print("Introdução ao Python")


```python
def encrypt(msg):
    msgseparada = split(msg.lower())
    lid = []
    cr = []
    for n in msgseparada:
        b = ABC.find(n)
        lid.append(b)
    for n in lid:
        b = NUMQWERT[n]
        cr.append(b)

    crypt = ''.join(cr)
    return crypt

def decrypt(msge):
    msgeseparada = split(msge)
    lid = []
    dcr = []
    for n in msgeseparada:
        b = NUMQWERT.find(n)
        lid.append(b)
    for n in lid:
        b = ABC[n]
        dcr.append(b)
```

Autores: Daniel José

PET: http://www.peteletrica.uff.br/

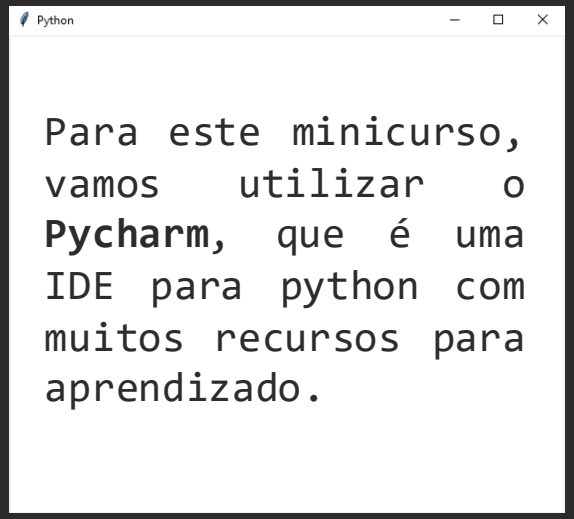
PET ELÉTRICA

Universidade Federal Fluminense

print("Introdução")

O **Python** é uma linguagem de programação **interpretada.** Além de ser de alto nível, ótima para iniciantes, também possui muitas bibliotecas excelentes voltadas para diversas áreas como **Data Science** e **Machine Learning**.

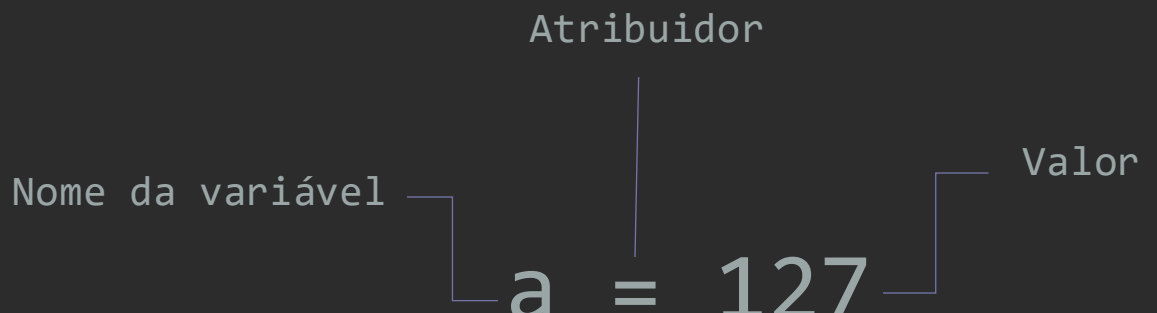
Para instalar o Python no seu sistema operacional, você precisa baixar o instalador no site oficial, isso fará o download do **Python 3** para sitemas de **32 bits**, para o instalador de **64 bits**, vá em downloads e escolha seu sistema operacional na aba aberta.



PET ELÉTRICA

print("Variáveis")

Uma **variável** é um espaço na memória do computador destinado a um dado que é alterado durante a execução do algoritmo. Elas possuem um nome e um valor e o sinal de "=" é o que atribui um valor a uma variável.



Exemplos:

```
nome = "Pet elétrica"

pi = 3.14159

type(a) #tipo de variável de a
type(nome)
type(pi)
```

print("Tipos de dados")

Têm-se três tipos “básicos” de dados em Python:
1.Tipos Numéricos (int, float, complex)
2.Tipos Textuais (str)
3.Tipos Lógicos (bool)

1. Tipos Numéricos:
int(x) – **inteiros**
float(x) – **decimais**
complex(*re*,*im*) - **complexos**

2. strings(str) são objetos **indexáveis** delimitados por aspas ou aspas duplas.

x="

0	1	2	3	4	5	6	7
O		P	y	t	h	o	n

"

3. boolean(bool) representa dois valores **True** e **False**.

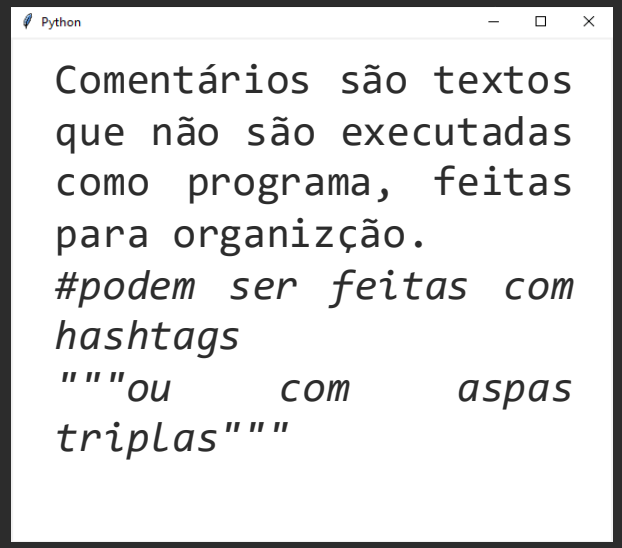
print("Funções básicas de entrada e saída")

print() é uma função para imprimir uma mensagem específica na tela (seu conteúdo vai sempre ser convertido a uma string para ser exibido).

```
print("Hello World!")
print("Hello {}".format("Pet Elétrica"))
```

input() é uma função que permite entrada de um usuário.

```
a = input("Digite seu nome: ")
```



Exemplo:

```
print("Digite seu nome: ")
x = input()
print("Hello, " + x)
#concatenando strings com um operador
```

print("Operadores")

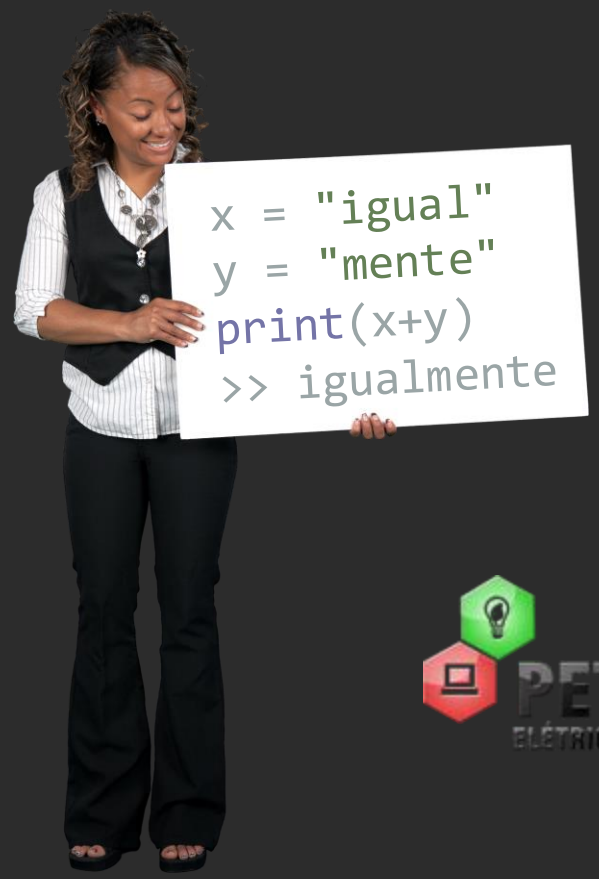
1. Operadores lógicos são bem usados em **condicionais** para verificar se as condições são validas, retornam True ou False.
Exemplos:
or, **and** e **not**.

2. Operadores relacionais são usados para **comparar** dois valores.
Exemplos: >, < , <=, >=, ==, !=.

3. Operadores aritiméticos são usados para **operações matemáticas** comuns com valores numericos.
Exemplos:
+, -, **, *, /, //, %.

Python

O operador " + " quando utilizado pode ser usado para "somar" strings, com o nome de **concatenação.**



PET ELÉTRICA

```
print("exercícios")
```

1. Faça um Programa que peça dois números e imprima a soma.

2. Faça um Programa que peça as 4 notas bimestrais e mostre a média.

3. Faça um Programa que pergunte quanto você ganha por hora e o número de horas trabalhadas no mês. Calcule e mostre o total do seu salário no referido mês.

4. Faça um Programa que peça a temperatura em graus Farenheit, transforme e mostre a temperatura em graus Celsius.
C = (5 * (F-32) / 9).

5. Faça um programa para uma loja de tintas. O programa deverá pedir o tamanho em metros quadrados da área a ser pintada. Considere que a cobertura da tinta é de 1 litro para cada 3 metros quadrados e que a tinta é vendida em latas de 18 litros, que custam R$ 80,00. Informe ao usuário a quantidades de latas de tinta a serem compradas e o preço total.

print("Estruturas de condição")

Estruturas de condição como **"if"** e **"else"** são utilizadas para verificão se uma expessão é verdadeira e executar um bloco de código a partir disso, o **"elif"** permite checar várias condições de uma vez.

Sintaxe:

```
if expressão1:
    bloco
elif expressão2:
    bloco
elif expressão3:
    bloco
else:
    bloco
```

Python

Para um bloco de código fazer parte de uma condicional, em python, ele tem que ser indentado para representar uma "hierarquia" na execução.

```
print("exercícios")
```

6.Faça um Programa que leia três números e mostre o maior e o menor deles.

7.Faça um Programa que pergunte em que turno você estuda. Peça para digitar M-matutino ou V-Vespertino ou N-Noturno. Imprima a mensagem "Bom Dia!", "Boa Tarde!" ou "Boa Noite!" ou "Valor Inválido!", conforme o caso.

8. As Organizações Tupiniquim resolveram dar um aumento de salário aos seus colaboradores e lhe contraram para desenvolver o programa que calculará os reajustes.
Faça um programa que recebe o salário de um colaborador e o reajuste segundo o seguinte critério, baseado no salário atual:

- salários até R$ 1000,00 (incluindo) : aumento de 20%
- salários entre R$ 1000,00 e R$ 1700,00 : aumento de 15%
- salários entre R$ 1700,00 e R$ 2500,00 : aumento de 10%
- salários de R$ 2500,00 em diante : aumento de 5% Após o aumento ser realizado, informe na tela:
- o salário antes do reajuste;
- o percentual de aumento aplicado;
- o valor do aumento;
- o novo salário, após o aumento.

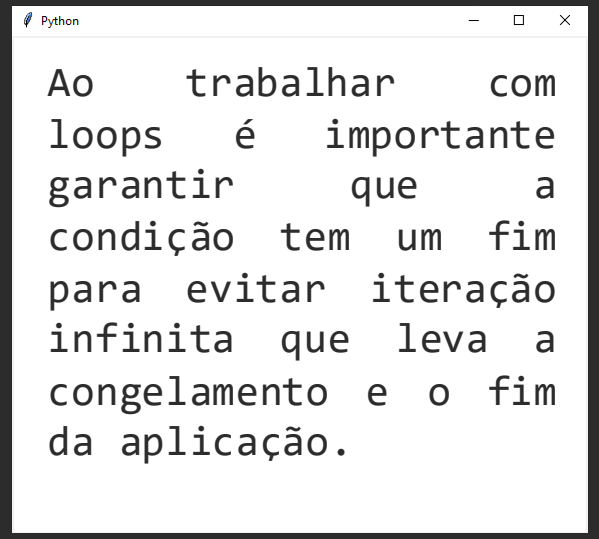
print("Estruturas de repetição")

Estruturas de repetição (loops) são utilizadas para executa um conjunto de instruções várias vezes seguidas. Os **loops** são condificados com o **"for"** e o **"while"**, o primeiro nos permite percorrer os itens de uma coleção e, para cada um deles, executar um bloco de código. Já o while, executa um conjunto de instruções várias vezes enquanto uma condição é atendida.

Sintaxe:

```
i = 1
while i < 6:
    print(i)
    i += 1
```

```
lugares =
["casa", "parque", "bairro"]
for x in lugares:
    print(x)
```



PET ELÉTRICA

```
print("while")
```

Declarações importantes:

break: interrompe o programa mesmo se a condição for verdadeira

continue: interrompe a iteração atual e segue para a próxima

else: executa outro bloco de código quando a condição não é mais verdadeira

```
i = 1
while i < 6:
    print(i)
    if i == 3:
        break
    i += 1
```

```
i = 1
while i < 6:
    print(i)
    i += 1
else:
    print("i não é menor que 6")
```

```
i = 0
while i < 6:
    i += 1
    if i == 3:
        continue
    print(i)
```

print("for")

Declarações importantes:
1.break: interrompe o programa mesmo se a condição for verdadeira
2.continue: interrompe a iteração atual e segue para a próxima
3.else: executa outro bloco de código quando a condição não é mais verdadeira

Função range(), essa função retorna uma sequência de números começando em 0 até o número do argumento (excluindo o argumento)

```
for x in range(6):
    print(x)
for x in range(2, 6):
    print(x)
```

Loop dentro do loop:

```
adj = ["vermelha", "grande", "saborosa"]
frutas = ["maça", "banana", "uva"]

for x in adj:
    for y in frutas:
        print(x, y)
```

1.

```
frutas = ["maça", "banana", "uva"]
for x in frutas:
    print(x)
    if x == "banana":
        break
```

Uma string também é um objeto iterável:

```
for x in "banana":
    print(x)
```

3.

```
for x in range(6):
    print(x)
  else:
    print("Acabou!")
```

2.

```
frutas = ["maça", "banana", "uva"]
for x in frutas:
    if x == "banana":
        continue
    print(x)
```

print("exercícios")

9. Faça um programa que peça uma nota, entre zero e dez. Mostre uma mensagem caso o valor seja inválido e continue pedindo até que o usuário informe um valor válido.

10. Faça um programa que em uma sequência de números de 0 a 50 se o número for divisível por 3 imprima Fizz, se for divisível por 5 imprima Buzz, se for divisível por 15 imprima FizzBuzz. Caso contrário a tudo isso, imprima o número

11. Faça um Programa que leia um vetor A com 10 números inteiros, calcule e mostre a soma dos quadrados dos elementos do vetor.

12. Supondo que a população de um país A seja da ordem de 80000 habitantes com uma taxa anual de crescimento de 3% e que a população de B seja 200000 habitantes com uma taxa de crescimento de 1.5%. Faça um programa que calcule e escreva o número de anos necessários para que a população do país A ultrapasse ou iguale a população do país B, mantidas as taxas de crescimento.

print("exercícios")

13. **Nome ao contrário em maiúsculas.** Faça um programa que permita ao usuário digitar o seu nome e em seguida mostre o nome do usuário de trás para frente utilizando somente letras maiúsculas. Dica: lembre-se que ao informar o nome, o usuário pode digitar letras maiúsculas ou minúsculas.

14. **Tamanho de strings.** Faça um programa que leia 2 strings e informe o conteúdo delas seguido do seu comprimento. Informe também se as duas strings possuem o mesmo comprimento e são iguais ou diferentes no conteúdo.

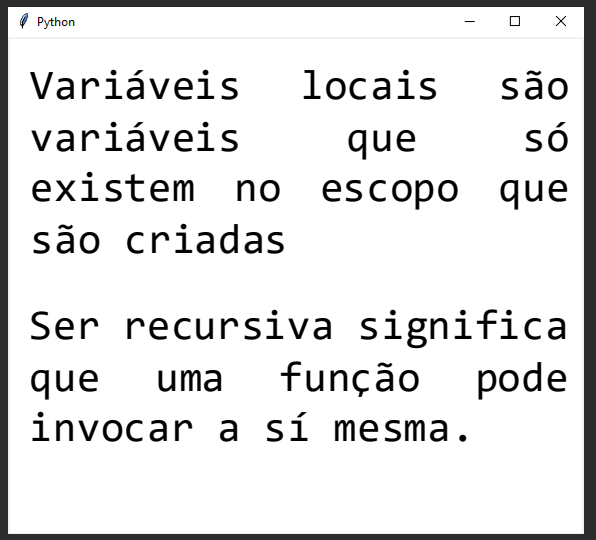
15. **Data por extenso.** Faça um programa que solicite a data de nascimento (dd/mm/aaaa) do usuário e imprima a data com o nome do mês por extenso.

```
print("Funções")
```

As funções em python são criadas com **"def"** para definir o bloco de ações. Elas podem ter variáveis locais e são recursivas.

Funções **lambda** são uma forma de deixar o programa mais simples e funcional.

```
x = lambda a : a + 10
print(x(5))
```



Nome da função

Parâmetros necessários para execução da função

```
def soma_metade(x,y):
    x1=x/2
    y2=y/2
    return x1+y2
```

Bloco de ações

o resultado que a função retorna

Exemplo:

```
a=4
b=66
print(soma_metade(a,b))
```

```
print("exercícios")
```

16.Faça um programa, com uma função que necessite de três argumentos, e que forneça a soma desses três argumentos.

17.Faça uma função que peça uma nota, entre zero e dez. Mostre uma mensagem caso o valor seja inválido e continue pedindo até que o usuário informe um valor válido.

18. Faça uma função que retorne o reverso de um número inteiro informado. Por exemplo: 127 -> 721.

19.Faça uma função para a leitura de duas notas parciais de um aluno.
O programa deve calcular a média alcançada por aluno e apresentar:
A mensagem "Aprovado", se a média alcançada for maior ou igual a sete;
A mensagem "Reprovado", se a média for menor do que sete;
A mensagem "Aprovado com Distinção", se a média for igual a dez.

print("Estruturas de dados")

Estrutura de dados é qualquer meio utilizado para **armazenar e recuperar informações**.
Todas as estruturas de dados têm como objetivo, armazenar um conjunto de informação **geralmente** do mesmo tipo.

Lista é uma coleção de valores modificáveis, pode-se usada para representar vetores e matrizes.

```
Lista1 = ["Jorge", "Maria", "João", 45456]
```

Set é uma coleção de itens desordenados e desindexados abertos com chaves.

```
Set1 = {72, "Salgado", "P"}
```

Dicionários são coleções de elementos desordenados em que cada elemento tem uma chave única atrelada, primeiro a chave e depois o elemento.

```
dicionário = {1 : 'Fabio', 2 : 'Maria', 3 : 'João', 4 : 'José'}
```

Tupla é basicamento uma lista com dados fixos, diferenciada pelo uso dos parênteses ao invés de colchetes.

```
Tupla1=(1706,"Jorge", "u")
```

PET ELÉTRICA

```
print("exercícios")
```

20. Faça um Programa que leia um vetor de 10 números reais e mostre-os na ordem inversa.

22. Faça um Programa que leia um vetor A com 10 números inteiros, calcule e mostre a soma dos quadrados dos elementos do vetor.

21. Faça um programa que tenha uma função notas() que pode receber várias notas de alunos e vai retornar um dicionário com as seguintes informações: - Quantidade de notas - A maior nota - A menor nota - A média das notas - A situação (opcional).

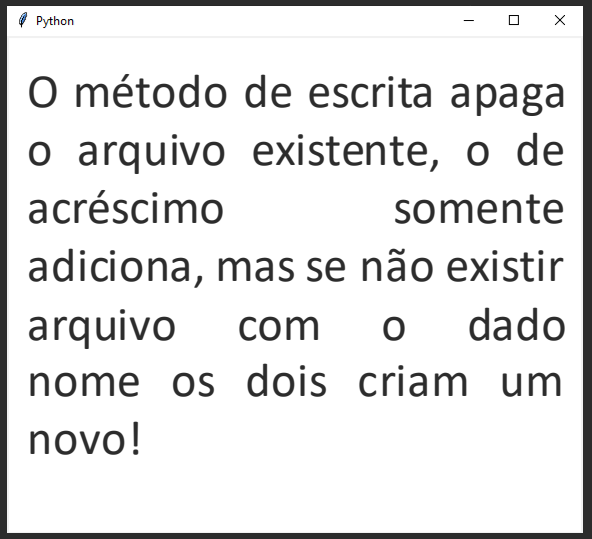
23. Utilizando listas faça um programa que faça 5 perguntas para uma pessoa sobre um crime. As perguntas são:

- "Telefonou para a vítima?"
- "Esteve no local do crime?"
- "Mora perto da vítima?"
- "Devia para a vítima?"
- "Já trabalhou com a vítima?" O programa deve no final emitir uma classificação sobre a participação da pessoa no crime. Se a pessoa responder positivamente a 2 questões ela deve ser classificada como "Suspeita", entre 3 e 4 como "Cúmplice" e 5 como "Assassino". Caso contrário, ele será classificado como "Inocente".**

print("manipulação básica de arquivos")

Para início de qualquer manipulação de arquivos em python é necessário preparar o arquivo declarando um **modo de operação** e abrir com o comando open(). Leitura **"r"**, escrita **"w"** e acréscimo **"a"**.

```
arquivo = open("exemplo.txt", "r")
texto = arquivo.read()
arquivo.close()
print(texto)
```



- read: lê todo o arquivo e retorna uma string com seu conteúdo texto.
- readline: lê apenas a primeira linha e a retorna em forma de string.
- readlines: lê todo o arquivo e retorna uma lista de linhas (string).
- write: escreve uma string.
- writelines: escreve as strings de uma lista de strings.

Try / Except é uma forma de manter o programa sendo executado mesmo quando executado.

PET ELÉTRICA

```
print("Orientação a objetos")
```

Orientação a objetos é um paradgma de programação feito para tratar sistemas mais complexos ao longo do tempo. Basicamente, você pode criar uma classe de objetos e métodos específicos para eles.

Construtor

Método

```
class pessoa:
    def __init__(self,nome,idade):
        self.nome = nome
        self.idade = idade

    def idade10anos(self):
        return int(self.idade) + 10

a = pessoa("Daniel","21")
print(a.idade10anos())
```

```
print("exercícios")
```

24. Classe Bola: Crie uma classe que modele uma bola:

- Atributos: Cor, área, volume
- Métodos: trocaCor e mostraCor

25. Classe Pessoa: Crie uma classe que modele uma pessoa:

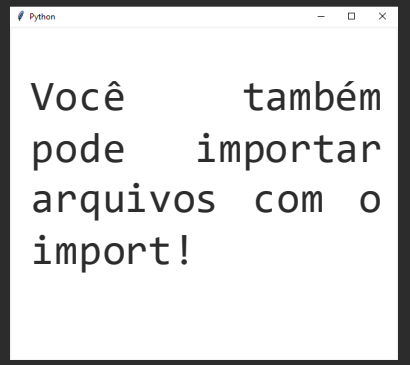
- Atributos: nome, idade, peso e altura
- Métodos: Envelhercer, engordar, emagrecer, crescer. Obs: Por padrão, a cada ano que nossa pessoa envelhece, sendo a idade dela menor que 21 anos, ela deve crescer 0,5 cm.

print("Bibliotecas")

Bibliotecas são modulos de comandos pronto pra você adicionar ao python. Você pode utilizar esses módulos no seu código com o comando import.

Você também pode importar somente uma parte especfica do módulo utilizando o from x import y.

tkinter é para desenvolvimento de interfaces gráficas.



random é um módulo gerador de números pseudo-aleatorios.

math é um bom módulo para tratar fatoriais e logaritmos.

Pillow(PIL) já serve para tratar imagens dentro do código.