CIÊNCIA DE DADOS COM PYTHON

CAP.2 – FUNDAMENTOS DE PYTHON

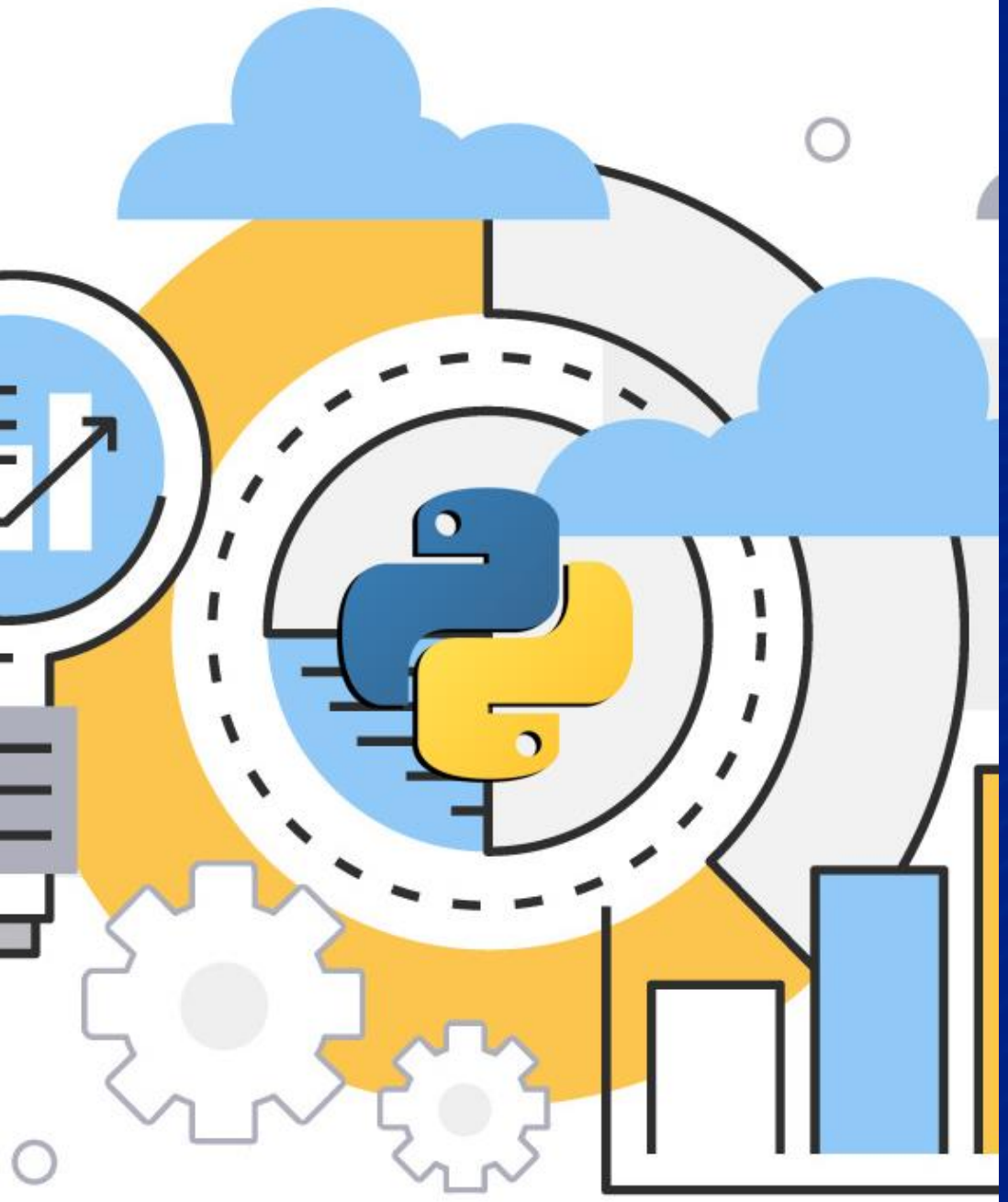
Prof. Renzo Paranaíba Mesquita

Inatel

OBJETIVOS

- Conhecer detalhes de funcionamento da linguagem de programação Python;
- Compreender a sintaxe e recursos primordiais dessa linguagem de programação;
- Se familiarizar com a linguagem por meio da prática de exercícios fundamentais.

Inatel



TÓPICOS

1. Por que usar Python?
2. Instalando o Python;
3. Começando com Python;
4. Tipos de Dados;
5. Operadores Aritméticos;
6. Módulos e Pacotes em Python;
7. Manipulando cadeiras de caracteres;
8. Estruturas de Decisão;
9. Laços de Repetição.

Inatel

2.1. POR QUE USAR PYTHON?

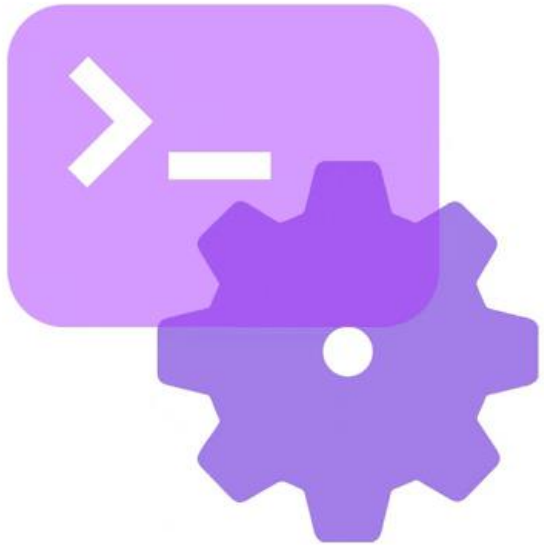
- Linguagem de Propósito Geral
- Consideradas umas das linguagens mais fáceis e intuitivas dos últimos anos
- Linguagem multiplataforma
- Conjunto de Bibliotecas muito completa
- É software livre
- Forla o programador a ser organizado para o código executar
- Orientada a Objetos

2.2. INSTALANDO O PYTHON

- O Python possui duas versões: Python 2 e Python 3
  - O Python 2 foi descontinuado
  - Um código feito em Python 2 não executará no Python 3
  - Daqui pra frente, seguiremos apenas com Python 3.X

- Para instalá-lo localmente, é necessário:
  - Inicialmente instalar uma versão do Python 3 em sua máquina
  - Ela pode ser baixada em https://www.python.org/downloads/
  - Em seguida, instalar uma IDE ou Editor capaz de usar desta instalação para executar seu código (Ex: PyCharm IDE)
  - Para uso de dependências (bibliotecas externas), será necessário o download quando necessário (consulte como fazer isso na sua IDE ou editor predileto)

- Para uso de ambientes Python *online* como o Google Colab, geralmente não é necessário qualquer instalação de ambiente ou dependências, pois já possui tudo que é necessário (pelo menos para uso de bibliotecas populares)



2.3. COMEÇANDO COM PYTHON

- Todo comando em Python é uma Função;
  - Função nada mais é que um comando que é escrito dentro de um bloco que começa com (e termina com)
- Para escrevermos algo na tela, usamos a função **print()**
  - Para mensagens textuais, o conteúdo deve ser cincundado por ‘’
- **EXEMPLOS:**

  print(‘Olá Python’) #somente texto

  print(7+7) #operações matemáticas

  print(‘O resultado de 7 + 7 é’, 7+7) #texto + operações
- **OBSERVAÇÕES:**
  - O Python não usa ; no final das linhas (é uma linguagem interpretada e trabalha com identação);
  - Veja que para concatenar coisas no print usamos a , (vírgula)
  - Observe o uso de # para comentário de uma única linha

2.3. COMEÇANDO COM PYTHON

- Variáveis permitem armazenar valores
  - São espaços de memória capazes de armazenar dados de diferentes tipos;
- Para se criar uma variável em Python, não é necessário colocar o tipo dela;
- Basta escrever o nome dela, seguido do operador = (recebe) e do valor que deseja armazenar;
- **EXEMPLOS:**

```
nome = 'Vincent' #str
idade = 30 #int
peso = 83.5 #float
print(nome, idade, peso)
```

- **OBSERVAÇÕES:**
  - O Python tem tipagem dinâmica, ou seja, o próprio interpretador baseado no perfil do dado armazenado consegue inferir seu tipo.

2.3. COMEÇANDO COM PYTHON

- Para solicitarmos alguma informação ao usuário, podemos fazer que a variável receba o resultado de uma função input

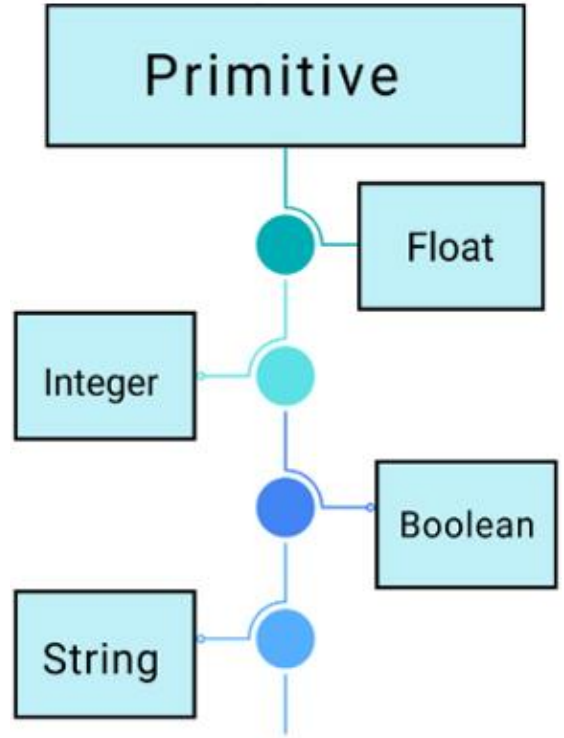
- **EXEMPLOS:**

```
nome = input('Qual o seu nome?')
idade = int(input('Qual sua idade?'))
peso = float(input('Qual o seu peso?'))
print('Seu nome é {} sua idade {} e seu peso {}'.format(nome,idade,peso))
```

- **OBSERVAÇÕES:**
  - Vale ressaltar que o conteúdo inserido pelo usuário será guardado naturalmente no formato str
  - Observe a necessidade de funções específicas aninhadas para se realizar os devidos *parsing* de dados
  - Observe o uso da função format() juntamente à função print() para mostrar os resultados de forma customizada

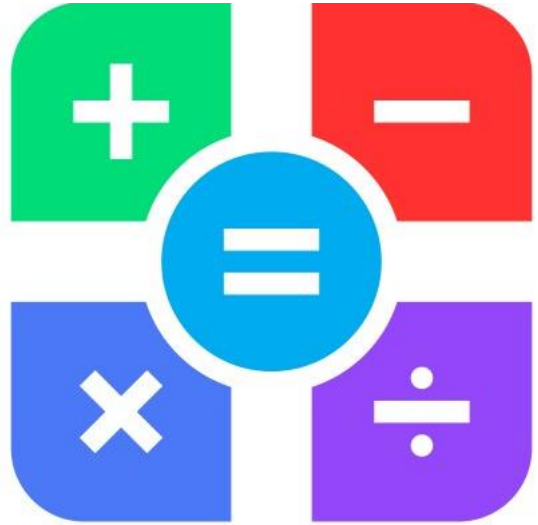
2.4. TIPOS DE DADOS

- Em todas as linguagens de programação existem os chamados tipos básicos de dados (também chamados de tipos primitivos)
- Os tipos primitivos do Python são:
  - int: usado para representar números inteiros positivos e negativos
  - float: usado para representar números reais positivos e negativos
  - bool: usado para aceitar valores True e False;
  - str: usado para armazenar textos dentro de ‘’
- **OBSERVAÇÕES:**
  - Para saber qual é o tipo de uma determinada variável em Python, use da função **type(variável)** e imprima o resultado



2.5. OPERADORES ARITMÉTICOS

- No Python, os operadores aritméticos mais populares são:
  - + (soma)
  - - (subtração)
  - * (multiplicação)
  - / (divisão)
  - == (igualdade)
  - ** (potenciação)
  - // (divisão inteira)
  - % (resto da divisão)
- **OBSERVAÇÕES:**
  - Da mesma forma que acontece na matemática, devemos nos atentar às precedências dos operadores
  - O == é diferente do =. O == significa “igual a” e o = significa “recebe”



2.6. MÓDULOS E PACOTES EM PYTHON

- Um dos destaques da linguagem Python é o seu conjunto incrível de bibliotecas disponíveis;
- Existem duas formas de importar bibliotecas externos no Python, são elas:

  import novaBiblioteca #importa uma biblioteca inteira com seus recursos;

  from novaBiblioteca import parteDaBiblioteca #importa apenas um recurso específico da biblioteca
- **EXEMPLOS:**

```
import math
num = 2.5
print(math.trunc(num))
```

- **OBSERVAÇÕES:**
  - Dentre as bibliotecas mais conhecidas, podemos citar a **math**, replete de funções matemáticas úteis prontas para serem utilizadas como ceil(), floor(), trunc(), sqrt(), factorial(), etc.
  - Para mais detalhes, consulte: https://docs.python.org/3/library/math.html
  - O repositório oficial de bibliotecas do Python é o https://pypi.org onde poderá encontrar e ler mais sobre bibliotecas úteis para diferentes fins.

2.7. MANIPULANDO CADEIAS DE CARACTERES

- Qualquer conteúdo que seja colocado dentro de ‘’ ou “” é chamado de string (str), que nada mais é que uma cadeia de caracteres
- Ao ser criada uma String, o Python a organiza da seguinte forma dentro da memória, dando um índice para cada um de seus caracteres:

H	e	l	l	o		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10

- **EXEMPLOS:**

```
var[6] #captura a letra W da String
var[:5] #captura a palavra Hello
var[6:11] #captura a palavra World (6 inclusive e 11 exclusive)
var[6:] #também captura a palavra World
var[0:10:2] #mostra HloWrd (ou seja, pula de 2 em 2)
```

- **OBSERVAÇÕES:**
  - Observe que ao inserir índices como argumentos no Python, o primeiro índice é **inclusive** e o segundo **exclusive** (como no exemplo 3)

2.7. MANIPULANDO CADEIAS DE CARACTERES

- Outras formas legais para manipulação de Strings:

H	e	l	l	o		W	o	r	l	d
0	1	2	3	4	5	6	7	8	9	10

- **EXEMPLOS:**

```
len(var) #retorna o número de caracteres de uma String
var.count('o') #conta o número de o's da palavra
var.count('l',0,5) #conta quantos l's tem dentro de Hello
var.find('lo') #indica a posição de onde começa 'lo'
'World' in var #retorna True se a palavra se encontrar dentro do texto
var.replace('World', 'Python') #troca uma palavra por outra no texto
var.upper() #todas as letras do texto ficam MAIÚSCULAS
var.lower() #todas as letras do texto ficam MINÚSCULAS
var.split() #quebra o texto em partes
```

- Para mais detalhes, consulte: https://docs.python.org/pt-br/3.13/library/string.html

2.8. ESTRUTURAS DE DECISÃO

- No Python, a indentação (organização correta do código) é essencial para que seus comandos funcionem corretamente
- No bloco if-else isso não seria diferente:
- **EXEMPLOS:**

```
idade = int(input('Entre com sua idade:'))
if idade < 18:
        print('Você é menor de idade')
else:
        print('Você é maior de idade')
```

- **OBSERVAÇÕES:**
  - Observe a indentação e sintaxe do if-else
  - Não é necessário colocar um condicional dentro do ()
  - Espaçamentos corretos podem ser definidos apertando a tecla **tab** a partir do canto esquerdo da tela
  - Condicionais internos podem ser criados adicionando-se o **elif**

2.9. LAÇOS DE REPETIÇÃO

- Em Python, podemos fazer laços de repetição com **for** ou **while**
- **EXEMPLOS:**

```
for c in range(0,10): #mostrando um conteúdo x vezes
        print('Python é legal')

for c in range(0,10): #a variável c assume um novo valor a cada iteração
        print(c)

for c in range (10,0,-1) #realizando uma contagem regressiva
        print(c)
```

- **OBSERVAÇÕES:**
  - Observe mais uma vez que o segundo argumento é exclusive
  - No lugar de valores fixos, variáveis podem ser utilizadas

2.9. LAÇOS DE REPETIÇÃO

- Em Python, podemos fazer laços de repetição com **for** ou **while**
- **EXEMPLOS:**

```
var = 1
while var < 5:
        print(var)
        var +=1

senha = ‘’
while senha != ‘python123’:
        senha = input(‘Entre com a senha correta:’)
print(‘Bem-vindo ao sistema! ☺’)
```

- **OBSERVAÇÕES:**
  - No primeiro exemplo observe o laço sendo executado até uma condição ser atingida
  - No segundo exemplo observe o laço executando infinitamente até a senha correta ser inserida
  - Quando uma condição for atendida dentro do laço e você desejar pará-lo, basta usar a palavra-chave **break**

EXERCÍCIOS (PARTE 1)

1. Crie um programa que leia seu nome completo e mostre:
   - Seu nome com todas as letras maiúsculas
   - Seu nome com todas as letras minúsculas
   - Quantas letras ao todo tem seu nome
   - E como seria se trocássemos seu último nome para “do Inatel”

2. Mostre a tabuada de um número que o usuário escolher dentro de um intervalo específico também escolhido por ele

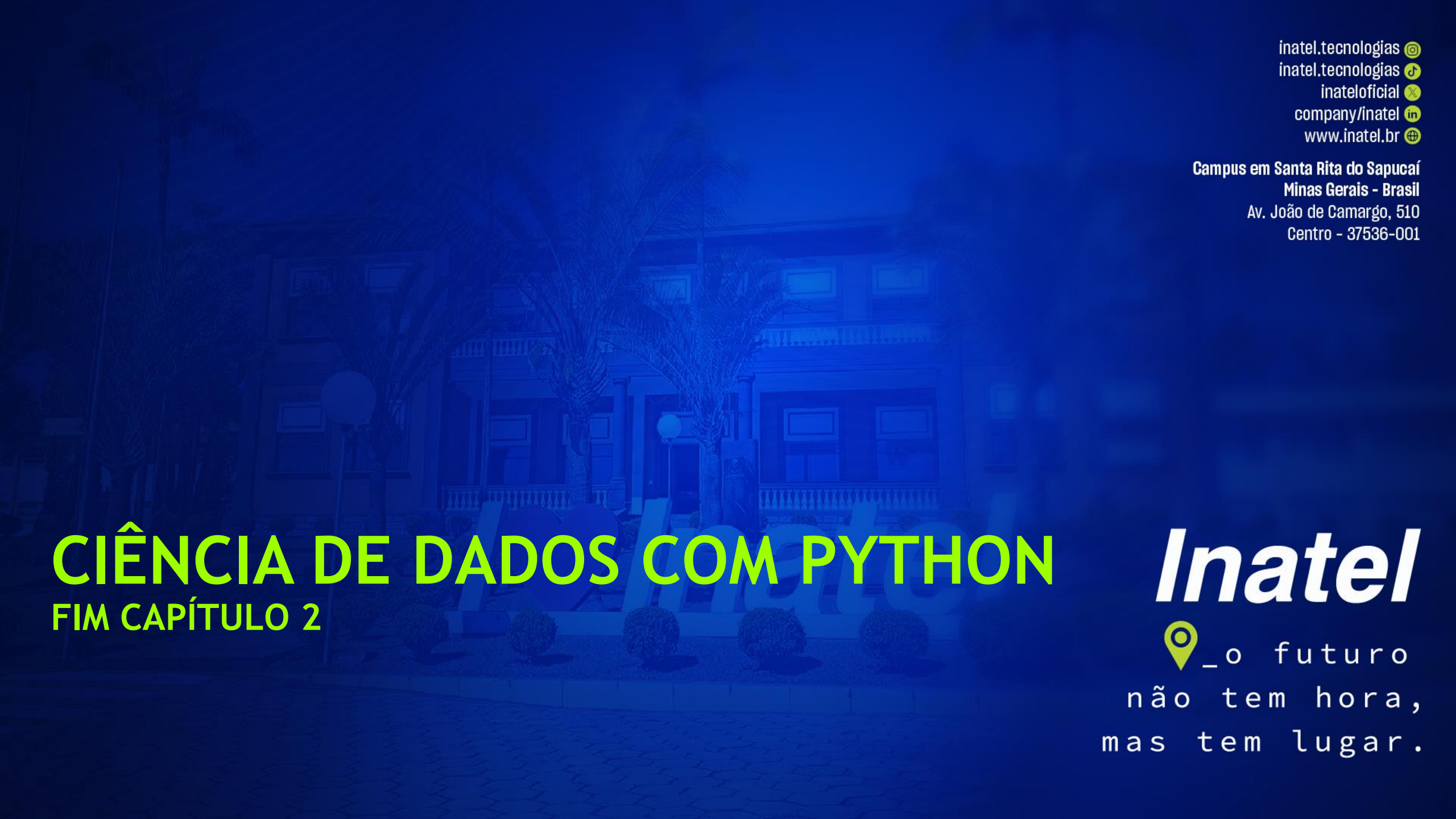
3. Faça um programa que leia o sexo de uma pessoa e diga se ela é homem (caso seja digitado M) ou mulher (caso seja digitado F). Caso seja digitado algo inválido, continue perguntando até que o usuário entre com um sexo válido

EXERCÍCIOS (PARTE 2)

4. Desenvolva um script que pergunte a distância de uma viagem em Km. Calcule o preço da passagem, cobrando R$0.50 por Km para viagens até 200Km e R$0.45 para viagens mais longas

5. Faça um programa que leia um número entre 1000 e 9999 e mostre na tela
- qual o número da unidade
- número da dezena
- número da centena
- E número do milhar

6. Peça ao usuário para entrar com um número decimal. Em seguida, aplique e mostre o resultado:
- da raiz quadrada deste número
- função teto
- função chão
- sua parte inteira



inatel.tecnologias
inatel.tecnologias
inateloficial
company/inatel
www.inatel.br

Campus em Santa Rita do Sapucaí
Minas Gerais - Brasil
Av. João de Camargo, 510
Centro - 37536-001

CIÊNCIA DE DADOS COM PYTHON

FIM CAPÍTULO 2

Inatel

_o futuro
não tem hora,
mas tem lugar.