Studio Code e o utilizar para programar em Python, acesse o site oficial do Visual Studio Code: <https://code.visualstudio.com/>. Clique no botão “download” para baixar o instalador do Visual Studio Code para o seu sistema operacional. Após o download, execute o arquivo de instalação e siga as instruções na tela para concluir a instalação.

Então, abra o Visual Studio Code e clique na opção “Extensions” no menu lateral esquerdo. Na barra de pesquisa, digite “Python” e selecione a extensão “Python” da Microsoft. Clique no botão “Install” para instalar a extensão Python no Visual Studio Code.

Com o Visual Studio Code e a extensão Python instalados, você pode criar, editar e executar seus projetos em Python diretamente na IDE. Para criar um arquivo Python, basta clicar no botão “New File”, na barra lateral esquerda, e salvar o arquivo com a extensão “.py”. Para executar o código, use o terminal integrado ou acesse as opções de execução disponíveis no menu superior do Visual Studio Code. Recomendamos o uso desse ambiente para que você possa escrever e testar seus projetos.

## Considerações Finais da Aula

A linguagem de programação Python é uma das linguagens mais populares entre os desenvolvedores por sua eficiência e simplicidade. Com ela, podemos construir variáveis, trabalhar com diferentes tipos de dados e utilizar comandos de entrada e saída, além de operadores para processar os dados.

Em resumo, Python é uma linguagem de programação popular e versátil, que possui uma grande comunidade de desenvolvedores e bibliotecas disponíveis. Ela é fácil de ser aprendida, é legível e pode ser executada em diversas plataformas. Para programar em Python, existem diversas opções de ambientes integrados de desenvolvimento, cada um com suas próprias características e recursos.

## Materiais Complementares



### Python e Django

2020, Francisco Marcelo de B. Maciel. Editora Alta Books.

Esse livro apresenta mais informações sobre os conceitos básicos apresentados nesta aula sobre as características da linguagem Python, a sua sintaxe e os seus comandos básicos.



### AlexandreLouzada/Pyquest: Pyquest/envExemplo/Lista01

2023, Alexandre N. Louzada. Github.

O link indicado permite o acesso a um repositório com várias listas de exemplo de programas em Python para auxiliar estudantes de programação.

Link para acesso: <https://github.com/AlexandreLouzada/Pyquest/tree/master/envExemplo/Lista01> (acesso em 24 maio 2023.)

## Referências

ALVES, William P. *Programação Python*: aprenda de forma rápida. [s.l.]: Editora Saraiva, 2021.

PYTHON SOFTWARE FOUNDATION. *Python Language Site*. Documentation, 2023. Página de documentação. Disponível em: <https://docs.python.org/pt-br/3/tutorial/index.html>. Acesso em: 8 mar. 2023.