



# INFINITY SCHOOL

V I S U A L   A R T   C R E A T I V E   C E N T E R



# Lógica da programação - Aula 02

## Objetivos da aula:

1. Iniciação da lógica em Python.
2. Introdução ao VS Code.
3. Introdução ao Shell do Python.
4. Tipos de dados e entradas (variáveis).
5. Operadores aritméticos, relacionais e lógicos.
6. Condicionais simples e compostas.

```
0 response = requests.get(url)
1
2 # checking response.status_code (if you get 502, try removing the code)
3 if response.status_code != 200:
4     print(f"Status: {response.status_code} - Try removing the code")
5 else:
6     print(f"Status: {response.status_code}\n")
7
8 # using BeautifulSoup to parse the response object
9 soup = BeautifulSoup(response.content, "html.parser")
10
11 # finding Post images in the soup
12 images = soup.find_all("img", attrs={"alt": "Post image"})
13
14 # downloading images
15 for i in range(len(images)):
16     image = images[i]
```

# Lógica da programação

## ■ Introdução ao VS Code

- O VS Code é um editor de texto que foi lançado em 2015 pela Microsoft.
- É totalmente gratuito e também open-source, ou seja, todo o seu código fonte é aberto e está disponível.
- É uma ferramenta multiplataforma e está disponível para os sistemas operacionais Mac, Linux e Windows.
- Tem suporte às mais populares linguagens, como Python, Javascript, Java, PHP, HTML, CSS e várias outras.



# Lógica da programação

## ■ Extensões do Python para VS Code

- **Python for Visual Studio Code**

É uma extensão com suporte avançado para a linguagem Python incluindo diversas funcionalidades.

- **Tabnine**

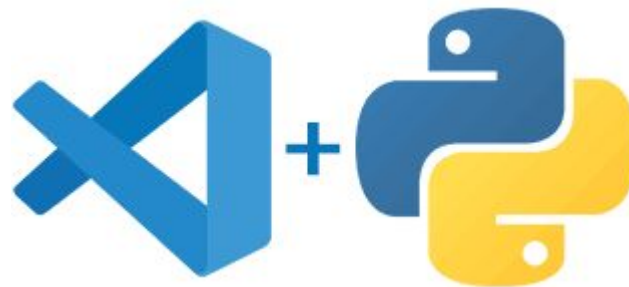
Tabnine é um poderoso assistente de Inteligência Artificial projetado para te ajudar a codificar muito mais rápido.

- **Python Preview**

Essa extensão tem como objetivo de mostrar o estado dos dados que você está trabalhando no Python de forma gráfica.

- **Bracket Pair Colorizer**

O funcionamento dessa extensão é muito simples porém ajuda bastante na hora de visualizar o nosso código.



## ■ Link para baixar o python

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

## ■ O que é Python?

- Uma das principais características da linguagem Python é a facilidade de leitura do código escrito. A sua simplicidade facilita o aprendizado da programação.
- Python é uma linguagem open-source, usado bastante em data science, machine learning, desenvolvimento web, desenvolvimento de aplicativos, automação de scripts, fintechs e mais.
- Python é um software livre, ou seja, permite que usuários e colaboradores possam modificar seu código fonte e compartilhar essas novas atualizações.



## ■ O que é Python?

- **Linguagem multiparadigma**

O fato de Python ser considerada multiparadigma é a possibilidade de programar em vários paradigmas, como: procedural, funcional ou orientado a objetos.

- **De alto nível**

As linguagens de programação de alto nível são aquelas que, grosso modo, estão mais próximas da linguagem humana do que da linguagem de máquina.

- **Interpretada**

Linguagem interpretada é uma linguagem de programação em que o código fonte nessa linguagem é executado por um programa de computador chamado interpretador, que em seguida é executado pelo sistema operacional ou processador.

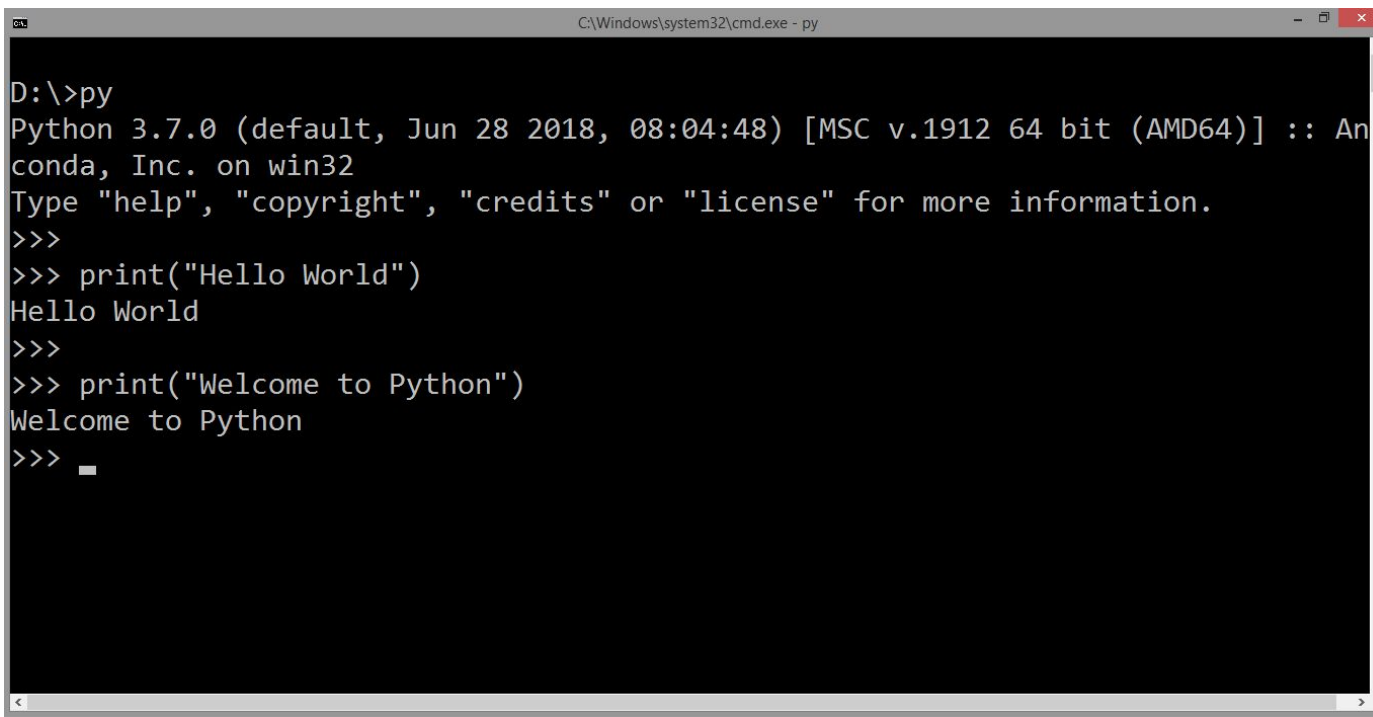
- **De tipagem dinâmica**

De tipagem dinâmica: Nestes casos, a linguagem observa qual é o tipo de dado que está sendo atribuído a cada variável e, a partir daí, determina qual é o tipo daquela determinada variável.



# Lógica da programação - Aula 02

## Shell do Python:



```
C:\Windows\system32\cmd.exe - py
D:\>py
Python 3.7.0 (default, Jun 28 2018, 08:04:48) [MSC v.1912 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> print("Hello World")
Hello World
>>>
>>> print("Welcome to Python")
Welcome to Python
>>> █
```



## ■ Shell do Python:

- A Python Shell é o interpretador que executa os seus programas Python, outras peças do código Python ou comandos simples.
- Neste interpretador você pode executar qualquer comando ou expressão Python.
- O Shell é uma ótima solução para todo programador, e por isso é uma boa ideia adquirir o hábito de usá-lo.
- Se existir alguma dúvida sobre como algo funciona no Python, o Shell pode esclarecer para você.





## ■ Criando seu ambiente virtual no shell:

Todo programador python deve saber isolar seu ambiente de trabalho pois em muitos casos somos alocados em projetos diferentes onde usamos, em alguns casos, um determinado pacote em 2 ou mais projetos, porém em versões distintas a depender de cada cliente.

Para nos ajudar a resolver essa situação, o python disponibiliza/permite a criação de ambientes virtuais onde conseguimos isolar um determinado ambiente em nossa máquina.

Há diversos pacotes, mas aqui no curso utilizaremos o módulo **venv**, que já vem por padrão no python.



# Lógica da programação - Aula 02

## ■ Criando seu ambiente virtual no shell:

Para ativar o ambiente virtual, primeiro devemos entrar em nossa pasta de trabalho.

Dentro dela, digite o comando a seguir: **python -m venv <nome do ambiente virtual>**

Ex: python -m venv IN

**ATENÇÃO:** nenhuma mensagem aparecerá no terminal, o que significa que o comando rodou com sucesso. Quando o sistema operacional devolver o prompt de comando pra você sem alguma mensagem, significa que o comando rodou com sucesso.



# Lógica da programação - Aula 02

## ■ Criando seu ambiente virtual no shell:

Agora é hora de ativar seu ambiente virtual.

Em máquinas **Windows** digite o comando: **IN \ Scripts \ activate**

Em máquinas **Linux** digite o comando: **source IN/bin/activate**

Repare que o nome do seu ambiente virtual aparecerá entre parêntesis informando que o mesmo está ativo.

Agora poderá instalar os pacotes que irá trabalhar em cada projeto.

Para isso, utilize o gerenciador de pacotes do python: pip



# Lógica da programação - Aula 02

## ■ Criando seu ambiente virtual no shell:

**Para instalar** um pacote no seu ambiente virtual ativo, digite o comando **pip install** e o nome do pacote

Obs: para instalar uma versão específica: `pip install pacote==0.0.0`  
(substitua pacote pelo nome do pacote desejado e 0.0.0 pela versão a ser instalada. Se não informar a versão, a mais atual será instalada)

**Para desinstalar:** `pip uninstall pacote`

Os pacotes são instalados do repositório [www.pipy.org](http://www.pipy.org)

**Para desativar** o ambiente virtual, basta digitar o comando `deactivate` em ambos os sistemas operacionais.



## ■ Criando seu ambiente virtual no shell:

O comando **pip freeze** listará todos os pacotes com suas versões instaladas no seu ambiente virtual ativado.

Para reproduzir este ambiente de trabalho em outra máquina ou para enviar a um outro desenvolvedor, execute os seguintes passos:

1. Com seu ambiente virtual ativado, digite o comando: `pip freeze > requirements.txt`

Isso fará com que a saída do `pip freeze` vá para o arquivo `requirements.txt`. o nome desse arquivo é uma convenção mas você pode utilizar o nome que desejar.



## ■ Criando seu ambiente virtual no shell:

2. Compartilhe o arquivo com outro desenvolvedor ou leve-o para outra máquina.

3. Na máquina nova, crie um novo ambiente virtual, ative-o e rode o comando: `pip install -r requirements.txt`

Isso instalará todos os pacotes instalados nas mesmas versões.

Os outros devs que receberem esse arquivo deverão executar os mesmos passos descritos acima.



# Lógica da programação - Aula 02

## Tipos de variáveis

- **int**  
Guardam números inteiros.  
Exemplo: 2, -3, 2008, -1987
- **float**  
Guardam números de ponto flutuante.  
Exemplo: 1.34, -56.9, 7.0
- **str**  
Guardam informações de texto.  
Exemplo: "s", "Olá", "G", "casa", "João"
- **bool** (True, False)  
São **variáveis** capazes de conter apenas 1 de 2 valores: verdadeiro ou falso (booleano).



# Lógica da programação - Aula 02

## Regras de variáveis

- Não comece sua variável com números.
- Não separe os nomes (ex: quantidade de pessoas).
- O único caractere especial permitido é o `_`.  
O python segue a convenção de nome snake case para nomes de variáveis e para isso recomenda-se o uso do `_`.
- Letras maiúsculas se diferenciam das minúsculas em linguagens CaseSensitive.





# Lógica da programação - Aula 02

## Operadores aritméticos

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

+	soma	$3+2=5$
-	subtração	$3-2=1$
*	multiplicação	$3*2=6$
/	divisão	$3/2=1.5$
%	módulo	$3\%2=1$
**	exponenciação	$3**2=9$
//	floor division	$3//2=1$

## Operadores relacionais

- == igual
- != diferente
- < menor que
- > maior que
- >= maior igual a
- <= menor igual a

# Lógica da programação - Aula 02

## Operadores lógicos:

Esses Operadores nos possibilitam construir um tipo de teste muito útil e muito utilizado em qualquer programa Python: os **testes lógicos**.

- **and**

Retorna True se ambas as afirmações forem verdadeiras.

- **or**

Retorna True se uma das afirmações for verdadeira.

- **not**

Retorna False se o resultado for verdadeiro e vice-versa.

and

p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

or

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

not

p	$\sim p$
V	F
F	V

## ■ Tipos de entradas e saídas de dados:

- `input()`

Quando precisamos que o usuário passe ao programa algum tipo de dado.

Em Python, fazemos isso utilizando a função `input()`, que é literalmente 'entrada' em inglês.

- `print()`

A função para imprimir dados em Python é a função `print()`. Ela é responsável por mostrar valores em seu terminal.



# Lógica da programação - Aula 02

## ■ .format

**format()** é um dos métodos de formatação de string em Python. Este método permite várias substituições e formatação de valor, com ele é possível concatenar elementos em uma string por meio da formatação posicional.

Exemplo:

```
varnome = 'Asterix'
```

```
print(' É um prazer te conhecer, {}'.format(varnome))
```



# Lógica da programação - Aula 02

## Formatação com f-string:

*F-strings* foram criados para facilitar nossa vida.

Foi introduzida no python 3.6 e também é chamada de “strings literais formatadas” (*formatted string literals*).

F-strings são strings com a letra *f* no início e chaves `{}` para realizar a interpolação de expressões.

### Exemplo:

```
numero1 = int(input('digite o primeiro número: '))
```

```
numero2 = int(input('digite o segundo número: '))
```

```
soma = numero1 + numero2
```

```
print(f'A soma dos números que você digitou é {soma}.')
```



## ■ Chegou a hora de praticar

1. Faça um Programa que mostre a mensagem "Olá, mundo!" na tela.
1. Faça um Programa que peça dois números e imprima a soma.
1. Faça um Programa que peça as 4 notas bimestrais e mostre a média.



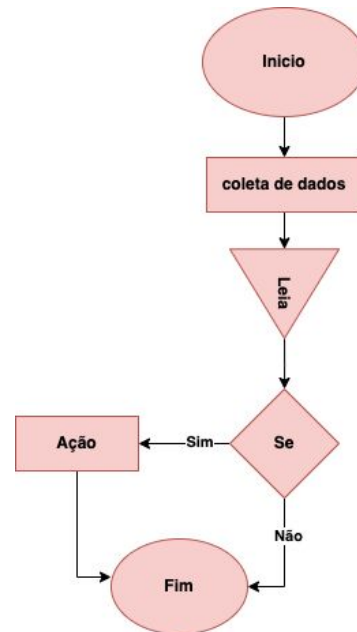
# Lógica da programação - Aula 02

## Condicionais:

- if
- elif (else if)
- else

A condicional if é uma **estrutura condicional** que executa a afirmação, dentro do bloco, se determinada condição for verdadeira. Se for falsa, executa as afirmações dentro de else.

A estrutura condicional permite a escolha do grupo de ações e **estruturas** que serão executadas, quando determinadas condições, representadas por expressões lógicas (verdadeiro ou falso), forem ou não satisfeitas.



## Estrutura condicional:

Uma estrutura condicional, como o próprio nome já diz é uma estrutura que vai analisar uma condição e baseado no resultado dessa condição é que vamos executar uma determinada ação.

# Lógica da programação - Aula 02

## Exemplo de estrutura condicional:

```
var1 = 4
```

```
if var1 == 4:  
    print("x é igual a 4")
```

```
else:  
    print("x é diferente de 4")
```





# Lógica da programação - Aula 02

## ■ Condicionais - exercício:

- Faça um Programa que peça dois números e imprima o maior deles.
- Faça um Programa que verifique se uma letra digitada é vogal ou consoante.
- Faça um Programa que leia três números e mostre o maior deles.



Você concluiu a aula 02 do seu módulo de  
Lógica da programação.  
Continue praticando e até a próxima aula!



INFINITY SCHOOL

V I S U A L   A R T   C R E A T I V E   C E N T E R