



# Introdução ao Processamento de Dados

## Turma (2025.1)



# Introdução ao Python

Tassio Sirqueira (IME/UERJ)

[tassio.sirqueira@ime.uerj.br](mailto:tassio.sirqueira@ime.uerj.br)

# Histórico



Foi concebida no final de 1989 pelo holandês Guido Van Rossum

O código só foi publicado em 1991

A versão 1.0 foi lançada em 1994



# Histórico



Foi concebida no final de 1989 pelo holandês Guido Van Rossum

O código só foi publicado em 1991

A versão 1.0 foi lançada em 1994



# Histórico

De acordo com a revista [Spectrum da IEEE](#) (Institute of Electrical and Electronics Engineers) o Python é em 2020 a **principal linguagem de programação** no mundo.

É a linguagem escolhida como a **mais adequada para o ensino** introdutório de computação na maior parte das principais universidades dos EUA.

Algumas organizações que usam Python: Google, Yahoo!, Wikipedia, CERN, NASA, Facebook, Amazon, Instagram, Spotify, ...

## Histórico

Muito utilizada para computação científica, na área financeira, em programação web, etc.

É a linguagem mais utilizada em *machine learning* e *deep learning*.

Se você for trabalhar com **inteligência artificial**, não vai escapar do Python.

# Um exemplo...

Estas faces foram produzidas por um programa escrito em Python (e com a biblioteca TensorFlow).

Elas não são de pessoas vivas/reais.

O programa (StyleGAN) aprende a gerar faces através de exemplos.

Baseado em um conjuntos de técnicas chamadas de *deep learning*.



Karras et al., “Analyzing and Improving the Image Quality of StyleGAN”, 2020

# Um exemplo...



Karras et al., “A Style-Based Generator Architecture for Generative Adversarial Networks” 2019

# Características

Python é uma linguagem interpretada (não compilada).

Linguagem de altíssimo nível (*Very High Level Language*),  
Moderna, de sintaxe fácil e concisa.

Tipagem dinâmica (não há declaração de variáveis).

Modular.

Multiplataforma.

Código livre.

# Características

Python é uma linguagem interpretada (não compilada).

Linguagem de altíssimo nível (*Very High Level Language*),

Permite usar seguintes metodologias de programação:

- Estruturada

- Funcional

- Orientada a objetos

# Documentação

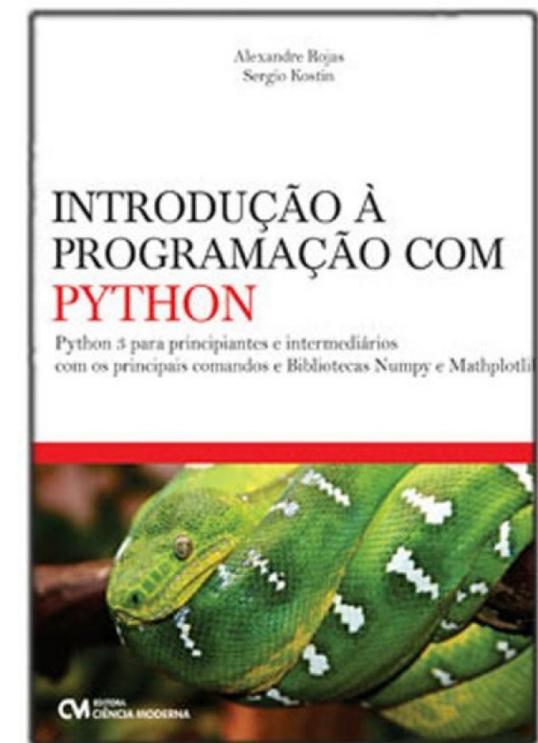
Existem diversas livros e sites sobre programação com Python.

Documentação oficial: <http://wiki.python.org.br/DocumentacaoPython>

Livro de professores do IME UERJ (Alexandre Rojas e Sérgio Kostin).

Introdução à Programação com Python.

Editora Ciência Moderna.



# Versões do Python

As principais versões são (<https://www.python.org/downloads/>)

2.7.x (última versão 2.7.13 – dezembro de 2016)

3.13.x (última versão 3.13.2 – fevereiro de 2025)

## Python 2

Foi o padrão da linguagem por muito tempo (primeira versão lançada em 2010). **Descontinuada** em janeiro de 2020.

## Python 3

Algumas mudanças a tornaram **incompatível com a versão 2**.

Constantemente evoluindo e recebendo novas funcionalidades, que não estarão presentes na versão anterior.

# Versões do Python

Existem inúmeras ferramentas de desenvolvimento para Python.

***Integrated Development Environments*** (IDEs) são pacotes de software que integram várias ferramentas de desenvolvimento, com o objetivo de aumentar a produtividade do desenvolvedor.

Geralmente as IDEs incluem recursos como ***syntax highlight*** (código fonte colorizado conforme a sintaxe da linguagem), ***code completion*** (o editor apresenta durante a digitação formas possíveis de completar o texto), ferramentas de ***depuração (debug)***, ***Shell*** integrado, etc.

# Integrated Development Environments (IDEs) Python

## Python.org (IDLE)

Ambiente Python padrão.

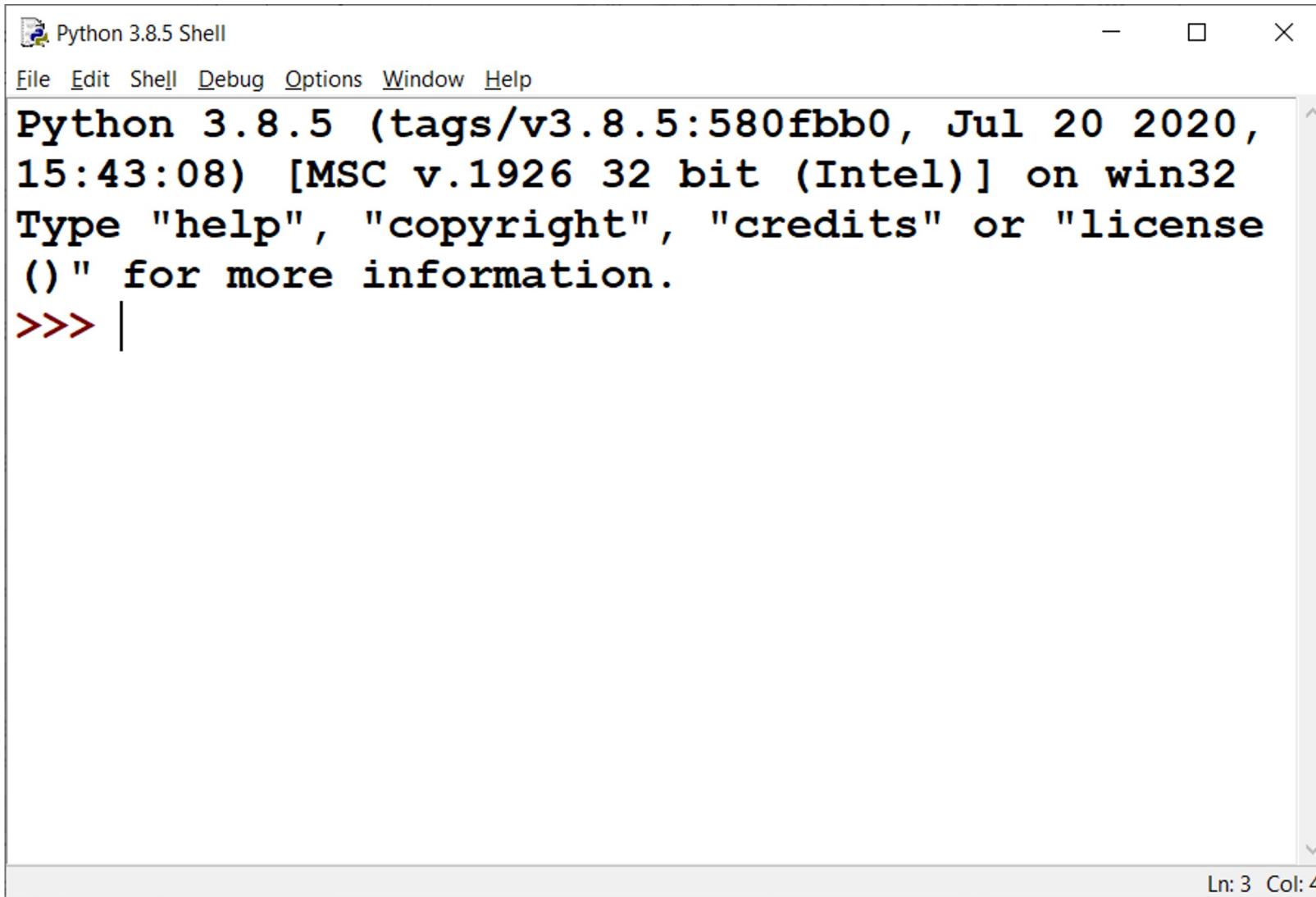
Vocês podem baixar a versão apropriada para o seu sistema operacional (Windows, MacOS, Linux) através do link:

<https://www.python.org/downloads/>

O IDE propriamente dito chama-se **IDLE**: contém interfaces para a criação, execução e depuração de programas Python.

Permite a execução de comandos do Python de forma interativa (Python **Shell**).

# IDLE (Python.org)

A screenshot of the Python 3.8.5 Shell window. The window title is "Python 3.8.5 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main text area displays the Python startup message:

```
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020,  
15:43:08) [MSC v.1926 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license  
The command prompt ">>> |" is visible at the bottom left. A status bar at the bottom right shows "Ln: 3 Col: 4".

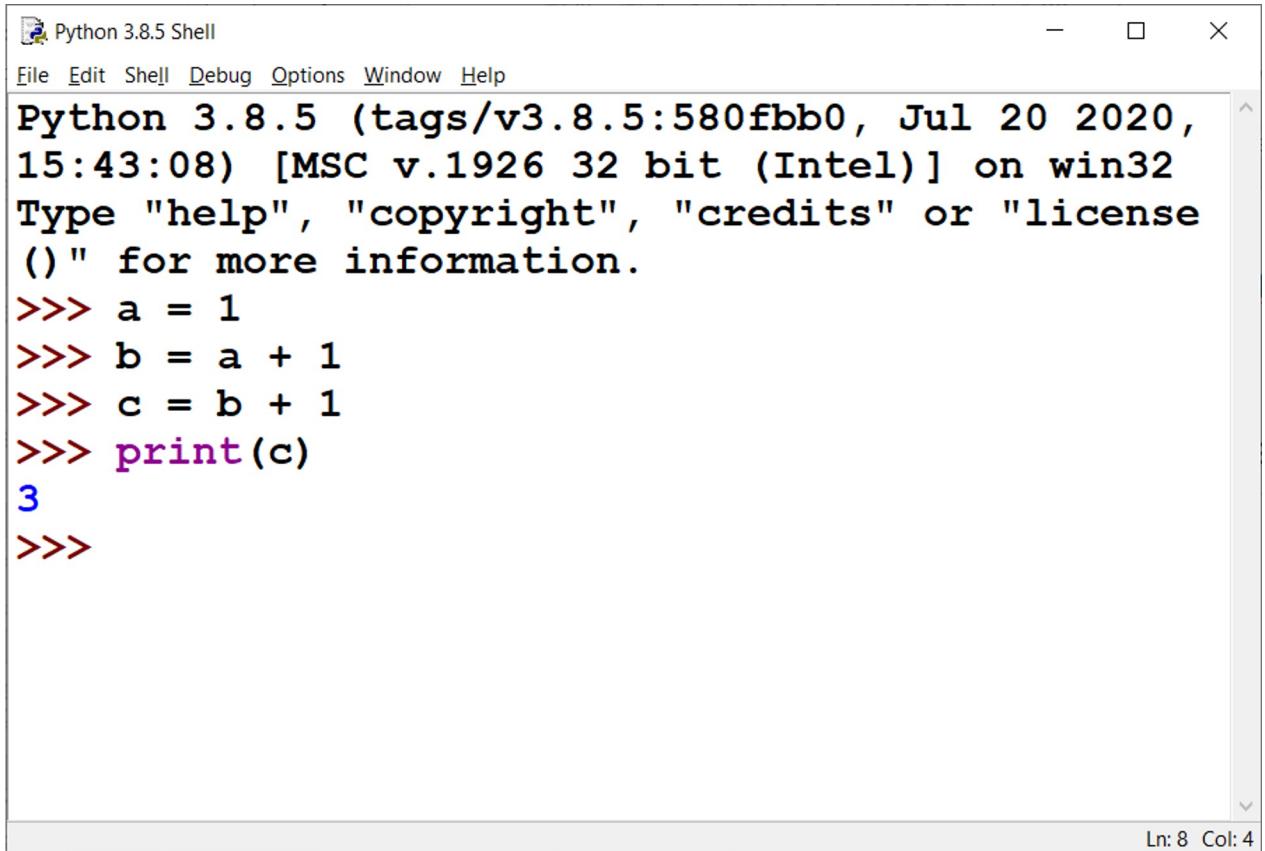
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020,  
15:43:08) [MSC v.1926 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license  
|>>>


```

# IDLE (Python.org)

Quando se abre o IDLE, ele mostra um **Python Shell**.

No Python *Shell* você pode  
**executar comandos Python**  
interativamente.



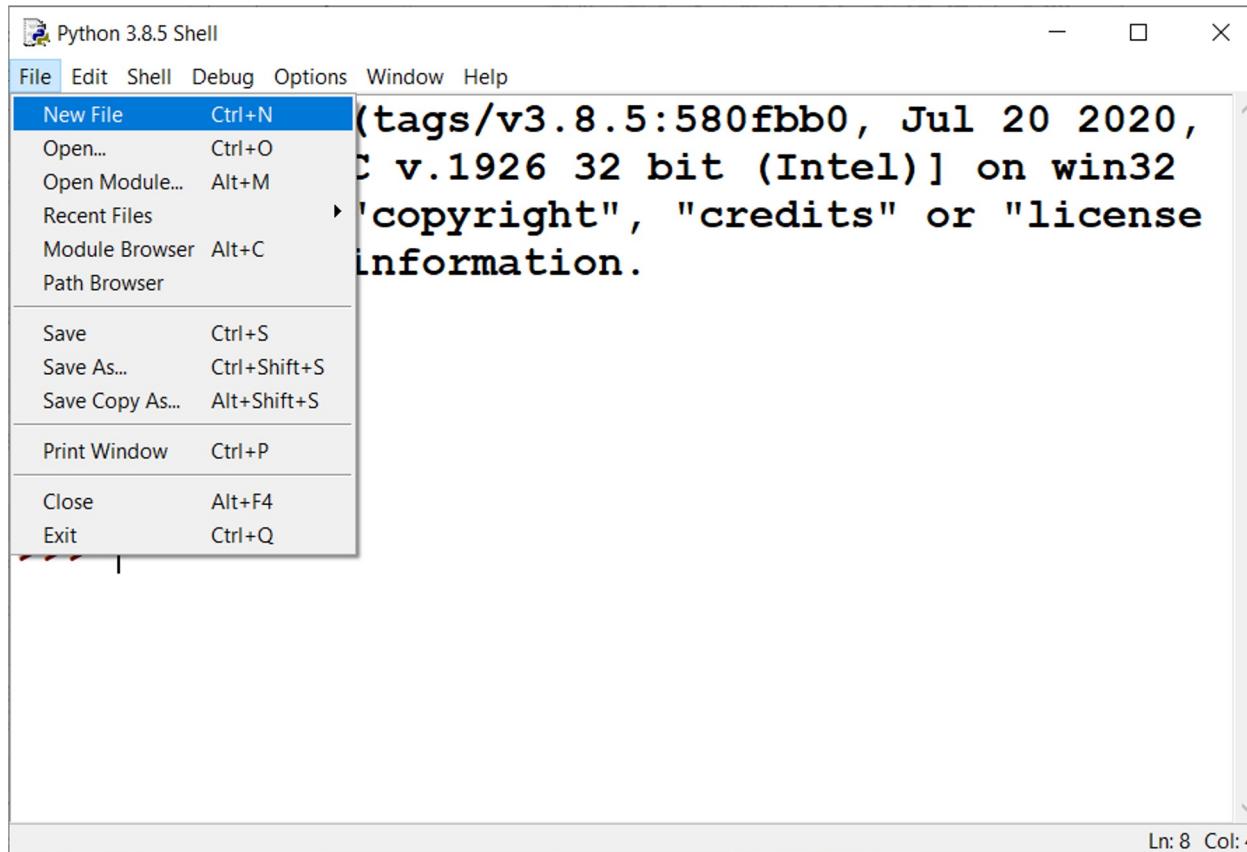
The screenshot shows a Windows application window titled "Python 3.8.5 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the Python interpreter's welcome message and a series of commands entered by the user:

```
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020,  
15:43:08) [MSC v.1926 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license"  
(()) for more information.  
>>> a = 1  
>>> b = a + 1  
>>> c = b + 1  
>>> print(c)  
3  
>>>
```

In the bottom right corner of the window, there is a status bar with "Ln: 8 Col: 4".

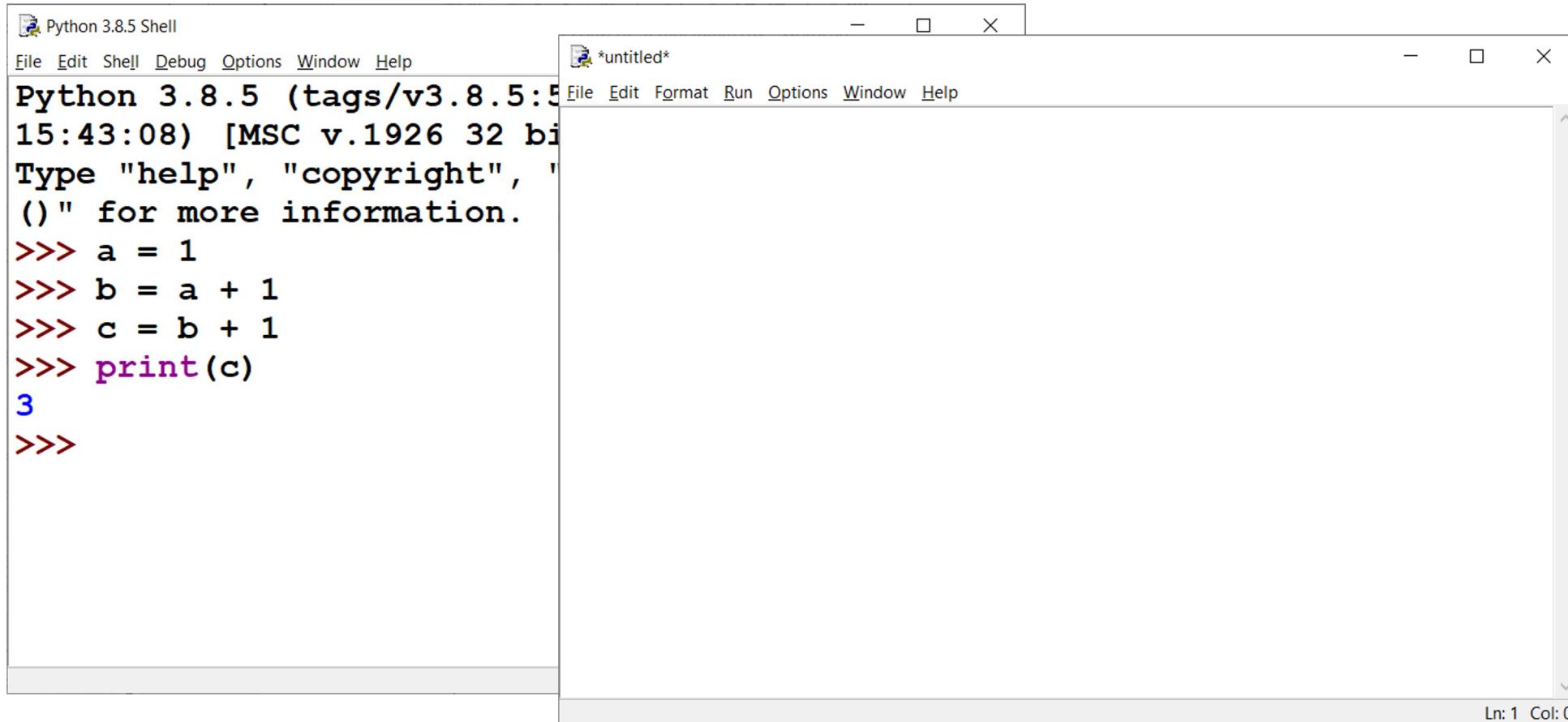
# IDLE (Python.org)

Para criar um programa Python, você deve **abrir uma nova janela**, escrever seu programa e salvá-lo num arquivo com extensão .py



# IDLE (Python.org)

Para criar um programa Python, você deve **abrir uma nova janela**, escrever seu programa e salvá-lo num arquivo com extensão .py



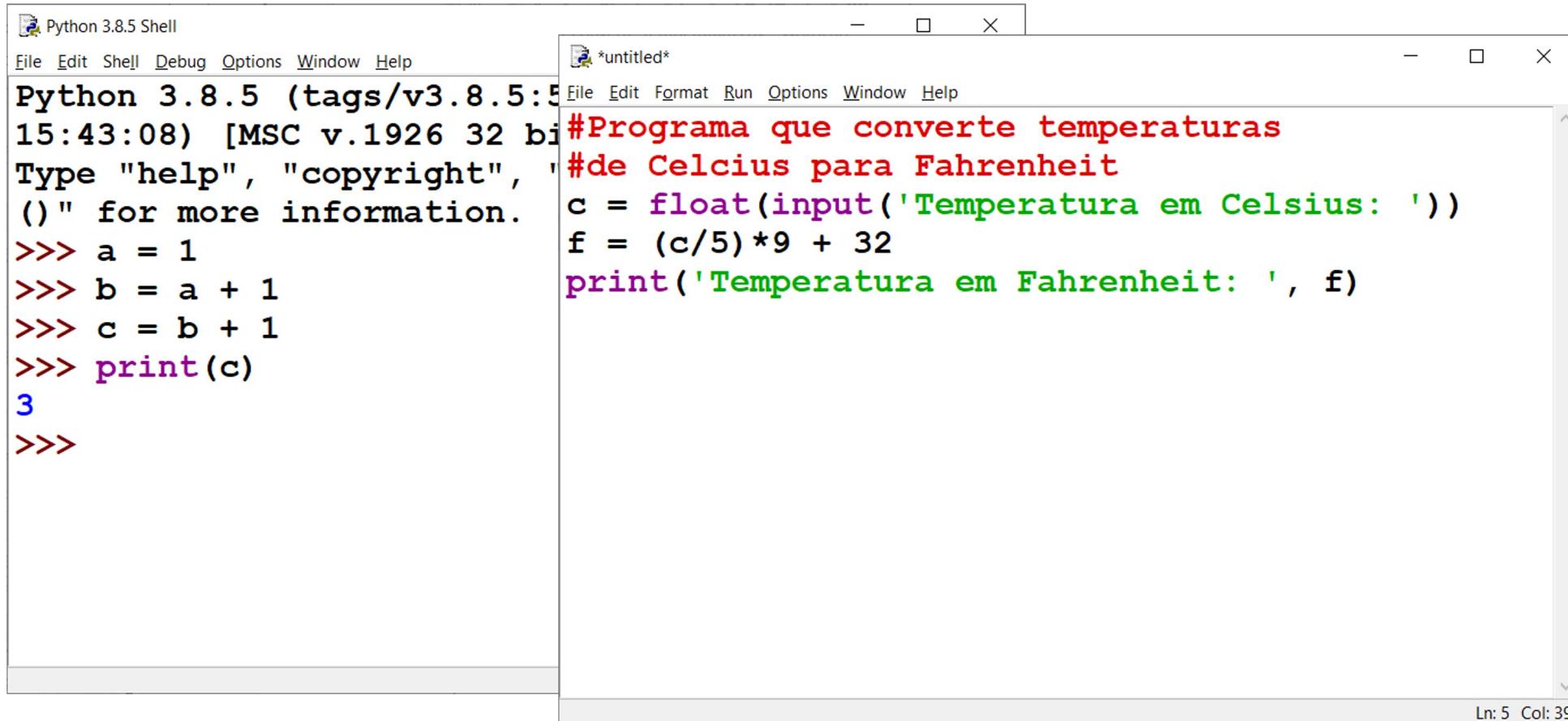
The screenshot shows the IDLE Python development environment. On the left is the "Python 3.8.5 Shell" window, which displays the Python interpreter's prompt and a short session of code execution:

```
Python 3.8.5 (tags/v3.8.5:5800a4d, Jan 29 2021, 15:43:08) [MSC v.1926 32 bit (Intel)]
Type "help", "copyright", 'credits' or "license" for more information.
>>> a = 1
>>> b = a + 1
>>> c = b + 1
>>> print(c)
3
>>>
```

On the right is the "\*untitled\*" script editor window, which is currently empty.

# IDLE (Python.org)

Para criar um programa Python, você deve abrir uma nova janela, **escrever seu programa** e salvá-lo num arquivo com extensão .py



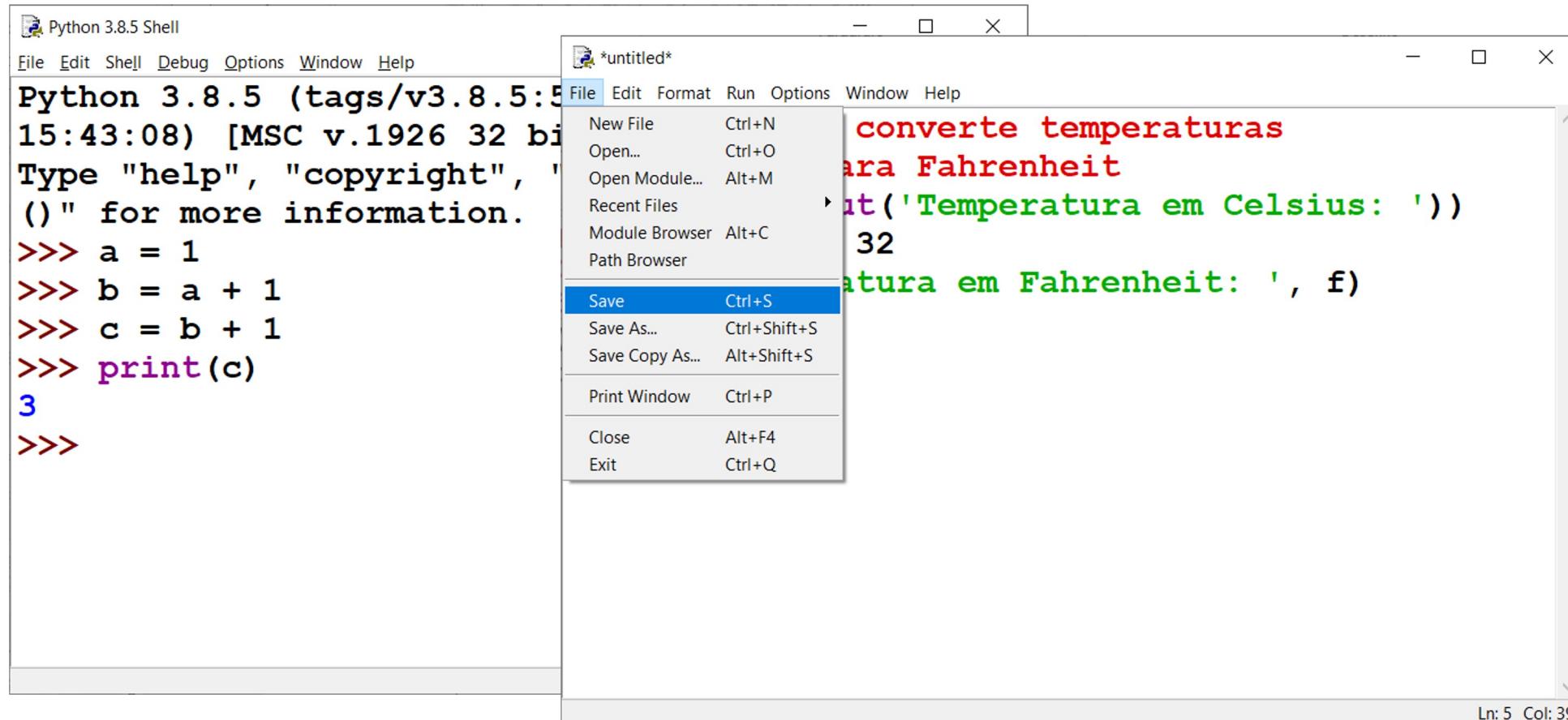
The image shows the Python IDLE interface with two windows open. The left window is the "Python 3.8.5 Shell", which displays the Python interpreter's prompt (">>>>"), some sample code, and the output of the execution. The right window is the "untitled\*" editor, which contains a script for converting Celsius to Fahrenheit.

```
Python 3.8.5 (tags/v3.8.5:5800a4d, Jan 29 2021, 15:43:08) [MSC v.1926 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> a = 1
>>> b = a + 1
>>> c = b + 1
>>> print(c)
3
>>>
```

```
#Programa que converte temperaturas
#de Celcius para Fahrenheit
c = float(input('Temperatura em Celsius: '))
f = (c/5)*9 + 32
print('Temperatura em Fahrenheit: ', f)
```

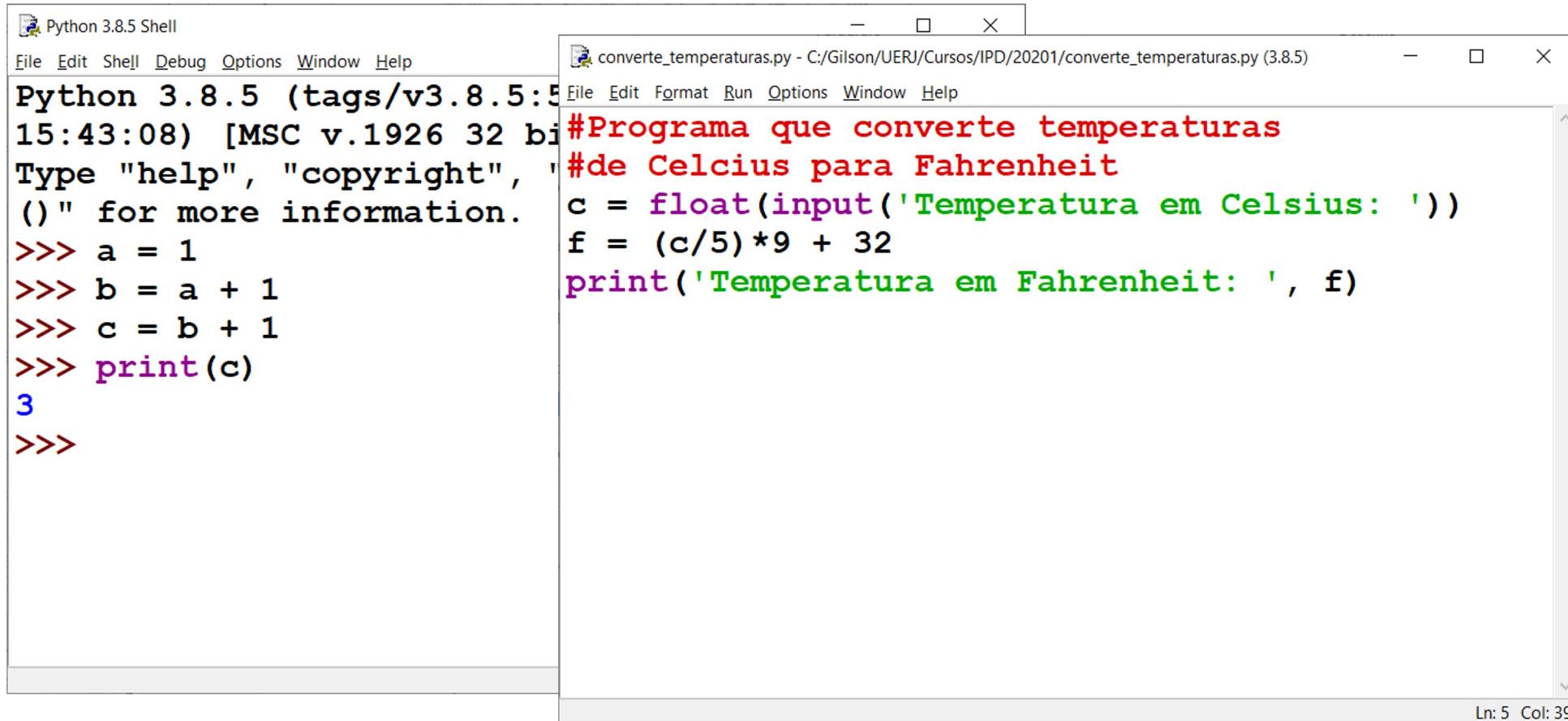
# IDLE (Python.org)

Para criar um programa Python, você deve abrir uma nova janela, escrever seu programa e **salvá-lo num arquivo** com extensão **.py**



# IDLE (Python.org)

Para criar um programa Python, você deve abrir uma nova janela, escrever seu programa e **salvá-lo num arquivo** com extensão **.py**



The image shows the Python IDLE interface with two windows open. The left window is the "Python 3.8.5 Shell", which displays the Python interpreter's prompt and some sample code execution:

```
Python 3.8.5 (tags/v3.8.5:5800a4d, Jan 29 2021, 15:43:08) [MSC v.1926 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> a = 1
>>> b = a + 1
>>> c = b + 1
>>> print(c)
3
>>>
```

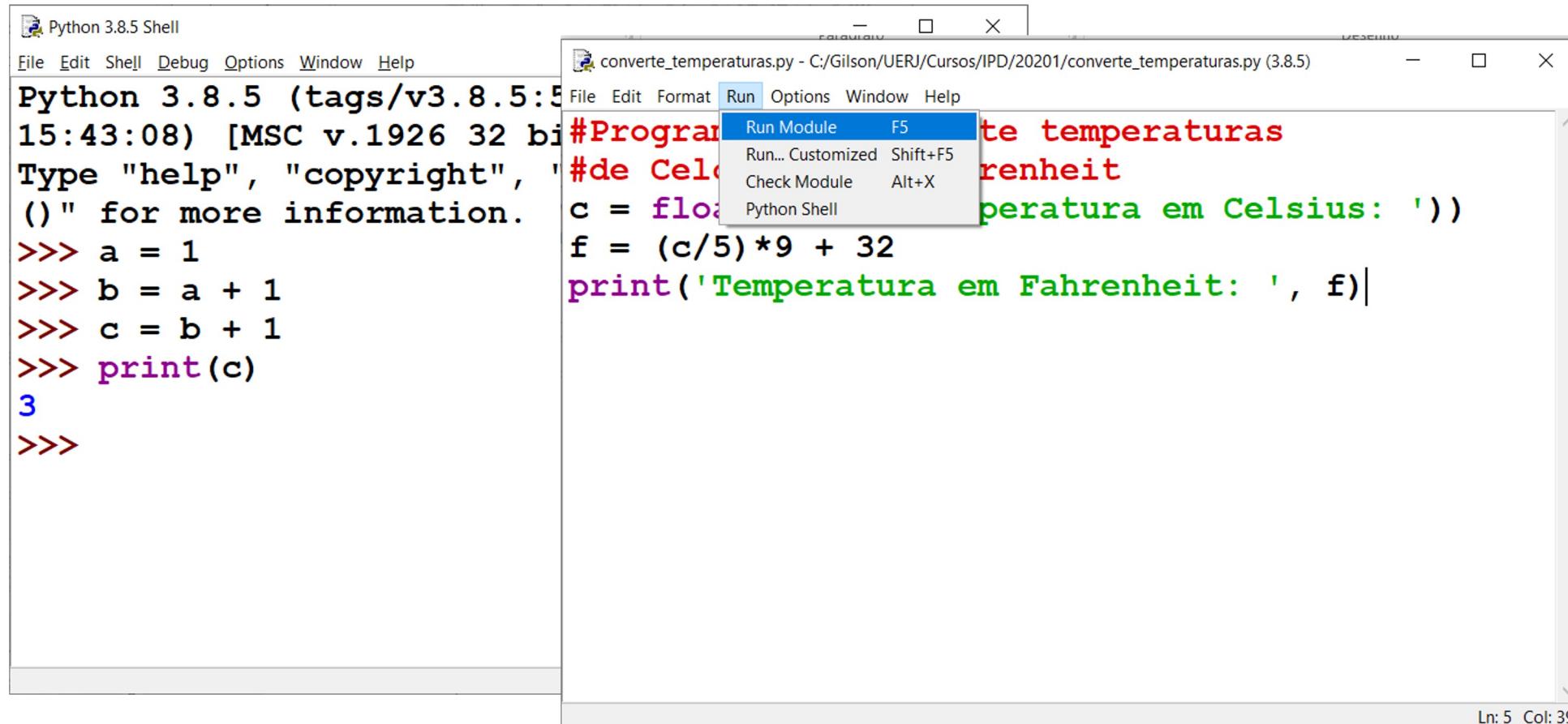
The right window is the "converte temperaturas.py" editor window, showing the source code for a temperature conversion program:

```
#Programa que converte temperaturas
#de Celcius para Fahrenheit
c = float(input('Temperatura em Celsius: '))
f = (c/5)*9 + 32
print('Temperatura em Fahrenheit: ', f)
```

The status bar at the bottom of the right window indicates "Ln: 5 Col: 39".

# IDLE (Python.org)

Para **executar o programa** basta escolher a opção *Run/Run Module*, ou pressionar F5.



# IDLE (Python.org)

As interações com o usuário e a apresentação do resultado ocorre no **Shell**.

The screenshot shows the Python 3.8.5 Shell window on the left and a script editor window on the right.

**Python 3.8.5 Shell:**

```
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:43:08) [MSC v.1926 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license"
() for more information.

>>> a = 1
>>> b = a + 1
>>> c = b + 1
>>> print(c)
3
>>>
= RESTART: C:\Gilson\UERJ\Cursos\IPD\20201\converte_temperaturas.py
Temperatura em Celsius: 39.0
Temperatura em Fahrenheit: 102.2
>>> |
```

**Script Editor:**

```
temperaturas
Fahrenheit
Temperatura em Celsius: ''))
Fahrenheit: ', f)
```

# Integrated Development Environments (IDEs) Python

## Programiz

IDE de Python **on-line**.

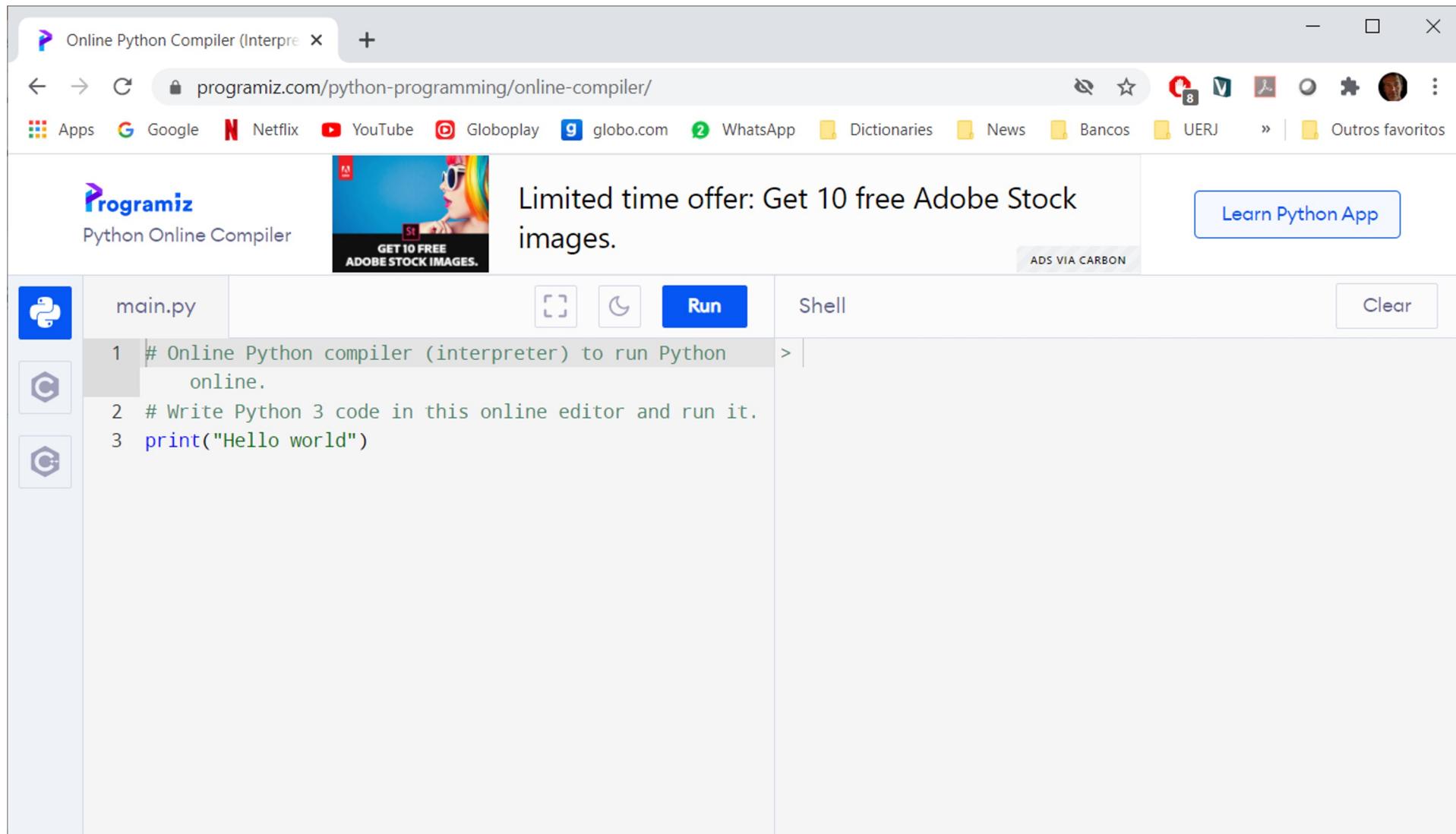
Permite a criação de programas Python e **execução diretamente num navegador**, independente do sistema operacional.

Não há necessidade de instalar nada no computador.

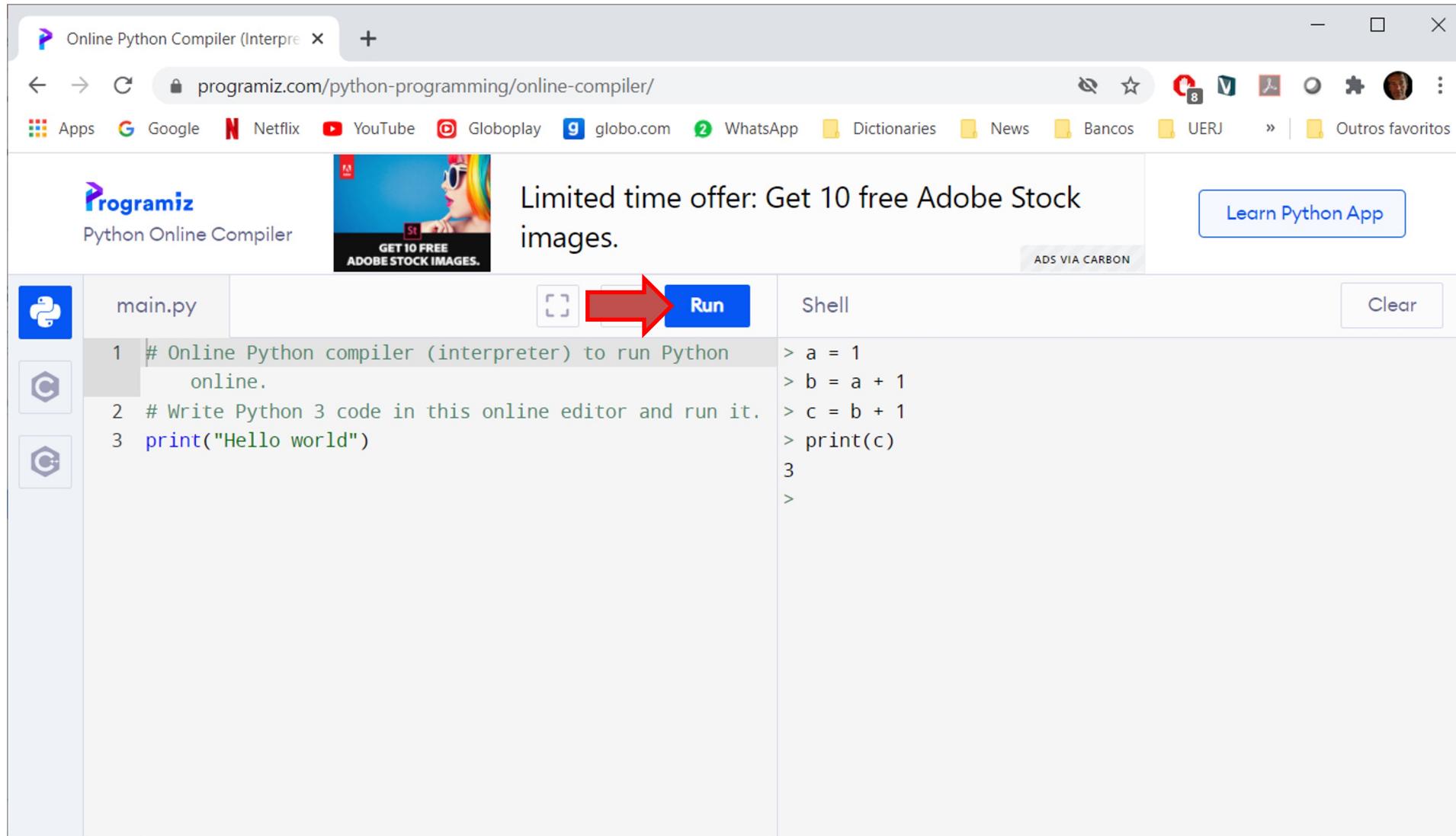
O Programiz pode ser acessado através do link:

<https://www.programiz.com/python-programming/online-compiler/>

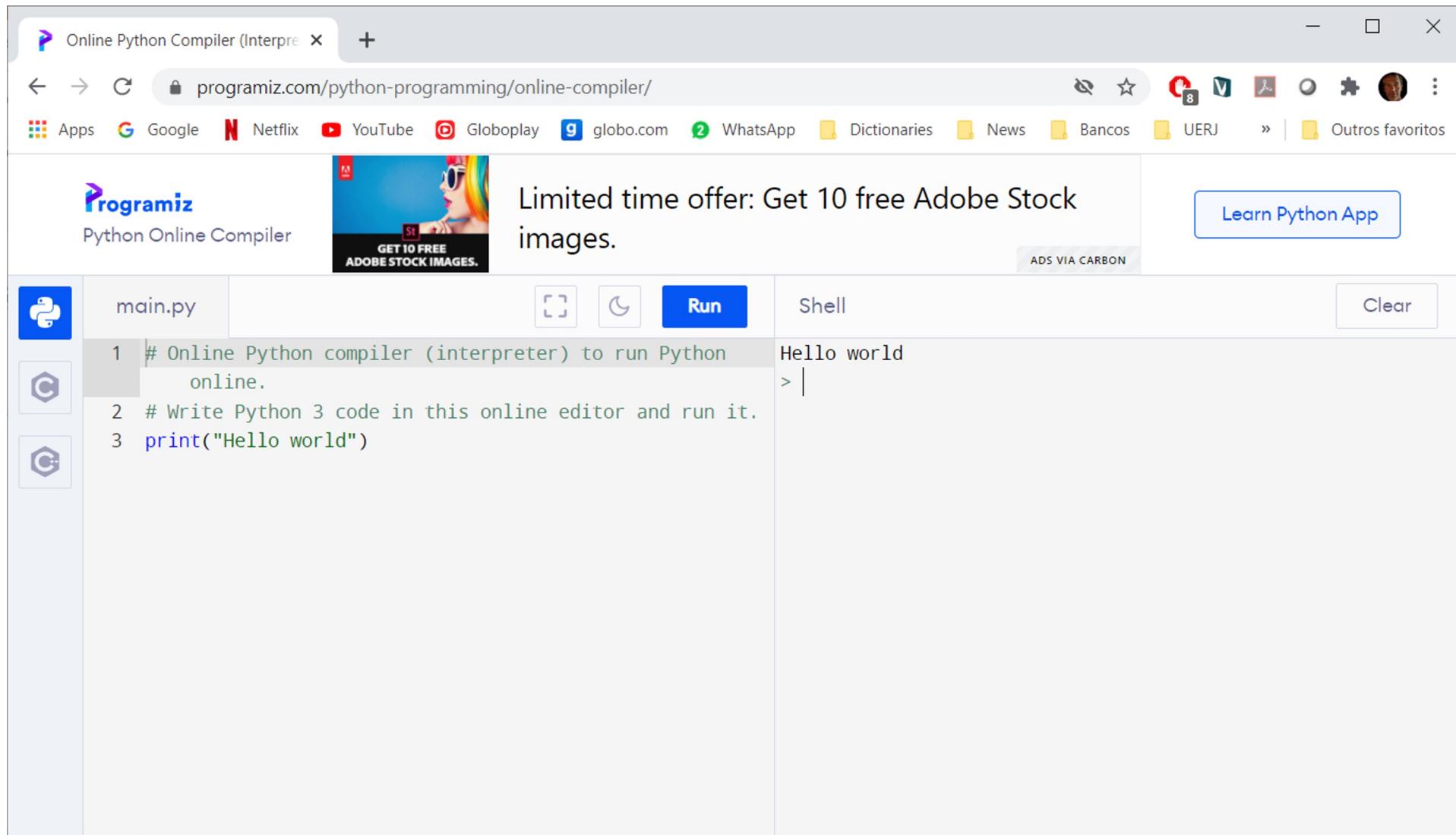
# Programiz



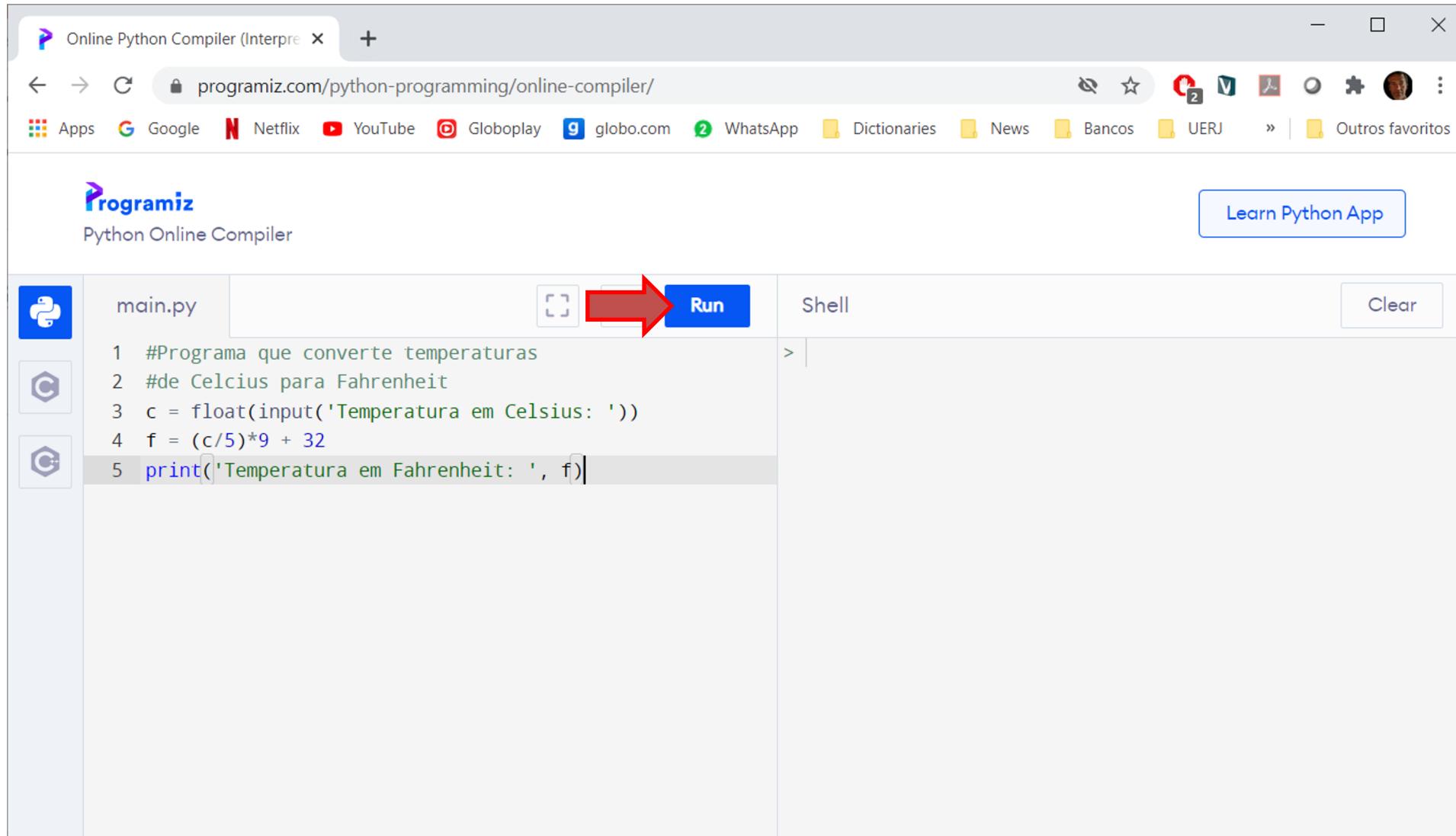
# Programiz



# Programiz



# Programiz



# Programiz

The screenshot shows the Programiz Python Online Compiler interface. At the top, there's a browser-like header with the title "Online Python Compiler (Interpre", a "+" button, and a search bar containing "programiz.com/python-programming/online-compiler/". Below the header is a toolbar with various icons for apps like Netflix, YouTube, and WhatsApp, along with links for "Dictionaries", "News", "Bancos", "UERJ", and "Outros favoritos". The main area features the "Programiz" logo and "Python Online Compiler" text. On the left, there's a sidebar with icons for Python, C, and C++. The main workspace has tabs for "main.py" and "Shell". The "main.py" tab contains the following Python code:

```
1 #Programa que converte temperaturas
2 #de Celcius para Fahrenheit
3 c = float(input('Temperatura em Celsius: '))
4 f = (c/5)*9 + 32
5 print('Temperatura em Fahrenheit: ', f)
```

The "Run" button is highlighted in blue. To the right, the "Shell" tab shows the output of the program:

```
Temperatura em Celsius: 39.0
Temperatura em Fahrenheit: 102.2
> |
```

A "Clear" button is located in the "Shell" tab area.

# Programiz

Para **salvar um programa** escrito no Programiz:

Copiar o programa para memória (Ctrl+C).

Abrir um editor de arquivos texto.

Colar o programa no editor (Ctrl+V).

Salvar o arquivo com a extensão .py

# **Integrated Development Environments (IDEs) Python**

## **QPython 3L**

Ambiente de desenvolvimento Python para celulares com sistema operacional Android.

É gratuito e pode ser obtido através do Google Play.

## **Pythonista e Pyto**

Ambientes de desenvolvimento Python para iPhone.

Aplicativos pagos.

Podem ser obtidos através do App Store.

# Identificadores

Um identificador é **qualquer nome aceito** pela linguagem:

Variáveis

Palavras reservadas

Python é ***Case Sensitive***

Diferencia maiúscula de minúscula.

AB, Ab, aB e ab são variáveis diferentes!

Podem ter qualquer tamanho.

São formadas por letras, números e sublinhado.

Iniciam sempre por uma letra.

# Variáveis

O **nome de uma variável** é um identificador.

Podem ter qualquer quantidade de caracteres.

Exemplo: notaAlunoUERJPrimeiroSemestre2020

**Tipagem dinâmica:** não precisam ser declaradas.

**Tipagem forte:** a partir da atribuição de um valor elas passam a ser do tipo do valor atribuído.

# Tipos de dados simples

Inteiros: *int* (4 bytes)

Inteiros longos: *long* (8 bytes)

Reais: *float* (8 bytes)

Números complexos: *complex* (8 bytes)

Literais: *str* (um byte por caractere)

Lógicos: *bool* (1 byte)

# Indentação

Os **blocos de comandos** são delimitados em Python pela **indentação**.

Significa **rekuo** (derivado da palavra em inglês *indentation*, também grafado nas formas identação e endentação)

O número de espaços no rekuo é variável, mas todas as instruções dentro de um bloco têm de ser recuadas na mesma quantidade de espaços.

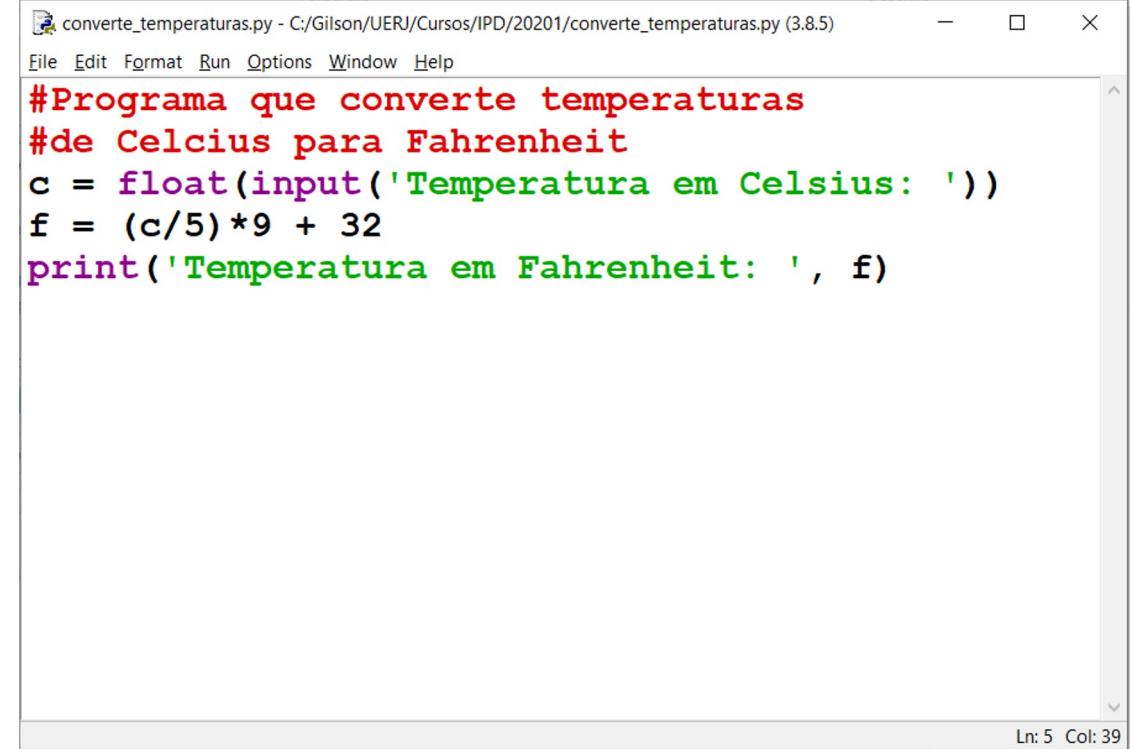
Em Python a **indentação é obrigatória!**

# Comentários

São muito importantes para dar **clareza aos programas**.

São ignorados pelo interpretador: é como se não existissem.

São precedidos pelo caractere # (tralha ou *hash*).



The screenshot shows a window titled "converte\_temperaturas.py - C:/Gilson/UERJ/Cursos/IPD/20201/converte\_temperaturas.py (3.8.5)". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code itself is as follows:

```
#Programa que converte temperaturas
#de Celcius para Fahrenheit
c = float(input('Temperatura em Celsius: '))
f = (c/5)*9 + 32
print('Temperatura em Fahrenheit: ', f)
```

In the bottom right corner of the code editor, there is a status bar displaying "Ln: 5 Col: 39".

# Operadores Aritméticos

- + (soma)
- (subtração)
- \* (multiplicação)
- / (divisão)
- // (divisão inteira)
- % (resto/módulo da divisão inteira)
- \*\* (exponenciação)

## *Testar no Python...*

Nesta aula vamos usar o *Programiz* para executar comandos e escrever programas Python.

<https://www.programiz.com/python-programming/online-compiler/>

# Operadores Aritméticos

Testar no Python:

`4 + 3 - 2`

`2 * 3`

`7 / 2`

`7 // 2`

`7 % 2`

`2 ** 3`

`a = a + 1`

`a += 1 # a = a + 1`

# Precedência de Operadores

1. ()
2. \*\*
3. \*, /, //, %
4. +, -

Em caso de mesma hierarquia resolve-se da esquerda para direita:

$$4 + (3^{**}2) // 3 - 4 \% 3 * 5 - 2$$

# Operadores Relacionais

`==` (igual)

`!=` (diferente)

`>` (maior que)

`>=` (maior ou igual)

`<` (menor que)

`<=` (menor ou igual)

Um operador relacional **retorna sempre um valor lógico *False* ou *True*.**

# Operadores Relacionais

Testar no Python:

```
a = 2
```

```
b = 3
```

```
a == b
```

```
a != b
```

```
a > b
```

# Operadores Lógicos

and        (*e* lógico)

or    (*ou* lógico)

not        (*não* lógico)

**Um operador lógico retorna sempre um valor lógico *False* ou *True*.**

# Operadores Lógicos

Testar no Python:

```
a = 2
```

```
b = 3
```

```
a == b
```

```
a == b and a != b
```

```
a == b or a != b
```

```
not (a == b and a != b)
```

# Comandos de Entrada e Saída

Comandos de **entrada e saída** são usados para enviar dados para o programa e mostrar os resultados do programa.

Comandos básicos (funções de entrada e saída):

Entrada: `input(mensagem)`

Saída: `print(lista_de_variaveis)`

# Comandos de Entrada e Saída

**Sintaxe** do comando de saída:

```
print(lista_de_variaveis)
```

*lista\_de\_variaveis* contém uma lista das variáveis (ou constantes) a ser impressa (apresentada na tela).

Por exemplo:

```
print(a)
```

```
print(b)
```

```
print(a, b)
```

```
print('Temperaturas:', c, 'Celsius ==', f, 'Fahrenheit')
```

# Comandos de Entrada e Saída

**Sintaxe** do comando de entrada:

*variavel* = input('mensagem')

*variavel* é o nome da variável a ter um valor atribuído pelo comando.

*mensagem* é um texto para auxiliar o usuário (para saber o que ele tem que fazer).

A mensagem é opcional:

*variavel* = input()

# Comandos de Entrada e Saída

Testar no Python:

```
a = input('Entre com um número: ')
b = input('Entre com outro número: ')
c = a + b
print('A soma dos dois números é:', c)
```

Como assim?

# Comandos de Entrada e Saída

O comando `input()` só lê ***strings*** (cadeias de caracteres).

Você deve **transformar a saída** do comando, indicando o tipo de dado desejado.

Por exemplo:

```
a = int(input('Entre com um número inteiro: '))
b = float(input('Entre com um número real: '))
```

**Cuidado:** se o usuário entrar com um valor de tipo errado, seu programa vai dar erro!

## *Vamos programar...*

Fazer um programa para ler 2 números e imprimir a soma.

```
a = int(input('n1: '))
b = int(input('n2: '))
c = a + b
print('soma: ', c)
```

## *Vamos programar...*

Fazer um programa para ler 4 números e imprimir a média.

```
a = float ('N1: ')
b = float ('N2: ')
c = float ('N3: ')
d = float ('N4: ')
result = (a+b+c+d)/4
print('soma: ', result)
```

## *Vamos programar...*

Fazer um programa para ler o valor do tempo em segundos e imprimir em hora, minuto e segundos. Ex: 4000s = 1h 6min 40s

```
totalSeg = int(input('tempo em segundos: '))
hora = totalSeg // 3600
minuto = (totalSeg % 3600) // 60
segundo = (totalSeg % 3600) % 60
print(hora,':',minuto,':',segundo)
```



# Introdução ao Processamento de Dados

## Turma (2025.1)



# Introdução ao Python

**Tassio Sirqueira (IME/UERJ)**

[tassio.sirqueira@ime.uerj.br](mailto:tassio.sirqueira@ime.uerj.br)