Bem-vindo ao roteiro de Introdução ao Python - Parte 1

*Confeccionado por Marcílio F. Oliveira Neto (marciliofoneto@gmail.com)
Github: github.com/marcilioliveira

Neste roteiro você irá aprender sobre a sintaxe do Python. Este roteiro tem o intuito de seguir o conceito *do-it-yourself*.

Neste roteiro você encontrará os seguintes tópicos sobre Python:

- Olá, mundo!
- Comando Print
- Comando Input
- Variáveis
- Tipo string
- Tipo int
- Tipo float
- ...

Instruções para utilizar este roteiro:

- Importar o arquivo deste roteiro no Jupyter;
- Clicar no menu superior **Cell**;
- Clicar na opção **Run All**;
- Para executar célular individualmente utilizar:
  - Selecionar a célula
  - Clicar no botão **Run** - menu superior - ou;
  - Clicar na opção **Cell** e **Run Cells** - ou;
  - Utilizar o atalho **Ctrl + Enter**.

Seu primeiro programa - Olá, mundo!

Programar consiste em designar (codificar) tarefas para que um computador execute-as. Acredite, em algum momento da sua vida de engenheiro você precisará fazer isso.

Comando print

Sendo assim, é importante que o computador consiga nos dizer algo. Em Python, você consegue fazer isso utilizando o comando **print**.

In [2]:

```
print("Meu primeiro comando - print")
```

```
Meu primeiro comando - print
```

Como visto acima, o comando (*)**print** permite que o computador exiba variados textos na tela.

(*)Aqui cabe um ponto de atenção, você deve seguir o template acima para exibir suas frases, ou seja, utilizando aspas.

Do-it-yourself

Tente executar seu primeiro comando no quadro abaixo. Utilize o **print** e exiba uma frase que lhe agrade.

Para executar o comando, você pode utilizar o botão - menu superior - **Run** ou utilizar o atalho **Ctrl + Enter**. Veja a saída no próprio quadro.

Variações do print

O comando **print** permite algumas variações para que a apresentação dos textos seja confortável. Além disso, permite exibir várias frases utilizando apenas uma única acerção. Veja abaixo alguns exemplos.

In [3]:

```
print("Este", "comando", "exibe", "uma", "frase", "com", "espaços!")
```

```
Este comando exibe uma frase com espaços!
```

In [4]:

```
print("Este" + "comando" + "exibe" + "uma" + "frase" + "sem" + "espaços!")
```

```
Estecomandoexibeumafrasesemespaços!
```

Note que utilizando a **vírgula(,)** podemos separar palavras sem o uso explícito do espaço. Mantenha isso em mente, isso o ajudará nos próximos passos.

O sinal de **soma(+)** é capaz de concatenar as palavras. Mantenha isso em mente, concatenação é algo muito utilizado na programação.

Do-it-yourself

Chegou a hora do ***do-it-yourself***. Utilize os quadros abaixo e faça alguns testes com as variações demonstradas acima.

Variáveis

Precisamos guardar o que sabemos em algum lugar, certo?! Para o computador não é diferente, ele possui memória, e para cada informação guardada na memória do computador precisamos definir um lugar - espaço - correto, esse espaço é denominado **variável**.

Variáveis são espaços (endereços) na memória do computador capazes de registrar a informação passada a ele.

Veja como é fácil criar variáveis capazes de guardar números em Python:

In [5]:

```
a = 10 #Atribui o valor numérico 10 a variavel a
```

In [6]:

```
b = 20 #Atribui o valor numérico 20 a variavel b
```

In [7]:

```
c = 1.5 #Atribui o valor numérico 1.5 a variavel c
```

Nos quadros acima foram criadas duas variáveis, **a** e **b**. Note que os nomes **a** e **b** foram escolhidos aleatoriamente, você poderá dar o nome que melhor convém. Porém, deve-se respeitar alguns parâmetros:

- Não começar com números;
- Não utilizar caracteres especiais;
- Não possuir espaços;
- Não utilizar palavras reservadas do Python (comandos).

As variáveis também podem guardar palavras, frases, textos em geral. Veja como é fácil, basta utilizar as aspas.

In [8]:

```
fruta = "maça" #Atribui o valor textual "maça" a variavel fruta
```

In [9]:

```
pessoa = "João Carlos" #Atribui o valor textual "João Carlos" a variavel pessoa
```

In [10]:

```
nome_completo = "Maria Silva de Bragança" #Atribui o valor textual "Maria Silva
 de Bragança" a variavel
                                          #nome_completo
```

In [11]:

```
sinopse_vingadores = "Os Vingadores são um grupo de super-heróis de história em
 quadrinhos publicados nos Estados\
                      Unidos pela editora Marvel Comics. O grupo também aparece
 em adaptações da Marvel para\
                      cinema, desenho animado e videogames."
                      #Atribui o texto da sinopse de vingadores a variavel sinop
se_vingadores
```

Atribuição

Um conceito importante visto nos quadros acima é o de atribuição. Ao definirmos variáveis, já podemos passar (atribuir) valores à elas, isso é feito pelo **símbolo de igual (=)**.

Nos próximos quadros, você conhecerá o comando **input** e, atente-se que o sinal de atribuição é utilizado também. O comando **input** recebe um valor arbitrário e atribui esse valor em uma variável.

Comando print e variáveis

Agora você já sabe que o computador é capaz de falar e guardar informações na memória. Assim, nada mais justo do que fazer com que ele exiba as informações contidas na memória para nós.

Veja como é simples fazer essa tarefa em Python. Utilizaremos as variáveis criadas nos quadros acima para esses exemplos.

In [12]:

```
print(a)
```

```
10
```

In [13]:

```
print(b)
```

```
20
```

In [14]:

```
print(c)
```

```
1.5
```

In [15]:

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

```
10 20 1.5
```

Note que agora podemos resgatar informações contidas na memória do computador e mostrá-las ao usuário, bastando para isso, utilizar o comando **print** junto com os nomes definidos para cada variável. Bacana, né?!

Do-it-yourself

Chegou a hora do **do-it-yourself**. Exiba as informações das variáveis: **fruta, pessoa, nome_completo e sinopse_vingadores**.

Utilize as variações do comando **print**, lembre-se das vírgulas e do sinal de soma.

Insira seus testes nos quadros abaixo, execute-os utilizando o botão **Run** ou o atalho **Ctrl + Enter**.

Exiba a informação (conteúdo) da variável **fruta**:

Exiba a informação (conteúdo) da variável **pessoa**:

Exiba a informação (conteúdo) da variável **nome_completo**:

Exiba a informação (conteúdo) da variável **sinopse_vingadores**:

Utilize o próximo quadro para fazer testes com o comando **print** e suas variáveis com **vírgula(,)** e o sinal de **soma(+)**:

Comando input

Você já sabe que o computador é capaz de guardar informações em memória e falar (exibir) essas informações.

Agora é hora de saber como fazer ele te ouvir. No dia-a-dia é comum que você passe informações ao computador, e em Python isso é simples, basta utilizar o comando **input**. Tenha em mente que essas informações deverão ser armazenadas em memória, ou seja, precisarão de **variáveis**.

Veja abaixo um exemplo simples do comando **input**:

```
In [16]:
```

```
informacao = input("Digite uma informação: ")
```

```
Digite uma informação: teste
```

Note, no quadro acima, que o comando **input** depende de uma variável, chamada **informacao**.

Além disso, é necessário utilizar o sinal de igualdade** **(=)** para que o dado fornecido pelo usuário, no caso a frase: *{{informacao}}*, seja guardada na memória do computador.

Podemos comprovar que a informação: ***{{informacao}}*** está na memória do computador (variável informacao) através do comando **print**. Veja como é simples:

In [17]:

```
print(informacao)
```

```
teste
```

Veja um exemplo de informação numérica:

In [18]:

```
nome = input("Digite seu nome: ")
idade = input("Digite sua idade: ")
```

```
Digite seu nome: joao
Digite sua idade: 12
```

Podemos comprovar que as informações inseridas pelo usuário: Nome: ***{{nome}}*** e Idade: ***{{idade}}*** está na memória do computador, exibindo o conteúdo da variável **nome** e **idade**.

In [19]:

```
print("Nome digitado:", nome)
print("Idade", idade)
```

```
Nome digitado: joao
Idade 12
```

****Sinal de atribuição de valores (=)**

Lembre-se, sempre que utilizar o comando **input** é necessário associar uma variável a ele, responsável por guardar a informação inserida. Essa associação é feita pelo comando de atribuição (=).

Do-it-yourself

Chegou a hora de praticar, utilizando os comandos **print**, **input**, **variáveis** e **de atribuição (=)**, utilize os quadros abaixo para realizar perguntas simples ao usuário e exibi-las na tela. Seja criativo, um simples formulário pedindo informações de Nome, RA e curso é uma boa pedida! :)

Utilize o quadro abaixo para realizar os inputs das informações. Você pode digitar mais de um comando no mesmo quadro

Utilize o quadro abaixo para exibir as informações contidas nas variáveis que você definiu.

*Lembre-se, para executar os comandos você pode utilizar o botão **Run** ou o atalho **Ctrl + Enter**.

Comando input com números

Você já entendeu que o comando **input** é poderoso e pode receber qualquer informação, não é mesmo?! Porém, é importante dizer que **toda e qualquer informação deste comando é textual (string)**, ou seja, caso o usuário digite um número, não seremos capazes de somá-los, por exemplo.

A pergunta é, como faço pra somar números digitados?! Simples...
Devemos converter o **input** para o formato numérico, que a princípio são dois: **int** e **float**.

Tipo int

Quando você desejar trabalhar com números inteiros, você deve realizar a seguinte conversão:

In [20]:

```
numero_um = input("Digite um numero inteiro ")
numero_um = int(numero_um)
```

```
Digite um numero inteiro 12
```

Até o momento, nada diferente, tirando o fato que convertemos a variável **numero_um** para **int**. Mas agora podemos somar outros números à essa variável. Veja abaixo:

In [21]:

```
numero_um = numero_um + 8
print(numero_um) #Imprime (mostra) o valor contido na variavel numero_um
```

```
20
```

Notou que bacana?! A variável que continha o número informado pelo usuário teve seu valor (conteúdo) incrementado em 8 (oito).

Agora podemos somar dois ou mais números informados pelo usuário, saca só como é simples! :D

In [22]:

```
digito_1 = input("Digite o número 1: ") #Atribui o valor digitado pelo usuário a
variavel digito_1
digito_1 = int(digito_1)
digito_2 = input("Digite o número 2: ") #Atribui o valor digitado pelo usuário a
variavel digito_2
digito_2 = int(digito_2)
soma_digitos = digito_1 + digito_2
```

```
Digite o número 1: 1
Digite o número 2: 44
```

Para recuperarmos a soma e exibi-la na tela é fácil, basta utilizar o comando **print**:

In [23]:

```
print("A soma é:", soma_digitos) #Imprime (mostra) o texto "A soma é" seguido do valor contido na variável
                                 #soma_digitos
```

```
A soma é: 45
```

Tipo float

Quando você desejar trabalhar com números fracionados - podendo **ter ou não casas decimais** -, você deve realizar a seguinte conversão:

In [24]:

```
digito_3 = input("Digite um número fracionado: ")
digito_3 = float(digito_3) #Converte o valor textual da variavel digito_3 para numérico decimal
digito_4 = input("Digite um número fracionado: ")
digito_4 = float(digito_4) #Converte o valor textual da variavel digito_4 para numérico decimal
soma_fracionada = digito_3 + digito_4
```

```
Digite um número fracionado: 0.5
Digite um número fracionado: 1.3
```

In [25]:

```
print(soma_fracionada)
```

```
1.8
```

Tipos **int** e **float** são extremamente úteis no dia-a-dia da nossa programação, matenha-os em mente! ;)

Concatenação de texto

Se você não realizar essa conversão, você terá um resultado parecido com o abaixo:

In [26]:

```
dig_um = input("Digite um número: ")
dig_dois = input("Digite outro número: ")
soma_digs = dig_um + dig_dois #Como os valores das variaveis são textuais (strings)
                              #os valores são concatenados (unidos)
```

```
Digite um número: 13
Digite outro número: 98
```

In [27]:

```
print("A soma foi:", soma_digs)
```

```
A soma foi: 1398
```

Note que as informações inseridas foram unidas - isso é concatenar -, então, se você não converter o dado informado para o tipo correto, você poderá obter algo totalmente diferente.

Agora, ao menos você sabe como unir dois textos quaisquer quando solicitado! :D

Operadores aritméticos

Se você estiver atento, já sacou que foi utilizado o operador aritmético de soma. Existem outros:

- Divisão: /
- Subtração: -
- Múltiplicação: *
- Potenciação: **
- Resto da divisão: %
- ...

Podemos utilizar todos eles para trabalhar com variáveis numéricas, beleza?!

Do-it-yourself

Chegou a hora de praticar um pouco, chega de mimimi :P Utilize os quadros abaixo para isso.

Dessa vez, vai rolar até uma **proposta** do que você pode fazer. Faça um formulário capaz de realizar as seguintes ações:

- Ler o primeiro nome do usuário;
- Ler o sobrenome do usuário;
- Ler a idade em anos;
- Exibir o nome e sobrenome concatenados;
- Converter a idade em anos para dias;
- Exibir o resultado final.

Vou até te dar uma ajuda, saca só... No quadro abaixo, crie o código para ler o nome do usuário:

Agora leia o sobrenome:

A idade:

Utilize o quadro abaixo para manipular os dados, converter o que for necessário e fazer os cálculos:

Mostre o resultado final neste último e seja feliz! ;)

Parabéns! Você finalizou o roteiro 1!

Agora é hora de você praticar com exercícios.

Este roteiro foi inspirado em metodologias ativas de ensino.

Este conteúdo foi confeccionado pelo professor:

Marcílio Francisco de Oliveira Neto (marciliofoneto@gmail.com)