

## Laboratório – Programação Python básica

### Objetivos

**Parte 1: Iniciar o VirtualBox e inserir a VM I2IoT Server**

**Parte 2: Fundamentos do Python**

**Parte 3: IDLE for Python**

### Histórico

Python, uma linguagem de programação, permite declarações mais simples. O Python é muito fácil de usar, poderoso e versátil. Essa linguagem se tornou a linguagem de escolha para muitos desenvolvedores de IoT. Um dos principais motivos para a popularidade do Python é a comunidade de desenvolvedores. Os desenvolvedores de Python criaram e disponibilizaram muitos módulos específicos que podem ser importados em qualquer programa para dar a ele, imediatamente, funcionalidades adicionais.

### Cenário

Neste laboratório, você irá aprender e praticar um pouco de programação básica do Python. Mais especificamente, usaremos o Python versão 3 no laboratório.

### Recursos necessários

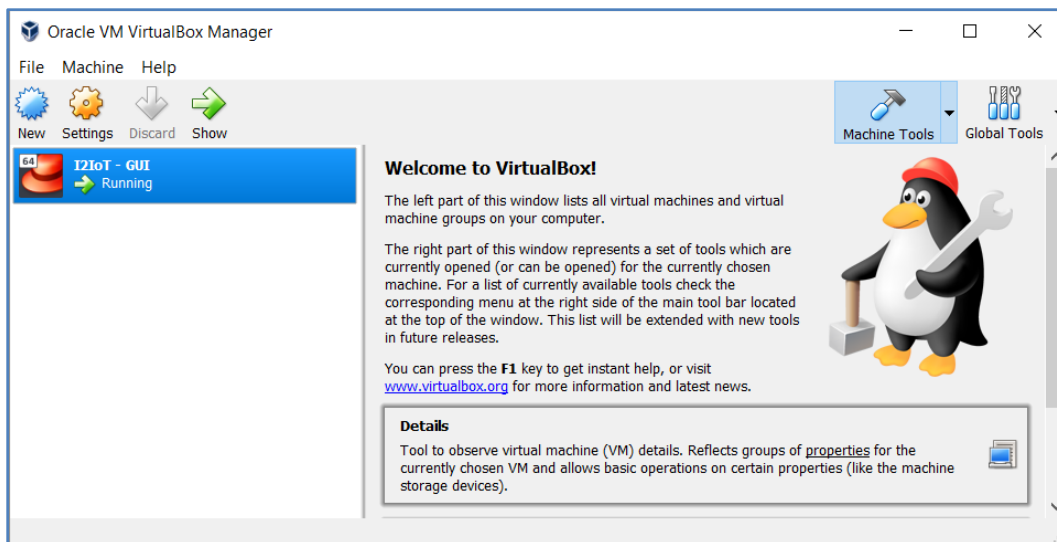
- Um computador pessoal moderno com acesso à internet e RAM suficiente.
- VirtualBox com I2IoT Server instalado.

## Parte 1: Inicie o VirtualBox e entre na VM I2IoT Server

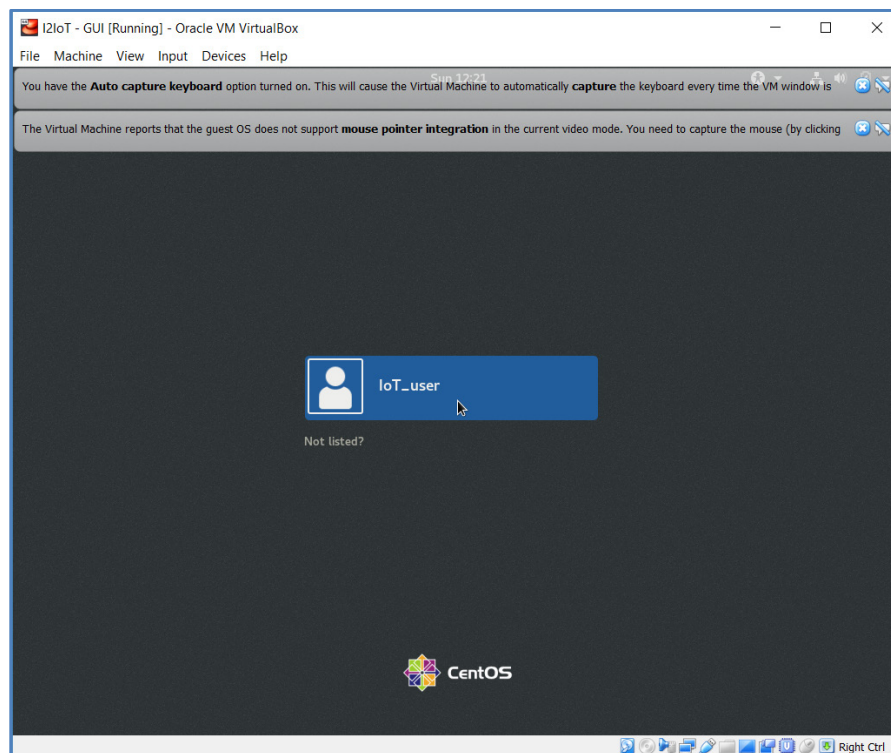
Na Parte 1, você inicia o software de virtualização VirtualBox e faz login na VM I2IoT Server.

### Etapa 1: Inicie o VirtualBox.

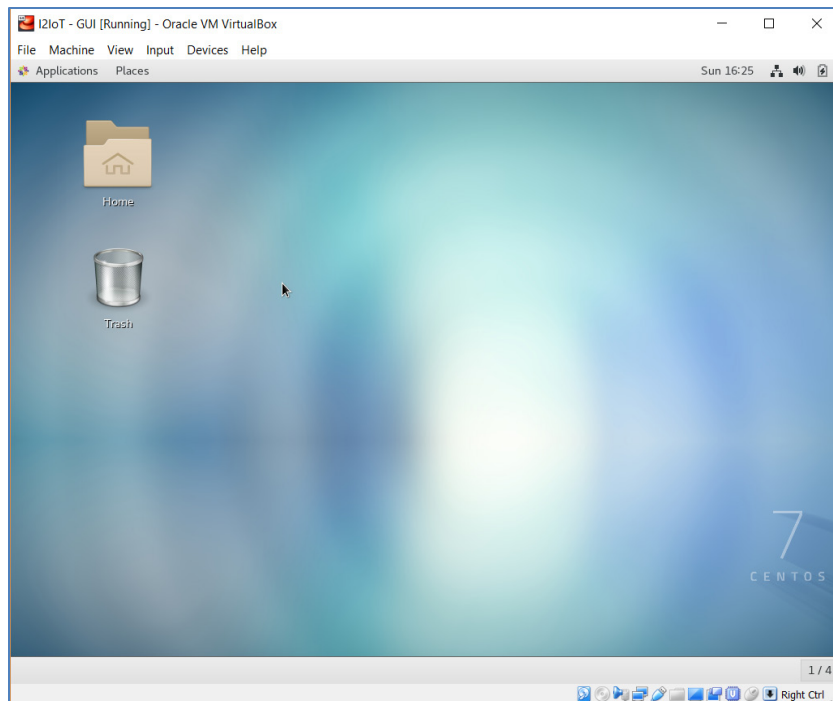
- a. Depois da instalação do VirtualBox (veja o laboratório 2.1.2.a), o ícone do VirtualBox deverá aparecer no ambiente de trabalho. Clique no ícone para iniciar o VirtualBox.



- b. Clique em **I2IoT – GUI** no painel à esquerda para iniciar a VM.

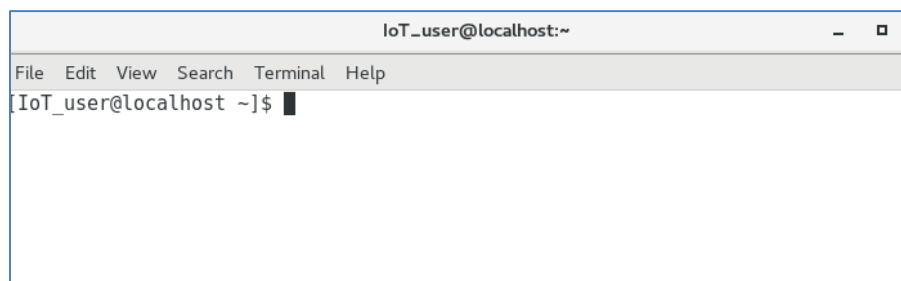


- c. O nome de usuário padrão é **IoT\_user**, sem senha. Clique na barra azul **IoT\_user** no meio da tela para fazer login na VM.

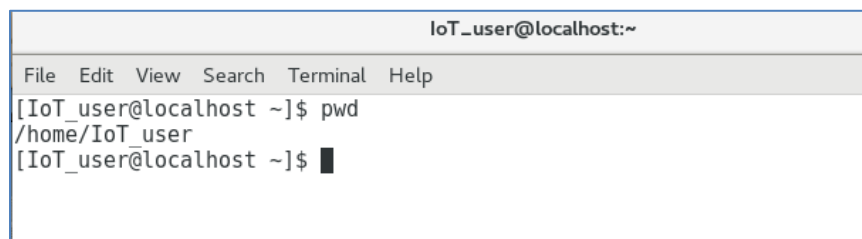


### Etapa 2: Navegue até o diretório do Documentos do usuário

- a. Para acessar a command line interface, clique em **Aplicativo** na barra de menu e escolha **Terminal**.



- b. Use o comando **pwd** para exibir o diretório atual.



- c. Use o comando **ls** para mostrar a lista de conteúdo no diretório atual. Use o comando **ls** com a opção **-l** para exibir informações detalhadas sobre o conteúdo.

```
IoT_user@localhost:~  
File Edit View Search Terminal Help  
[IoT_user@localhost ~]$ pwd  
/home/IoT_user  
[IoT_user@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[IoT_user@localhost ~]$ ls -l  
total 0  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Desktop  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Documents  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 15:24 Downloads  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Music  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Pictures  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Public  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Templates  
drwxr-xr-x. 2 IoT_user IoT_user 6 Apr 16 10:55 Videos
```

- d. Use o comando **cd Documents** para mudar o diretório para o diretório /home/IoT\_user/Documents. Verifique usando o comando **pwd**.

```
IoT_user@localhost:~/Documents  
File Edit View Search Terminal Help  
[IoT_user@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[IoT_user@localhost ~]$ cd Documents  
[IoT_user@localhost Documents]$ pwd  
/home/IoT_user/Documents  
[IoT_user@localhost Documents]$
```

- e. Para verificar a versão do Python instalado na VM, digite o comando **python3 --version**.

```
IoT_user@localhost:~/Documents  
File Edit View Search Terminal Help  
[IoT_user@localhost ~]$ cd Documents/  
[IoT_user@localhost Documents]$ python3 --version  
Python 3.6.5  
[IoT_user@localhost Documents]$
```

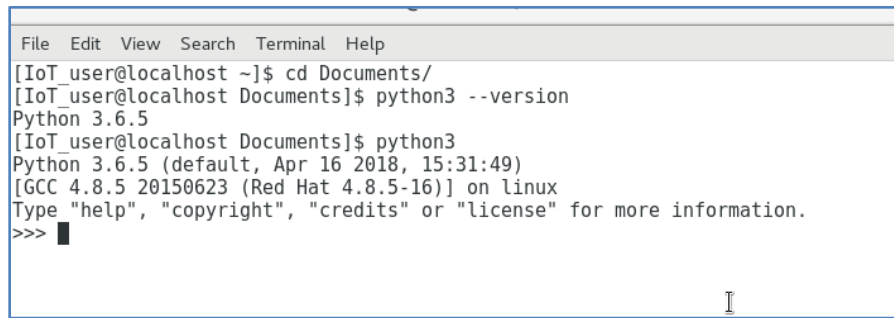
## Parte 2: Fundamentos do Python

Na Parte 2, você aprenderá e praticará um pouco de programação básica do Python.

### Etapa 1: Programação em Python no interpretador interativo.

Como uma linguagem interpretada, os comandos do Python podem ser emitidos em um interpretador interativo.

- a. Use o comando **python3** para iniciar o interpretador do Python.



```
File Edit View Search Terminal Help
[IoT_user@localhost ~]$ cd Documents/
[IoT_user@localhost Documents]$ python3 --version
Python 3.6.5
[IoT_user@localhost Documents]$ python3
Python 3.6.5 (default, Apr 16 2018, 15:31:49)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-16)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

- b. Realize cálculos.

```
>>> 1 + 2
3
>>> 2 * 4
8
>>> 6 / 2
3.0
>>>
```

- c. Imprima uma string de texto.

```
>>> "How are you?"
How are you?"
>>>
```

- d. Use o comando **type()** para determinar o tipo de dados básicos: int, flutuante, string, booleano.

```
>>> type(65)
<class 'int'>
>>> type(45,6)
<class 'float'>
>>> type("Hi!")
<class 'str'>
>>> type(True)
<class 'bool'>
>>> 1<2
Verdadeiro
>>> 1<1
Falso
>>> 1==1
Verdadeiro
>>> 1>=1
Verdadeiro
>>>
```

- e. Crie uma variável.

```
>>> x=3
>>> x*5
15
>>> "Good!"*x
'Good!Good!Good! '
>>>
```

- f. Combine várias strings e imprima como uma string única.

```
>>> str1="Cisco"
>>> str2="Networking"
>>> str3="Academy"
>>> space=" "
>>> print(str1+space+str2+space+str3)
Cisco Networking Academy
>>>
```

- g. Converta o tipo de dados de um número para uma string.

```
>>> x=5
>>> str(x)
>>> '5'
>>> y=4.2
>>> str(y)
>>> y='4.2'
>>>
```

- h. Observe que inteiros não são arredondados para cima ao serem convertidos de flutuantes. O decimal é ignorado.

```
>>> int(8.21)
8
>>> int(8.99)
8
>>> int(8.21) + int(8.99)
16
>>>
```

- i. Converta um número inteiro em flutuante.

```
>>> x=5
>>> x
5
>>> float(x)
5.0
>>> type(x)
<class 'int'>
>>> x=float(x)
>>> type(x)
<class 'float'>
>>> x
5.0
>>>
```

- j. Obtenha opinião do usuário.

```
>>> name=input("What is your name? ")
Qual é seu nome? John
>>> print("Hi " + name + ", it is nice to meet you!")
Olá, John, é bom vê-lo!
>>>
```

- k. Use **quit()** para sair interpretador interativo.

## Parte 3: IDLE for Python

IDLE significa Integrated Development and Learning Environment (Desenvolvimento e ambiente de aprendizado integrados). Ele é suportado e incluído no pacote Python. Alguns dos principais recursos do IDLE for Python incluem:

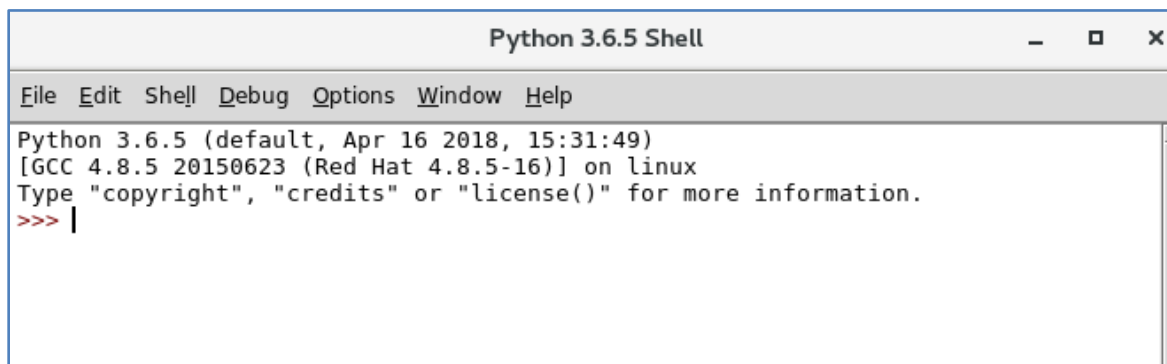
- Uma janela de shell do Python (interpretador interativo) com entrada e saída codificada por cores e com mensagens de erro
- Um editor de texto de várias janelas com vários recursos: Desfazer, codificação por cores do Python, recuo inteligente, dicas de chamada, preenchimento automático e outros recursos
- A capacidade de pesquisar em qualquer janela, substituir dentro de janelas do editor e pesquisar em vários arquivos (grep)
- Um depurador com pontos de interrupção persistentes, revisão e visualização de espaços de nomes globais e locais
- Configuração, navegadores e outras caixas de diálogo

Na Parte 3, você iniciará o IDLE e criará um script simples.

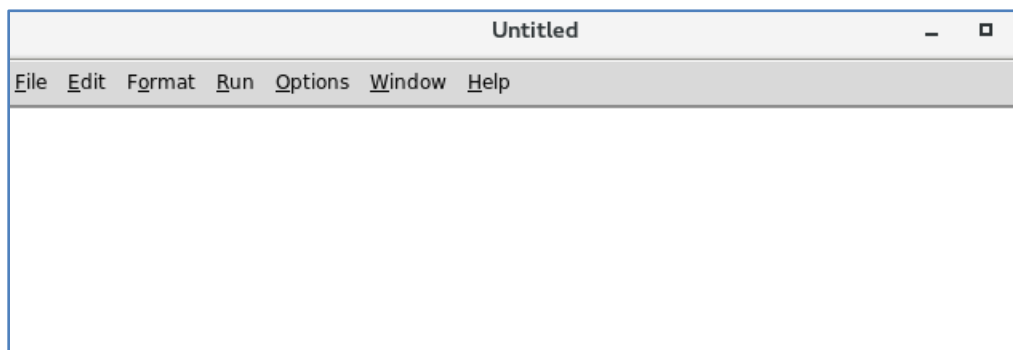
### Etapas 1: Inicie o IDLE.

- a. Use o comando **idle3** para iniciar o IDLE. Por padrão, ele começa na janela Shell do Python ou do interpretador interativo. Você já está familiarizado com o interpretador interativo.

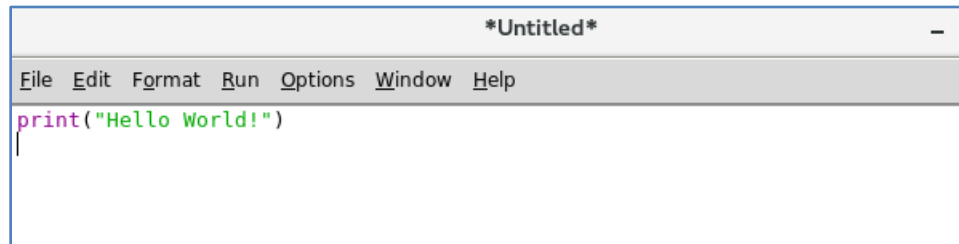
```
[IoT_user@stueverj-vm2 Documents]$ idle3
```



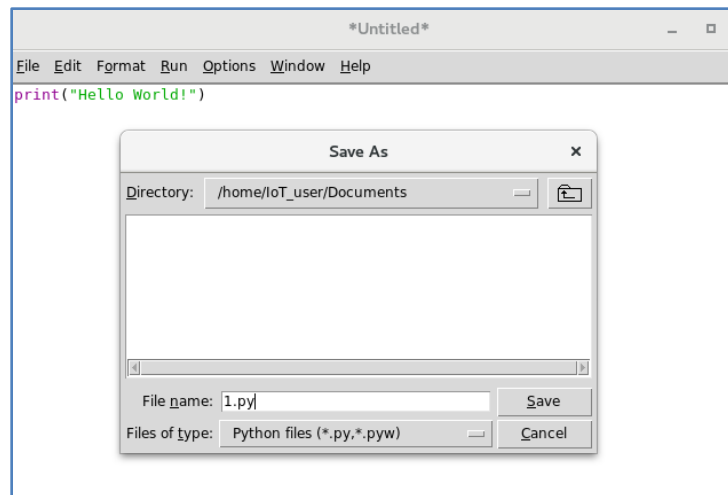
- b. Clique em **Arquivo** -> **Novo arquivo** para abrir um novo script Python (sem título).



- c. Digite o código no script de impressão `print("Hello World!")`, observe se os códigos são coloridos e se o parêntesis inicial e final correspondem.



- d. Clique em **Arquivo** -> **Salvar**, salve o script atual como 1.py no diretório atual. Clique no botão **Salvar**.



- e. Clique em **Executar** -> **Executar módulo** (ou pressione F5). A janela de shell exibirá o resultado.

