

>> UNIVERSIDADE FEDERAL DO PARÁ <<
>> INSTITUTO DE TECNOLOGIA <<
>> RAMO ESTUDANTIL IEEE UFPA <<
>> WOMEN IN ENGINEERING UFPA <<



>> GRUPO DE ESTUDOS EM PYTHON <<
#WIECODEPYTHON

>> SEMANA #2 <<
>> PYTHON COMO CALCULADORA <<
>> APOSTILA DE REFERÊNCIA <<

>> SUMÁRIO

1. Introdução.....	2
2. Guia de Instalação do Python.....	2
3. Ambientes de Desenvolvimento.....	4
4. Hello, World!.....	7
5. Python como Calculadora.....	8

>> Introdução

O que é python?

Python é uma linguagem de programação interpretada, orientada a objetos, de alto nível e com semântica dinâmica. Diz-se que uma linguagem é interpretada se esta não precisar ser compilada (traduzida para uma linguagem da máquina), mas sim “lida” por um outro programa (chamado de interpretador) que traduzirá para a máquina o que seu programa quer dizer. O interpretador para Python é interativo, ou seja, é possível executá-lo sem fornecer um script (programa) para ele. Ao invés disso, o interpretador disponibilizará uma interface interativa onde é possível inserir os comandos desejados um por um e ver o efeito de cada um deles.

A simplicidade do Python reduz a manutenção de um programa. Python suporta módulos e pacotes, que encoraja a programação modularizada e reuso de códigos.

Python é uma linguagem poderosa e divertida. Com ela você pode fazer diversas coisas como:

- Construção de sistemas Web com Django, Flask, Pyramid, etc.
- Análise de dados, Inteligência Artificial, Machine Learning e etc com Numpy, Pandas, Matplotlib, etc
- Construção de aplicativos com Kivy e Pybee
- Construção de sistemas desktop com Tkinter, WxPython, etc.

>> Guia de Instalação

Linux

Provavelmente você já tem o Python instalado e configurado. Para ter certeza que ele está instalado e descobrir qual versão, abra um terminal e execute o comando:

```
$ python --version
```

Se o resultado do comando for *Python 3.8* (ou alguma versão igual ou superior a 3.5) o Python já está instalado corretamente. Caso o resultado do comando anterior tenha sido *Python 2.7.13* (ou qualquer versão do Python 2) tente rodar o seguinte comando, pois seu computador pode ter ambas versões 2 e 3 instaladas:

```
$ python3 --version
```

Caso tenha aparecido a mensagem bash: *python: command not found*, você pode instalá-lo da seguinte maneira:

Ubuntu e Debian

```
$ sudo apt install python3 3.1.2
```

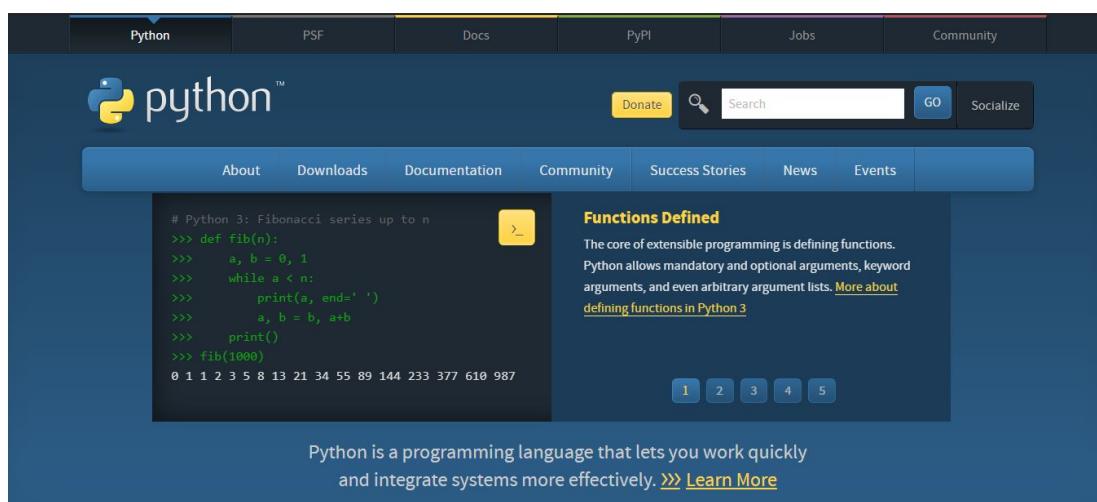
ArchLinux

```
$ sudo pacman -Sy python
```

Windows

Obtenha o arquivo de instalação última versão compatível com a arquitetura do seu computador no site oficial do Python.

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



A seguir, execute o instalador e uma imagem similar a essa aparecerá:



Deve ser selecionada a opção **Add Python 3.8 to PATH** e depois continuar a instalação até o fim.

>> Ambientes de Desenvolvimento

Há diversos programas para desenvolvermos códigos, alguns são mais bonitinhos, outros são mais poderosos, alguns são mais simples, outros são mais amigáveis. Dê uma olhada nesta seção e escolha o que você achar mais interessante.

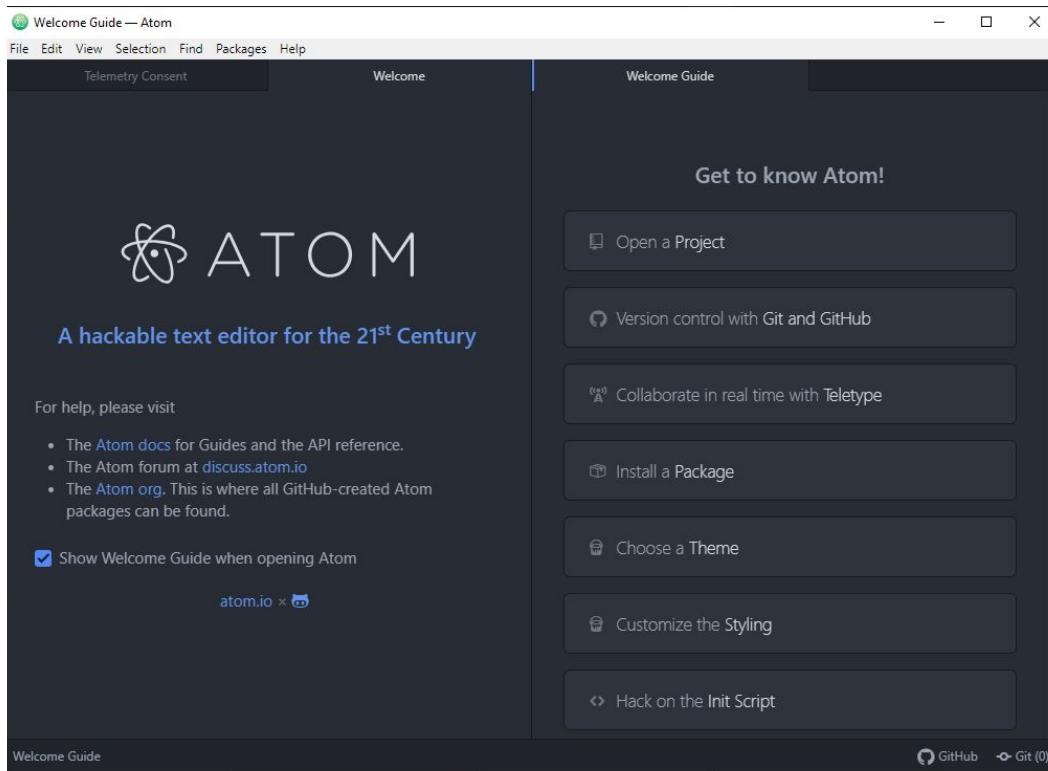
Somente você pode responder à pergunta «Qual o melhor ambiente de desenvolvimento para mim?»

IDE (Integrated Development Environment - Ambiente de Desenvolvimento Integrado, em português) é um editor de texto que possui ferramentas e recursos que facilitam a vida do programador.

Entre as ferramentas e recursos, podemos citar:

- Identificar quais variáveis foram declaradas.
- Identificar erros no código.
- Personalizar o ambiente de trabalho.
- Ocultar parte do código para melhor visualização.

Atom

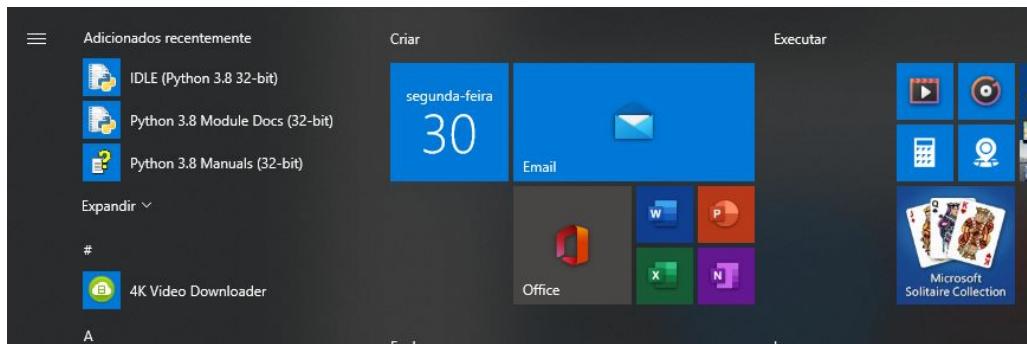


O programa ATOM é um IDE open-source que apresenta diversos pacotes para personalizar. No site oficial do ATOM, você encontrará um link para a Documentação do programa. Na documentação, é possível acessar o manual que mostrará passo a passo como instalar o programa (tanto para Windows como para Linux).

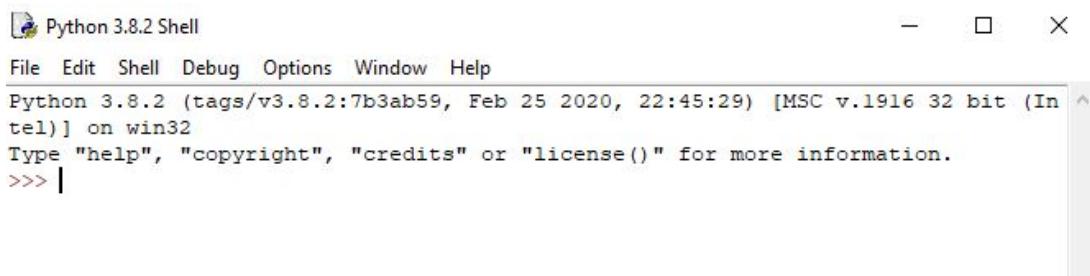
IDLE

Para usuários Windows é recomendado utilizar o IDLE. Ele é composto pelo interpretador do Python e um editor de texto para criar programas, e já vem junto com o Python.

Após seguir o Guia de Instalação do Python, o menu inicial deve estar da seguinte forma:



Ao abrir o IDLE (Python 3.X), aparecerá uma janela como na imagem abaixo:



PyCharm

Esta IDE é voltada especificamente para a linguagem Python. No site oficial é possível encontrar orientações para realizar o download e instalação (Linux, Mac, Windows).

É desenvolvido pela empresa tcheca JetBrains. Fornece análise de código, um depurador gráfico, teste de unidade integrado, integração com sistemas de controle de versão, ambiente virtual e suporta o desenvolvimento da Web com o Django, bem como Data Science com o Anaconda.

Visual Studio Code

O Visual Studio Code por si só é um editor de texto criado pela Microsoft que apresenta diversos pacotes para personalizá-lo da forma que você precisa.

No site oficial do Visual Studio Code, você encontrará um link para a Documentação do programa. Na documentação, é possível acessar o manual de Python que mostra todos as funcionalidades que o programa possui relacionadas a Python.

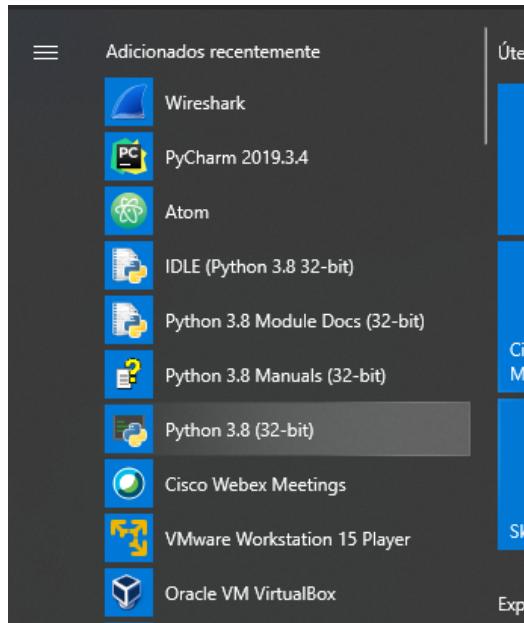
Python Shell

Existe também a possibilidade de trabalhar sem um ambiente gráfico, utilizando apenas a interface de linha de comando.

Se você instalou corretamente o Python, você tem à sua disposição um interpretador interativo, popularmente conhecido como o Shell do Python.

Se você utiliza Linux ou Mac, basta abrir um terminal e digitar **python**.

Caso utilize Windows, deverá existir uma pasta no menu iniciar chamada *Python 3.8* que permite abrir o interpretador.



Caso tenha sucesso, você deverá encontrar algo similar a esta imagem:

A screenshot of a terminal window titled "Python 3.8 (32-bit)". The window contains the following text:

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

The window has standard minimize, maximize, and close buttons at the top right.

Para sair do Python, digite `quit()`. Isso encerra a seção interativa e te retorna ao terminal (ou fecha a tela preta, no caso do Windows).

Note que, ao entrar no interpretador, a primeira linha começa com *Python 3.8.2*. Essa é a versão utilizada do Python. Caso a versão que aparecer em seu interpretador seja 2.7.6, encerre a sessão e tente o comando `python3` para utilizar a versão 3 do Python.

>> Hello, World!

É comum em programação começarmos com o famoso “Olá, Mundo!” ou em inglês “Hello, World!”.

Então vamos para nosso primeiro código.

Abra seu editor de texto favorito, aqui vamos utilizar o IDLE. Mas os códigos funcionam igualmente em qualquer editor. Digite o texto abaixo (não esqueça de apertar enter no final):

```
>>> print ("Hello, World!")  
Hello, World!
```

Função print()

print() é uma função nativa do Python. Basta colocar algo dentro dos parênteses que o Python se encarrega de fazer a magia de escrever na tela :)

>> Python como Calculadora

Operadores matemáticos

A linguagem Python possui operadores que utilizam símbolos especiais para representar operações de cálculos, assim como na matemática (teste os operadores no seu ambiente de desenvolvimento):

Soma (+)

```
>>> 3+5  
8  
>>> 9+12  
21  
>>> 7+32  
39  
.
```

Para utilizar números decimais, use o ponto no lugar de vírgula:

```
>>> 4.5 + 7.3  
11.8  
>>> 7.9 + 18.2  
26.1  
>>> 3.6 + 34.1  
37.7
```

Subtração (-)

```
>>> 5-2  
3  
>>> 9-7  
2  
>>> 15-20  
-5  
>>> 45-74  
-29
```

Multiplicação (*)

```
>>> 2 * 5  
10  
>>> 4 * 9  
36  
>>> 10 * 10  
100  
>>> 2 * 2 * 2  
8
```

Divisão (/)

```
>>> 45 / 5  
9.0  
>>> 100 / 20  
5.0  
>>> 9 / 3  
3.0  
>>> 10 / 3  
3.333333333333335
```

Se fizermos uma divisão por zero:

```
>>> 2 / 0  
Traceback (most recent call last):  
  File "<pyshell#44>", line 1, in <module>  
    2 / 0  
ZeroDivisionError: division by zero
```

Como não existe um resultado para a divisão pelo número zero, o Python interrompe a execução do programa (no caso a divisão) e mostra o erro que aconteceu, ou seja, «ZeroDivisionError: division by zero».

Divisão Inteira (//)

```
>>> 10 // 3  
3  
>>> 11 // 2  
5  
>>> 100 // 6  
16
```

Resto da Divisão (%)

```
>>> 10 % 2  
0  
>>> 15 % 4  
3  
>>> 100 % 6  
4
```

Agora que aprendemos os operadores aritméticos básicos podemos seguir adiante.

Potenciação/Exponenciação (**)

Como podemos calcular 2^{10} ? O jeito mais óbvio seria multiplicar o número dois dez vezes:

```
>>> 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2 * 2  
1024
```

Porém, isso não é muito prático...

Por isso há um operador específico para isso, chamado de potenciação/exponenciação: **

```
>>> 2 ** 10  
1024  
>>> 10 ** 3  
1000  
>>> (10 ** 800 + 9 ** 1000) * 233  
4072540016513778250507740862653659129332715595723989246501699067518899000309551  
8372884555899351487085850908781778957638858456096468279589640343544868198000136  
9688543958807077794866143976192872389017280782837244051514550016751431331392474  
7564632275685539037854000532937561051459919257116928284103659788141579291436461  
5491481758261416529044183498405437453490995411931544216941588442964551525886778  
0314133204645184212592152733050063403478054121909337278892530383627259086060904  
7207399040849052064030761412552848198170011285308519212147204798619082071689288  
7069652209896067731424289114032539096431029588907958895079878861232463405049578
```

Raiz Quadrada

Lembrando que $\sqrt{x} = x^{\frac{1}{2}}$, então podemos calcular a raiz quadrada do seguinte modo:

```
>>> 4 ** 0.5  
2.0
```

Mas a maneira recomendada para fazer isso é usar a função `sqrt()` da biblioteca `math`:

```
>>> import math  
>>> math.sqrt(16)  
4.0
```

Na primeira linha do exemplo importamos, da biblioteca padrão do Python, o módulo `math` e então usamos a sua função `sqrt` para calcular $\sqrt{16}$. E se precisarmos utilizar o número π ?

```
>>> math.pi  
3.141592653589793
```

Não esqueça que é preciso ter executado `import math` antes de usar as funções e constantes dessa biblioteca.

Expressões Numéricas

Agora que já aprendemos diversos operadores, podemos combiná-los e resolver problemas mais complexos:

```
>>> 3 + 4 * 2  
11  
>>> 7 + 3 * 6 - 4 ** 2  
9  
>>> (3 + 4) * 2  
14  
>>> (8 / 4) ** (5 - 2)  
8.0
```

Quando mais de um operador aparece em uma expressão, a ordem de avaliação depende das regras de precedência.

O Python segue as mesmas regras de precedência da matemática. O acrônimo PEMDAS ajuda a lembrar essa ordem:

1. Parênteses
2. Exponenciação
3. Multiplicação e Divisão (mesma precedência)
4. Adição e Subtração (mesma precedência)

Notação Científica

Notação científica em Python usa a letra `e` como sendo a potência de 10. Também pode ser usada a letra `E` maiúscula:

```
>>> 10e6  
10000000.0  
>>> 1e6  
1000000.0  
>>> 1e-5  
1e-05  
>>> 1E6  
1000000.0
```

Comentários

Caso precise explicar alguma coisa feita no código, é possível escrever um texto (que não será executado), que ajuda a entender ou lembrar o que foi feito. Esse texto é chamado de comentário, e para escrever um basta utilizar o caractere `#`. Exemplo:

```
>>> 3 + 4 # será lido apenas o cálculo, da # para frente o  
interpretado python irá ignorar  
7  
>>> #aqui vai um código só com comentários! Seja voluntário  
IEEE, procure o ramo estudantil da sua universidade. Partic  
ipe de atividades técnicas, atividades sociais e faça muito  
networking com estudantes e profissionais de engenharia do  
mundo todo.
```

Comparações

Os operadores de comparação em Python são:

Operação	Significado
<	menor que
<=	menor igual que
>	maior que
>=	maior igual que
==	igual
!=	diferente

```

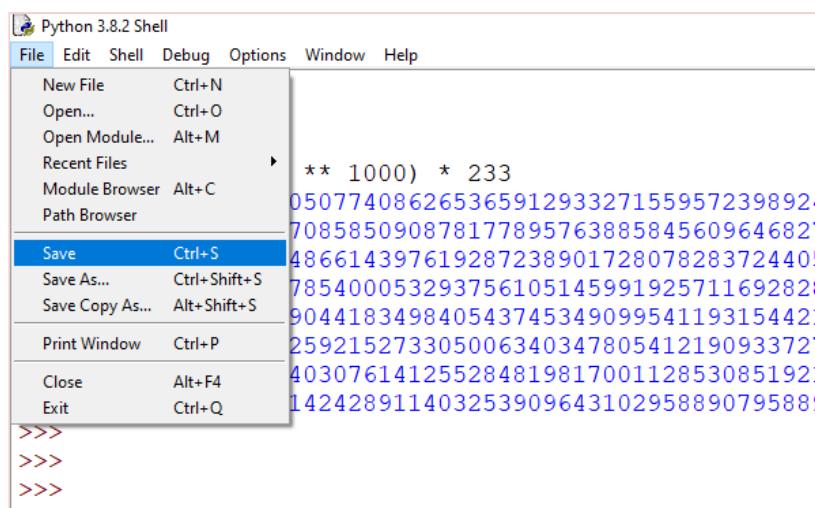
>>> 2 < 10
True
>>> 2 > 11
False
>>> 10 > 10
False
>>> 10 >= 10
True
>>> 42 == 25
False

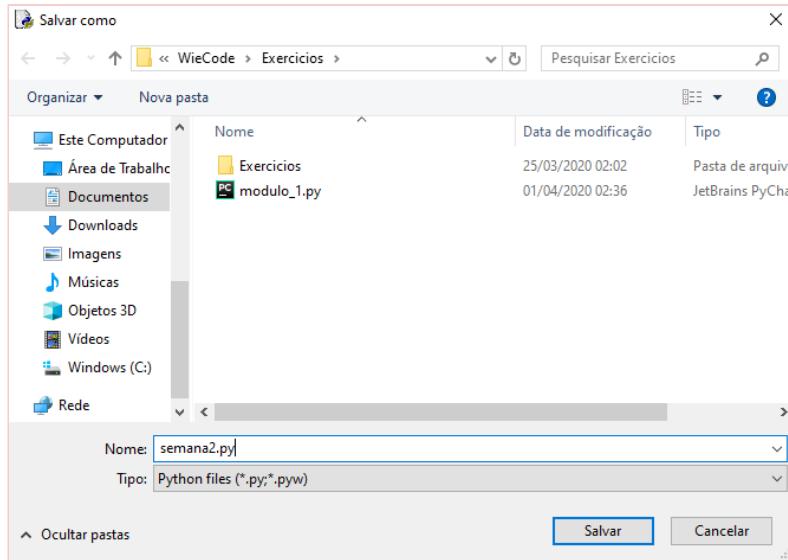
```

Agora, como sabemos (esperamos) que você executou todos os códigos do módulo 1, vamos salvá-los.

Salve seu código em um arquivo **.py**. Como fazer isso?

Vá em arquivo (file) no seu ambiente de desenvolvimento e salve seu arquivo como **semana2.py**.





Guarde para lembrar do seu primeiro código em python. =)

Agora que acabamos o conteúdo da Semana #2, vá até o repositório **Exercícios** no GitHub do WIECODE e acesse a pasta **exercício 2**. Abra uma nova aba ou um novo arquivo na sua IDE para fazer as questões deste módulo.

Quando terminar, guarde seus códigos em um arquivo chamado **exercicio_semana2.py** da forma que ensinamos anteriormente.

E no mesmo formato da Semana #1 (veja a apostila), coloque em uma pasta com seu nome e faça o upload do arquivo no repositório da web e dê um pull request no repositório do WieCode.

Bons Estudos!