



# PYTHON DO ZERO AO PRO



**mentorama.**  
@prof.felipeassuncao



# INTRODUÇÃO AO PYTHON



**mentorama.**  
@prof.felipeassuncao

# Neste módulo

Aula 1 - Introdução ao Python

Aula 2 - Instalação

Aula 3 - Variáveis, tipos de dados e operadores

Aula 4 - Condições

Aula 5 - Repetições

Aula 6 - Exercícios

# Recursos e ferramentas

- Python
- Jupyter Notebook

# 1. INTRODUÇÃO AO PYTHON

mentorama.



# O que é o Python

- É uma linguagem de programação de altíssimo nível
- Linguagem interpretada, orientada a objetos, tipagem forte e dinâmica
- Criada por Guido Van Rossun em 1991
- Modelo de desenvolvimento aberto mantido pela PSF (Python Software Foundation)
- Ênfase na produtividade do programador e legibilidade do código

**mentorama.**



# De onde vem o nome?



**mentorama.**

# O Python é um boa escolha?

- A <https://pt.stackoverflow.com/> é a maior comunidade de desenvolvedores do mundo
- O que dizer da popularidade do Python nessa comunidade?



Fonte: <https://insights.stackoverflow.com/survey/2020#technology-programming-scripting-and-markup-languages>

**mentorama.**



# Vantagens do Python

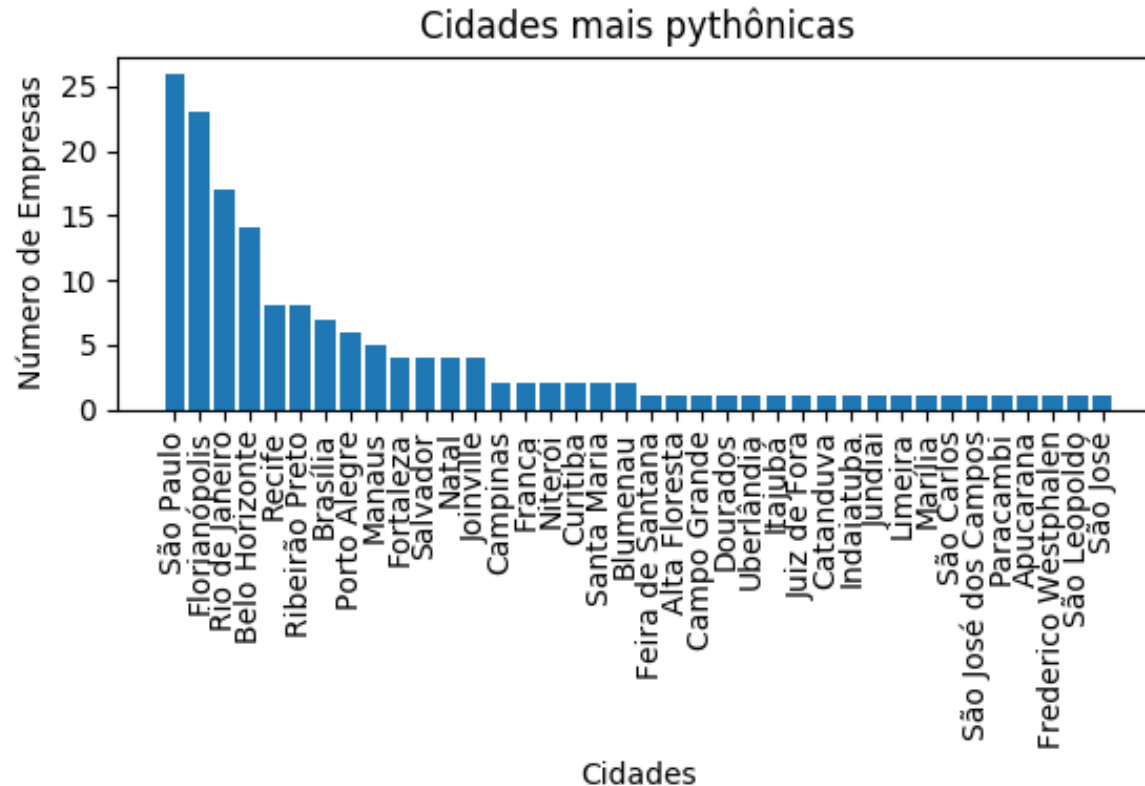
- Aprendizado mais rápido
- Em geral, envolve menos código
- Sintaxe fácil para ler e compreender
- Possui diversos pacotes e bibliotecas
- Utilizada pelas mais diversas companhias
- Ampla comunidade e gratuito

# Quem usa Python?



mentorama.

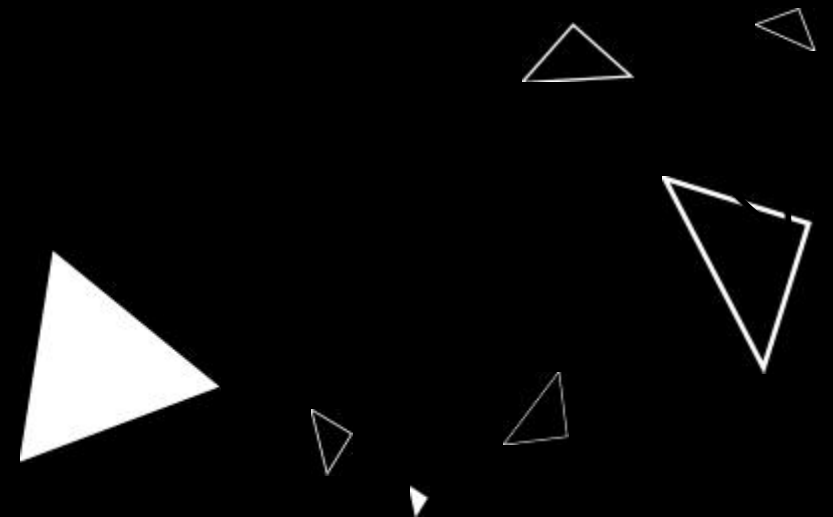
# Cidades mais Pythônicas



Fonte: <https://python.org.br/empresas/>

# PORQUE PROGRAMAR?

mentorama.



# Porque programar?

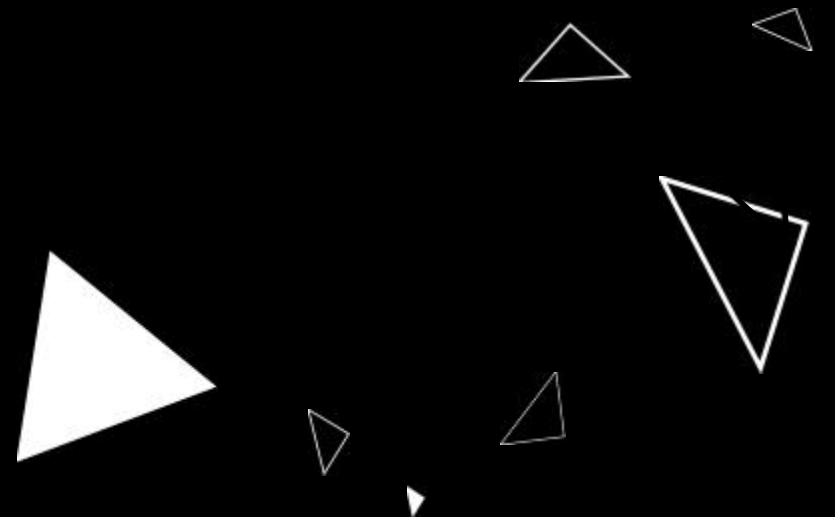


**mentorama.**

- Escrever páginas web
- Desenvolver aplicativos
- Salvar ou mundo, ou pelo menos ajudá-lo
- Testar suas habilidades
- Contribuir com softwares livres, código aberto e gratuito

# INTERPRETADOR VS. COMPILADOR

mentorama.



# Interpretador

- O processo de interpretação executa e traduz o código simultaneamente
- Não é gerado um arquivo final executável
- O arquivo executável já é o próprio código
- O interpretador traduz e executa cada uma das linhas

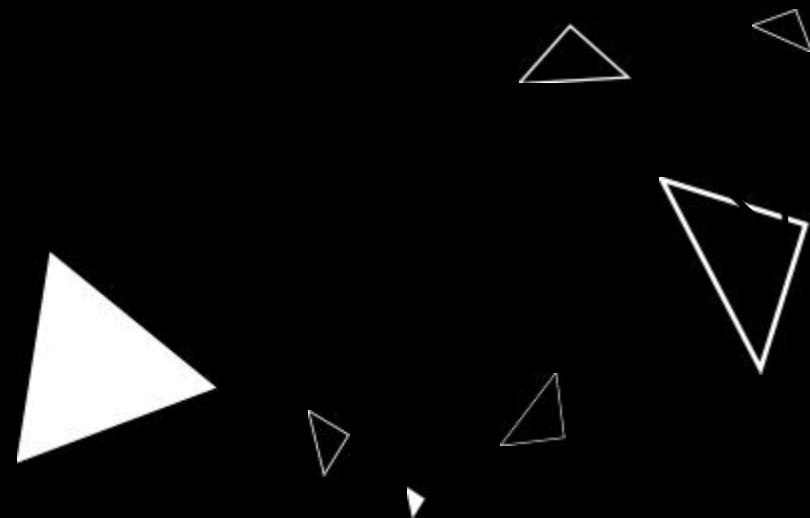
# Compilador

- Um compilador é um programa de sistema que traduz um programa descrito em uma linguagem de alto nível para um programa equivalente em código de máquina para um processador



# BOAS PRÁTICAS E DICAS

mentorama.



# Boas práticas de programação

- Letras maiúsculas vs. Minúsculas
- Uso das aspas
- Uso dos parênteses
- Uso dos espaços (incluindo indentação, alinhamento etc)
- PEP 8

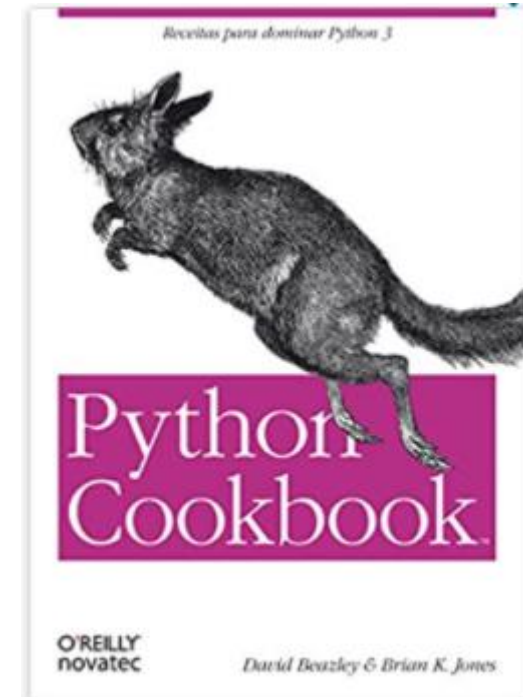
# Palavras reservadas

and	as	assert	break	class	continue
def	del	elif	else	except	exec
finally	for	from	global	if	import
in	is	lambda	nonlocal	not	or
pass	raise	return	try	while	with
yield	True	False	None		

# Progresso nos estudos

- Dedicção aproximada de duas horas diárias
- Escolha o melhor horário para você
- Seleção de materiais complementares (Livros, Vídeos)
- Resolução dos exercícios
- Participação nos fóruns (Mentorama e Python)
- Conhecer seu estilo de aprendizagem

# Bibliografia sugerida



**mentorama.**

# Muita calma nessa hora



**mentorama.**

- Não desanime, se no início, a sua programação com Python não parecer muito fluente
- Lembre-se de como é aprender uma língua nova
- Leia sempre, do início ao fim, de acordo com sua curiosidade
- Pare, relaxe e descanse quando necessário
- Retorne ao problema com novas perspectivas

# Resumo

- Introdução ao Python
- Motivação para programação
- Interpretador vs. compilador
- Boas práticas de programação
- Palavras reservadas
- Dicas de estudo
- Bibliografia sugerida



# 2. INSTALAÇÃO

mentorama.





# Instalação das ferramentas

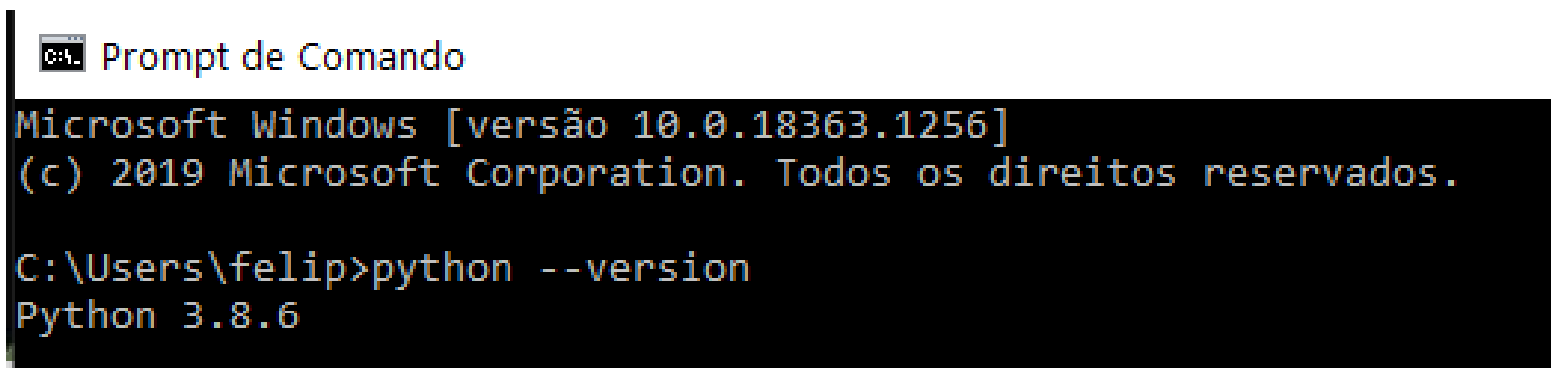
- Python
- Anaconda

# Instalação do Python

- Linux / UNIX: instalado por default
- Mac: instalado por default
- Windows: instalador disponível em <https://www.python.org/downloads/>

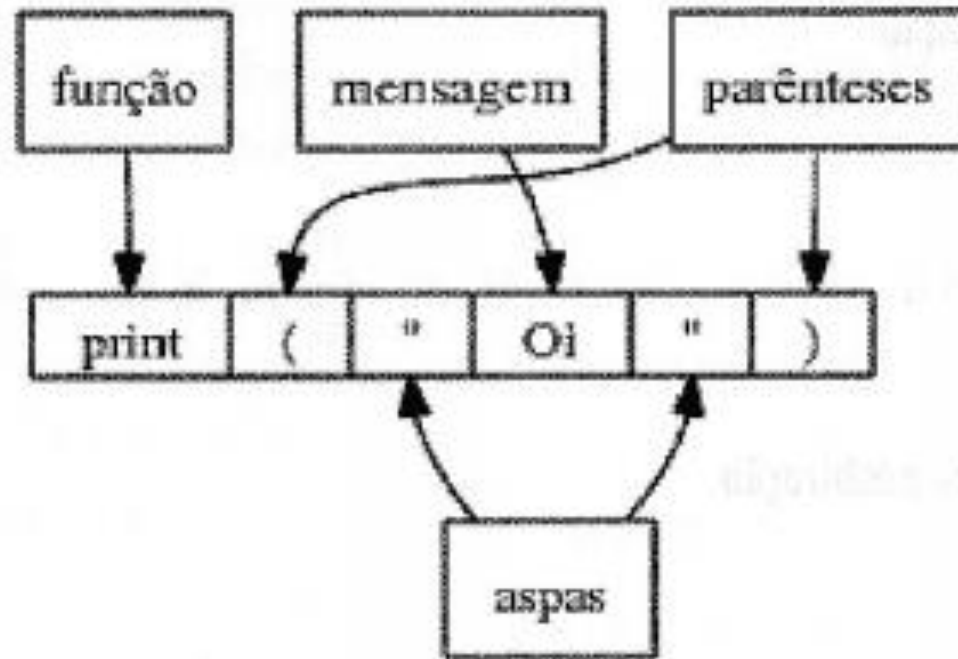
# Utilizando o interpretador

- Inicie o Prompt de Comando
- Digite o comando `python --version` na linha de comando
- Obs: Só vai funcionar se estiver instalado



```
C:\> Prompt de Comando  
Microsoft Windows [versão 10.0.18363.1256]  
(c) 2019 Microsoft Corporation. Todos os direitos reservados.  
  
C:\Users\felip>python --version  
Python 3.8.6
```

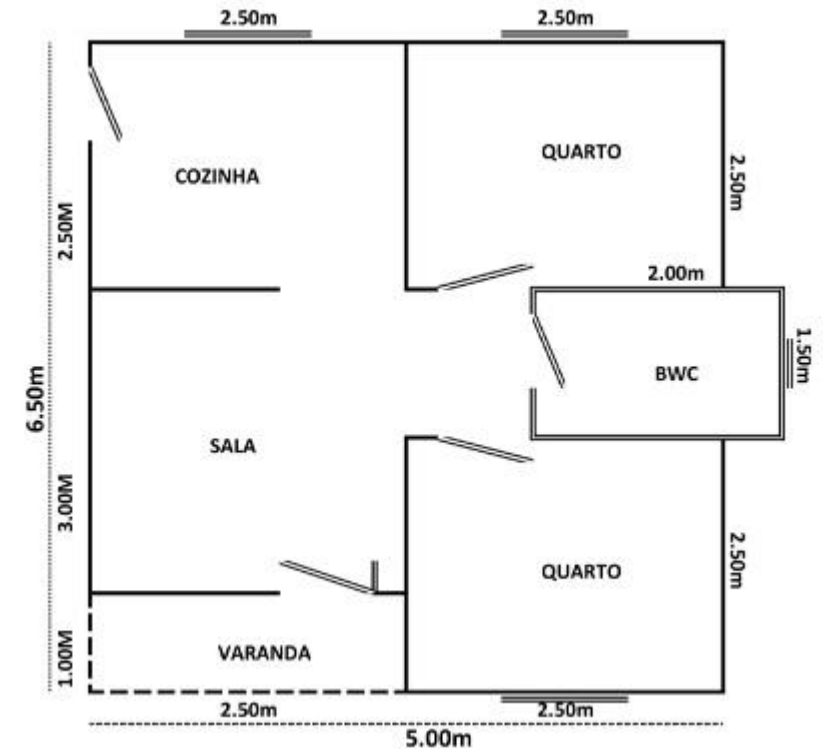
# Primeiro programa



mentorama.

# Ambiente virtual

*“Um ambiente virtual é um ambiente Python, de modo que o interpretador, as bibliotecas e os scripts Python instalados nele são isolados daqueles instalados em outros ambientes virtuais e (por padrão) quaisquer bibliotecas instaladas em um Python do “sistema”, ou seja, instalado como parte do seu sistema operacional.”*



# Ambiente virtual

- Criando um ambiente virtual:

```
python -m venv mentorama
```

- Ativando e desativando o ambiente virtual no Windows

```
mentorama\Scripts\activate.bat
```

```
mentorama\Scripts\deactivate.bat
```

- Ativando e desativando o ambiente virtual no Unix ou MacOS:

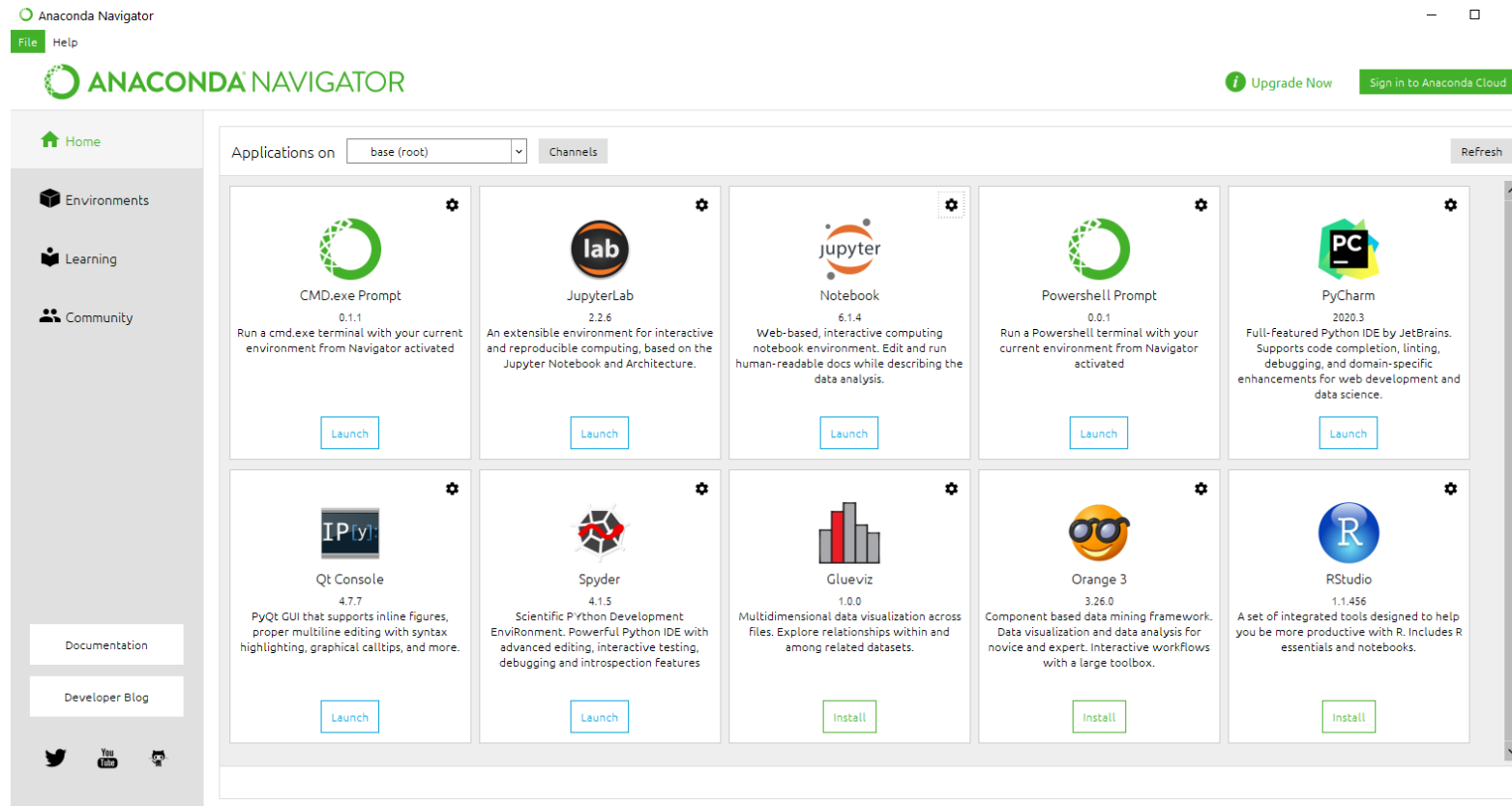
```
mentorama/bin/activate
```

```
mentorama/bin/deactivate
```

# Anaconda e Jupyter

- **Anaconda** é uma plataforma de ciência de dados para Python que possibilita a instalação de diferentes versões da linguagem com a criação de ambientes de desenvolvimento específicos.
- Jupyter é uma ferramenta que permite unir texto e código de maneira bastante eficiente.
- Disponível em: <https://www.anaconda.com/>

# Anaconda e Jupyter





# Instalação do Anaconda

- Disponível em: <https://www.anaconda.com/>
- Vamos instalar?

# Python Zen



**mentorama.**

- Vamos conhecer um pouco mais sobre a filosofia Python?
- Digite na linha de comando:  
`import this`

# Resumo

- Instalação do Python
- Utilizando o interpretador
- Primeiro programa
- Ambiente Virtual
- Filosofia Python (Python Zen)



# 3. VARIÁVEIS, TIPOS DE DADOS E OPERADORES

mentorama.



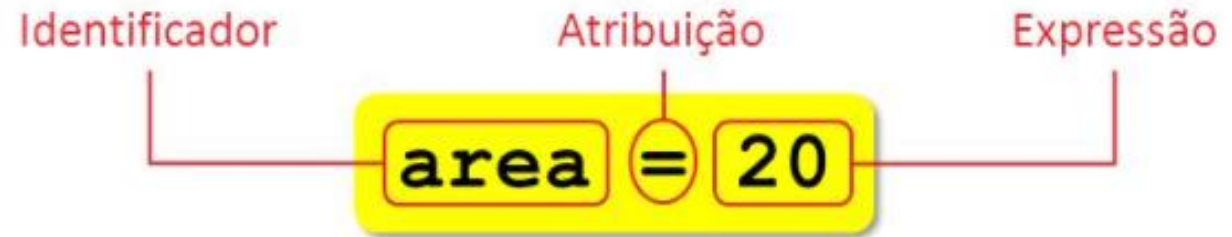
# Variáveis

- Posição na memória RAM que pode armazenar um valor (informação)
- São criadas no momento da execução
- Atribuímos um nome para as variáveis
- Possuem um “tipo” associado que determina a categoria de dados
- Permitem gravar e ler dados com facilidade

12
Maria
3.14
...
True

# Como definir variáveis?

- `n = 10`
- `n = "Olá mundo!"`
- `n = True`
- `n = 3.14`
- `n = (latitude, longitude)`
- `n = ['pera', 'maça', 'banana', 'melão', 'manga', 'uva']`



# Como escolher bons nomes?

NOME	VÁLIDO	COMENTÁRIOS
m1	Sim	Embora contenha um número, o nome m1 inicia com letra
mentorama	Sim	Nome formado por letras
mentorama20	Sim	Nome formado por letras e números, mas iniciado por letras
idade_média	Sim	O símbolo sublinha (__) é permitido e facilita a leitura de nomes
idade média	Não	Nomes de variáveis não podem conter espaços em branco
_m	Sim	O sublinha (__) é aceito em nomes de variáveis, mesmo no início
1m	Não	Nomes de variáveis não podem começar com números

**mentorama.**

# Tipos de variáveis

TIPOS DE VARIÁVEIS	DESCRIÇÃO
int	Para números inteiros
str	Para conjunto de caracteres
bool	Armazena True ou False
list	Para agrupar um conjunto de elementos
tupla	Semelhante ao tipo <b>list</b> , porém, imutável
dic	Para agrupar elementos que serão recuperados por uma <b>chave</b>



# Operadores relacionais

OPERADOR	DESCRIÇÃO
==	IGUAL A
>	MAIOR QUE
>=	MAIOR IGUAL QUE
<	MENOR QUE
<=	MENOR OU IGUAL QUE
!=	DIFERENTE DE

# Operadores lógicos

OPERADOR	DESCRIÇÃO	EXEMPLO
and	Operador “E”	<code>x == 10 and x &lt; y</code>
or	Operador “OU”	<code>x != 10 or x &lt; 0</code>
not	Operador de negação	<code>not(x &gt; y)</code>

# Operadores aritméticos

OPERADOR	DESCRIÇÃO	EXEMPLO
+	Soma	$1+1 = 2$
-	Subtração	$1-1 = 0$
*	Multiplicação	$2*2 = 4$
/	Divisão	$2/2 = 1$
**	Exponenciação	$2**3 = 8$
%	Módulo (resto da divisão)	$10\%5 = 0$
//	Extração da parte inteira da divisão	$10//9 = 1$

# Operações com strings

OPERADOR	DESCRIÇÃO	EXEMPLO
+	concatena (soma) duas strings	'abra' + 'cadabra' retorna 'abracadabra'
*	repete (replica) uma string múltiplas vezes	'ha' * 3 retorna 'hahaha'
[i]	retorna o i-ésimo caractere da string	'Brasil'[0] retorna 'B'
[i:j]	retorna a substring que vai dos índices i até j - 1	'Brasil'[0:3] retorna 'Bra'

# Composição de strings

MARCADOR	TIPO
%d	Números inteiros
%s	Strings
%f	Números decimais

“João tem %d anos” % X

- % indica a composição da string anterior com o conteúdo da variável X
- O %d dentro da primeira string chamamos de marcador de posição

# Entrada de dados

```
x = input("digite um número: ")  
print(x)
```

- A função input é utilizada para solicitar dados do usuário
- A função recebe um parâmetro que é a mensagem a ser exibida e retorna o valor digitado pelo usuário



# Vamos praticar?

- Nesta prática iremos explorar a utilização das variáveis, tipos de variáveis e algumas operações com números e strings
- Vamos testar alguns comandos?



# Resumo

- Variáveis
- Tipos de variáveis
- Operadores lógicos
- Operadores relacionais
- Operadores aritméticos
- Operações com strings
- Composições com strings
- Entrada e saída de dados

**mentorama.**





# 4. CONDIÇÕES

mentorama.



# IF

```
if <expr>:  
    <statement>  
    <statement>  
    ...  
    <statement>  
<following_statement>
```

# ELSE

```
if <expr>:  
    <statement>  
    ...  
else <expr>:  
    <statement>  
    ...  
<following_statement>
```

# Estruturas aninhadas

- Estruturas condicionais aninhadas são várias condições em cascatas, ou seja, um “if” dentro de outro “if”

# ELIF

```
if <expr>:  
    <statement(s)>  
elif <expr>:  
    <statement(s)>  
elif <expr>:  
    <statement(s)>  
...  
else:  
    <statement(s)>
```

# Vamos praticar?

- Iremos testar alguns códigos que usam condições (if, else, elif) para resolução dos problemas



# Resumo

- If
- Else
- Estruturas aninhadas
- Elif



# 5. REPETIÇÕES

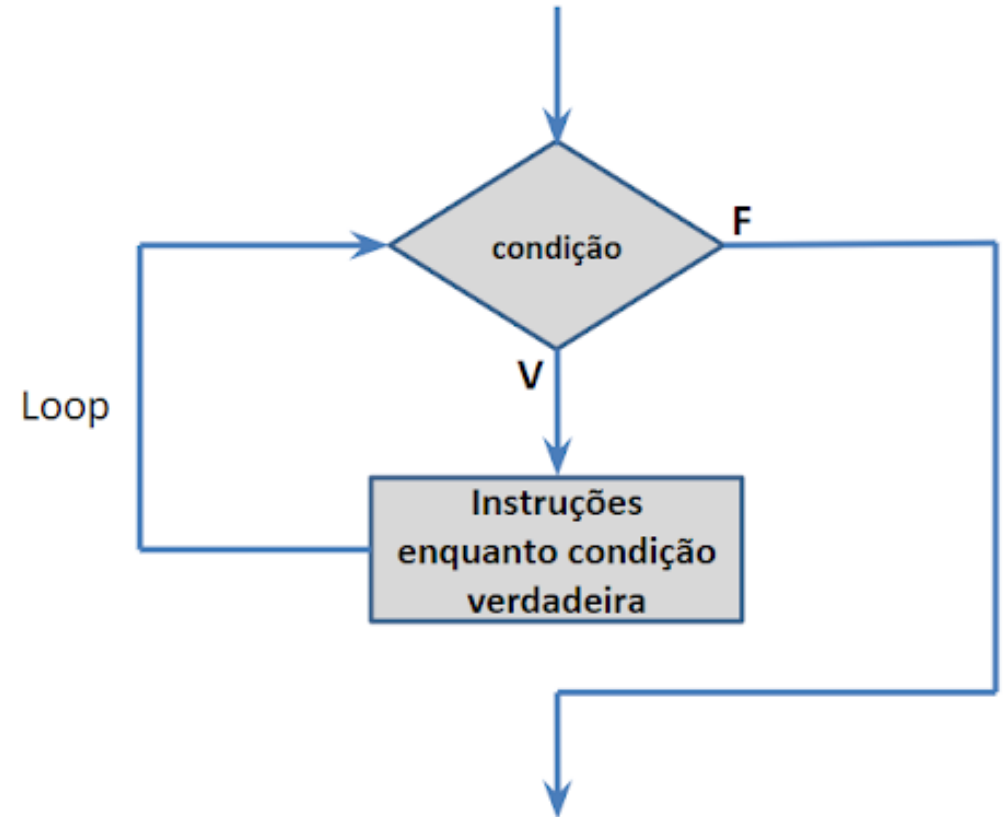
mentorama.





# O que são repetições

- Utilizadas para executar a mesma parte de um programa várias vezes, normalmente, dependente de uma condição
- Para executar a repetição, utilizamos os comandos while e/ou for



# WHILE

```
i = 0
```

```
while i < 5:
```

```
    print(i)
```

```
    i += 1
```



# FOR

```
for x in range (5):  
    print(x)
```



# Contadores

```
contador = 1
while contador <= 5:
    print("Olá, mundo")
    contador = contador + 1
```

```
contador = 0
for elemento in range(0,10):
    print(contador)
    contador = contador +1
```

**mentorama.**



# Acumuladores

- Podem ser utilizados em programas para calcular o total de uma soma

A diferença entre um contador e um acumulador é que nos contadores o valor adicionado é constante e, nos acumuladores, variável.

# Acumuladores

```
soma = 0
while True:
    valor = int(input("Digite um número para somar ou 0 para sair"))
    if valor == 0:
        break
    soma = soma + valor
print(soma)
```

**mentorama.**



# Interrompendo uma repetição

- Para interrompermos uma repetição utilizamos o comando `break`

```
soma = 0
```

```
while True:
```

```
    valor = int(input("Digite um número para somar ou 0 para sair"))
```

```
    if valor == 0:
```

```
        break
```

```
    soma = soma + valor
```

```
print(soma)
```

# Usando o continue

- O comando continue interrompe a execução do comando que vem logo abaixo do bloco, não o executando.

```
for val in "string":  
    if val == "i":  
        continue  
    print(val)  
print("The end")
```

**mentorama.**



# Vamos praticar?

- Nesta prática abordaremos os comandos de repetição, incluindo o while e o for
- Iremos verificar o funcionamento de um contador e um acumulador e o efeito do comando break



# Resumo

- For
- While
- Contadores
- Acumuladores
- Interrompendo a repetição



# EXERCICIOS

mentorama.

