

Python para Ciência de Dados e Inteligência Artificial

Aula 02: Introdução ao Python.

IMT – Instituto Mauá de Tecnologia

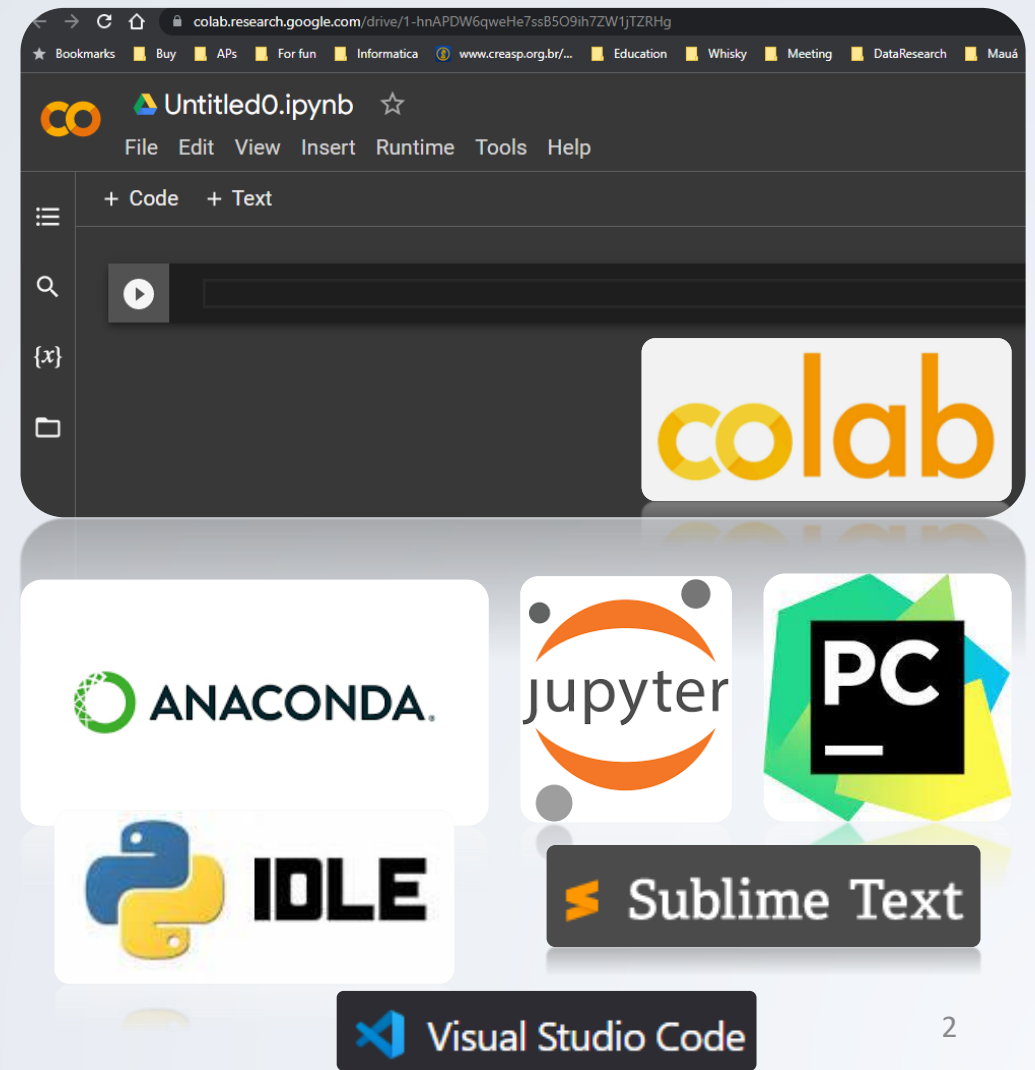
Fevereiro/2023



IDEs e Editores de código para Python

IDE é um ambiente de desenvolvimento que fornece muitos recursos como codificação, compilação, depuração, execução, autocompletar, bibliotecas, em um só lugar para o desenvolvedor.

O **editor de texto** é uma plataforma para editar e modificar apenas o código.





Exibindo uma mensagem na tela

- **Saída de dados via monitor:** Saídas de dados por meio de um monitor podem ser feitas pela função **print**.

```
[1] print("Salve, Mauá!")
```

```
Salve, Mauá!
```



Variáveis

- **Variáveis:** Uma variável armazena valores na memória do computador.

```
[2] mensagem = "Salve, Mauá"
    print(mensagem)

Salve, Mauá
```

O Python é sensível a caracteres
maiúsculos e minúsculos!

Cuidado!

```
[3] mensagem = "Salve, Mauá"
    print(Mensagem)

-----
NameError                                Traceback (most recent call last)
<ipython-input-3-d72021c6ba56> in <module>
      1 mensagem = "Salve, Mauá"
----> 2 print(Mensagem)

NameError: name 'Mensagem' is not defined

SEARCH STACK OVERFLOW
```



Tipos de Variáveis: Strings

- **Strings:** Strings é um tipo de dado que armazena um conjunto de caracteres, ou melhor, um texto.

```
▶ texto1 = "Exemplo 1 de strings"
  texto2 = 'Exemplo 2 de strings'

  print(texto1, texto2)

  texto3 = 'Atenção às "aspas"'
  print(texto3)
```

Exemplo 1 de strings Exemplo 2 de strings
Atenção às "aspas"



Tipos de Variáveis: Strings

- Modificando Strings:

```
▶ texto1 = "Exemplo de texto!"  
print(texto1.title())  
print(texto1.upper())  
print(texto1.lower())  
print(texto1.swapcase())
```

Exemplo De Texto!
EXEMPLO DE TEXTO!
exemplo de texto!
eXEMPLO DE TEXTO!

- Combinando Strings:

```
▶ # Combinando strings  
nome = "Jones"  
sobrenome = "Egydio"  
print(nome + sobrenome)
```

JonesEgydio

```
[10] # Combinando strings  
nome = "Jones"  
sobrenome = "Egydio"  
print(nome + " " + sobrenome)
```

Jones Egydio



Tipos de Variáveis: Números

- **Números:** O tipo de variável Número no Python é composto pelos números inteiros (**int**) reais (**float**) e complexos (**complex**).
- **Operações aritméticas:** Exemplos com números inteiros (**int**):

```
# Operações com números inteiros
print(3 + 2)
print(3 - 2)
print(3 * 2)
print(3 / 2)
print(3 // 2) # divisão inteira
print(5 // 2)
print(3 % 2) # resto da divisão
print(23 % 3)
print(3 ** 2)
```



Tipos de Variáveis: Números

- Precedência geral dos operadores aritméticos: de acordo com as regras das operações matemáticas aprendidas.



```
expressão1 = 6/2*1+2  
print(expressão1)
```

```
expressão2 = 6/2*(1+2)  
print(expressão2)
```

```
expressão3 = 6/(2*(1+2))  
print(expressão3)
```



```
5.0  
9.0  
1.0
```




Tipos de Variáveis: Números

- Números reais (float):

```
[15] # Exemplos envolvendo números reais
      expressão1 = 0.1 + 0.1
      print(expressão1)

      expressão2 = 0.1 + 0.2
      print(expressão2)

      expressão3 = 0.1 + 0.2 - 0.3
      print(expressão3)
```

0.2

0.30000000000000004

5.551115123125783e-17



Marcadores

- **Marcadores:** são caracteres que são substituídos por valores, os mais comuns são o %i (ou %d), %f e %s.

```
# Exemplos de substituição
x = 3.14159
y = 3.3
unid = "Hz"

print("O valor é", x, unid)
print(" ")
print("O valor é %f" %x)
print("O valor é %2f" %x)
print("O valor é %2f" %y)
print(" ")
print("O valor é %4.3f" %x)
print("O valor é %7.2f" %x)
print("O valor é %10f" %x)
print(" ")
print("x vale %f e y vale %f" %(x,y))
print("O valor é %.2f %s" %(x, unid))
print("O" + " valor" + " é " + str(x))
```



f-String

- **f-String:** É a forma mais moderna e eficiente de exibir informações (a partir do Python 3.6).

```
[1] nome = "Jones"
    profissão = "Engenheiro"

    print(f"Meu nome é {nome} e eu sou {profissão}.")

    x = 2
    print(f"O quadrado de {x} é {x**2}")
```

```
Meu nome é Jones e eu sou Engenheiro.
O quadrado de 2 é 4
```



Entrada de dados

- **Entrada de dados:** são realizadas por meio do comando `input`. Obs.: a saída sempre é uma string.
- **Exemplo 1:** realizando uma soma?



```
x = input("Digite a variável 1: ")  
y = input("Digite a variável 2: ")  
  
print(x + y)
```



```
Digite a variável 1: 2  
Digite a variável 2: 3  
23
```

Cuidado!



Entrada de dados

- Exemplo 2: realizando uma soma de números inteiros:
- Exemplo 3: realizando uma soma de números reais:

```
[3] # Entrada de números inteiros
x = int(input("Digite a variável 1: "))
y = int(input("Digite a variável 2: "))

print(x + y)
```

```
Digite a variável 1: 2
Digite a variável 2: 3
5
```

```
# Entrada de números reais
x = float(input("Digite a variável 1: "))
y = float(input("Digite a variável 2: "))

print(x + y)
```

```
Digite a variável 1: 2.1
Digite a variável 2: 3.2
5.3000000000000001
```



Estrutura condicional

- **Estrutura condicional:** de acordo com o resultado de uma expressão lógica, pode se desviar o fluxo de um programa.

```
if expressão:
    #declarações. Obs.: Cuidado com a indentação!
elif expressão:
    #declarações
else:
    #declarações
```



Estrutura condicional

- Exemplo: Programa para cálculo das raízes de uma função polinomial do segundo grau.



```
print("Programa para cálculo das raízes de uma função polinomial do segundo grau")
a = int(input("Digite o valor de a: "))
b = int(input("Digite o valor de b: "))
c = int(input("Digite o valor de c: "))

delta = b**2 - 4*a*c

if delta > 0:
    x1 = (-b + delta**(1/2))/(2 * a)
    x2 = (-b - delta**(1/2))/(2 * a)
    print("Duas raízes reais distintas x1 = %.2f e x2 = %.2f" %(x1, x2))
elif delta == 0:
    x = (-b) / (2 * a)
    print("Duas raízes reais iguais x = %.2f" %x)
else:
    print("Não existem raízes reais.")
```

```
Programa para cálculo das raízes de uma função polinomial do segundo grau
Digite o valor de a: 1
Digite o valor de b: -1
Digite o valor de c: -1
Duas raízes reais distintas x1 = 1.62 e x2 = -1.62
```



Estrutura repetitiva: while

- **While:** enquanto a condição for verdadeira o trecho de programa dentro da estrutura do while será repetido.

```
while expressão:  
    #declarações
```

- Exemplo:



```
# While  
contagem = 0  
while (contagem < 9):  
    print('A contagem é:', contagem)  
    contagem = contagem + 1  
print("Fim")
```




Funções

- **Funções:** são um conjunto de ações que recebem um nome com objetivo de serem reaproveitadas ao longo do código.

```
# Definição da função
def Nome_funcao(parâmetros):
    C O M A N D O S
    return Resultado

# Chamada da função no programa principal
variavel = Nome_funcao(argumentos)
```



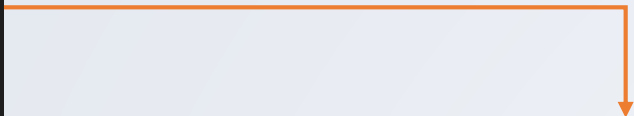
Funções

- Exemplo 1: funções sem retorno de valor.

```
[12] # Funções sem retorno de valor - código #1
      print("\nVocê fez um bom trabalho, Huguinho!")
      print("Muito obrigado por seus esforços neste projeto.")

      print("\nVocê fez um bom trabalho, Zézinho")
      print("Muito obrigado por seus esforços neste projeto.")

      print("\nVocê fez um bom trabalho, Luizinho!")
      print("Muito obrigado por seus esforços neste projeto.")
```



```
# Funções sem retorno de valor - código #2
def obrigado(nome):
    #Esta função agradece a pessoa recebida como parâmetro.
    print("\nVocê fez um bom trabalho,%s!" % nome)
    print("Muito obrigado por seus esforços neste projeto.")

obrigado('Huguinho')
obrigado('Zézinho')
obrigado('Luizinho')
```



Funções

- Exemplo 2: funções com retorno de valor.

```
# Funções com retorno de valor
def retornaExtenso(numero):
    #Recebe um valor numérico e retorna seu valor por extenso
    if numero == 0:
        return 'zero'
    elif numero == 1:
        return 'um'
    elif numero == 2:
        return 'dois'
    elif numero == 3:
        return 'três'
    else:
        return "Desculpe me, eu não conheço este número."

#Programa principal
op = 'S'
while op == 'S':
    N = int(input("Digite um número: "))
    resposta = retornaExtenso(N)
    print(resposta)
    op = input("Deseja inserir outro número (S/N): ").upper()

↳ Digite um número: 0
zero
Deseja inserir outro número (S/N): s
Digite um número: 4
Desculpe me, eu não conheço este número.
Deseja inserir outro número (S/N): n
```



Listas

- **Listas:** são variáveis que permitem armazenar uma sequência mutável de valores, mesmo que sejam de tipos distintos.



```
# Lista - exemplo
L = [7, "texto", True, 3.1416, ['outra lista', 18, 25]]

print(L[0])
print(L[2])
print(L[4])
print(L[-1])
print(L[-2])
print(L[-3])
print(L[4][1])
print(L[4][2])
print(L[0:2])
print(L[1:])
print(L[0:4:2])
print(L[::-1])
```

I



Listas – Funções

- **Funções:** algumas funções úteis já previamente criadas no Python:

- `len(lista)`: retorna o número de itens.
- `sum(lista)`: retorna a soma dos itens, se a lista possuir conteúdo numérico.
- `min(lista)`: retorna o item que precede os demais da lista.
- `max(lista)`: retorna o item que sucede os demais da lista.

□ `max(lista)`: retorna o item que sucede os demais da lista.



Listas – Métodos

- **Métodos:** são funções associadas a objetos, abaixo os métodos mais utilizados com listas:

- `lista.append(valor)`: adiciona o “valor” ao final da lista.
- `lista.insert(i, valor)`: adiciona o “valor” na posição “i” da lista.
- `lista.pop()`: remove o último elemento da lista.
- `Lista.pop(i)`: remove o elemento “i” da lista.
- `Lista.sort(reverse = True)`: ordena a lista na ordem crescente, caso seja adicionado o parâmetro opcional “reverse = True”, ordena na ordem decrescente.

o parâmetro opcional “reverse = True”, ordena na ordem decrescente.



Listas

- Exemplo:

```
# Exemplos de funções e métodos em listas
Lista = []
N = int(input("Quantos elementos deseja digitar: "))
i = 0
while i < N:
    x = float(input("Digite o elemento %i: " % (i + 1)))
    Lista.append(x)
    i = i + 1

media = sum(Lista) / len(Lista)
Lista.sort()
if len(Lista)%2 == 0:
    mediana = (Lista[len(Lista)//2] + Lista[len(Lista)//2 - 1])/2
else:
    mediana = Lista[len(Lista)//2]
print("A média é %.2f e a mediana é %.2f." % (media, mediana))

Lista.clear()
```

```
Quantos elementos deseja digitar: 3
Digite o elemento 1: 1
Digite o elemento 2: 2
Digite o elemento 3: 3
A média é 2.00 e a mediana é 2.00.
```

```
[78] # Cuidados ao copiar uma lista
lista_1 = [" banana", " maçã", "laranja"]
lista_2 = lista_1
lista_2.pop()

print(lista_1)
print(lista_2)

# Isso sim é uma cópia
lista_1 = [" banana", " maçã", "laranja"]
lista_2 = lista_1.copy() # list(lista_1) ou lista_1[:]
lista_2.pop()

print(lista_1)
print(lista_2)
```

Cuidado!

```
[' banana', ' maçã']
[' banana', ' maçã']
[' banana', ' maçã', 'laranja']
[' banana', ' maçã']
```



Estrutura For

- **Estrutura For:** assim como o `while`, o `for` também é uma estrutura repetitiva, porém diferente de outras linguagens, o `for` do Python é iterativo.

```
for variável_de_iteração in sequência:  
    #declarações
```




Estrutura For

- Exemplos:

```
# Exemplo 1
for i in range(10):
    print('i =', i)

for letra in 'Python':
    print('Letra atual:', letra)

frutas = ['banana', 'maçã', 'manga']
for fruta in frutas:
    print('Fruta atual:', fruta)
```

```
i = 0
i = 1
i = 2
i = 3
i = 4
i = 5
i = 6
i = 7
i = 8
i = 9
Letra atual: P
Letra atual: y
Letra atual: t
Letra atual: h
Letra atual: o
Letra atual: n
Fruta atual: banana
Fruta atual: maçã
Fruta atual: manga
```

```
#Exemplo 2
x = [1, 2, 3, 4, 5]
x_2 = [i*i for i in x]
x_3 = [i*i for i in x if i%2 == 0]
x_4 = [i*i if i%2 == 0 else i for i in x]

print(x)
print(x_2)
print(x_3)
print(x_4)
```

```
[1, 2, 3, 4, 5]
[1, 4, 9, 16, 25]
[4, 16]
[1, 4, 3, 16, 5]
```



Dicionários

- Dicionários: variável que permite armazenar uma estrutura de dados composta por conjuntos de chaves e valores.

```
#Exemplo
D = { 10 : True,
      "Um texto" : [42, 1.6],
      4.5 : 20,
      False : { 9 : 4, True : "algo" }
}

print(D[10])
print(D["Um texto"])
print(D[False])
print(D[False][9])
print(D[False][True])
```

```
True
[42, 1.6]
{9: 4, True: 'algo'}
4
algo
```



Web Scrapping

- Web Scrapping: Utilizar web para coletar informações.

```
import webbrowser, sys, requests, bs4, json

CEP = input("Digite o CEP desejado: ")
res = requests.get('https://viacep.com.br/ws/' + CEP + '/json/')
soup = bs4.BeautifulSoup(res.text, 'html.parser')
JSON_Datalist = soup.get_text()
dados = json.loads(JSON_Datalist)

if 'erro' in dados:
    print("CEP inválido.")
else:
    print('CEP: %s' % dados['cep'])
    print('Endereço: %s' % dados['logradouro'])
    print('Complemento: %s' % dados['complemento'])
    print('Bairro: %s' % dados['bairro'])
    print('Cidade: %s' % dados['localidade'])
    print('Estado: %s' % dados['uf'])
```

Digite o CEP desejado: 09580-900
CEP: 09580-900
Endereço: Praça Mauá
Complemento: 01
Bairro: Mauá
Cidade: São Caetano do Sul
Estado: SP



Referências bibliográficas

MENEZES, N. N. C., Introdução à programação com Python, 3ª Edição. São Paulo: Editora Novatec, 2019.

Notas de aula: Prof. Anderson Harayashini Moreira, pós-graduação em CD e IA, IMT, 2022.