

# PROGRAMAÇÃO 1

---

Aula 1 – Variáveis, atribuição, lógica booleana

Prof. Emanoel Barreiros



# Variáveis e tipos

- Variáveis armazenam valores
- Na realidade, variáveis em Python referenciam valores
- Tipos estão associados a valores, não variáveis
  - Tipagem dinâmica
  - Tipagem forte
- Demonstração
  - Abra o IDLE digite a = 10 e pressione ENTER
  - Digite type(a) e pressione ENTER
  - Digite a = 'abc' e pressione ENTER
  - Digite type(a) e pressione ENTER
  - Percebeu alguma diferença?

# Variáveis e tipos

- A tipagem é dinâmica pois variáveis podem apontar, em momentos diferentes para valores de tipos diferentes, o que dá a impressão que os tipos das variáveis muda
- A tipagem é forte, pois os tipos dos valores não mudam de formas inesperadas e as operações realizadas só acontecem quando os tipos envolvidos são compatíveis
- Variáveis são criadas no momento em que são usadas (atribuição) pela primeira vez
- Tentar usar o valor de uma variável antes de declará-la causará um erro

# Nomeando variáveis

- Nomes de variáveis...
  - ... só podem conter letras, números e caracteres *underscore*
  - ... podem iniciar com uma letra ou um *underscore*
  - Exemplo:
    - mensagem\_1 → ✓
    - 1\_mensagem → X
  - ... não podem conter nomes de variáveis
  - ... não podem ser palavras reservadas pela linguagem
  - Nomes de variáveis devem ser concisos e relevantes para o problema sendo resolvido
    - nome é melhor que n, endereço é melhor que e
  - Nomes em Python são *case sensitive*

# Aprendendo a interpretar erros

```
>>> mensagem = 'Olá mundo!'
>>> print(mensagem)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'mensagem' is not defined
>>> □
```

- O que aconteceu aqui?
- Observe o nome da variável e depois seu uso.
- Sempre leia as mensagens de erro e tente interpretá-las

# Exercício

- 1) Tente reproduzir este comportamento:

```
> ✓ python3 ex1.py  
Olá! Qual o seu nome?  
Emanoel ←  
Prazer em conhecê-lo, Emanoel!  
Quantos anos você tem?  
34 ←  
Legal, você terá 35 daqui a um ano...  
É, eu sou bom em matemática.
```

Entradas do usuário

Lembre-se disto aqui...

```
1 arquivoSenha = open('SecretPasswordFile.txt')  
2 senhaRecuperada = arquivoSenha.read()  
3 print('Digite sua senha:')  
4 senhaDigitada = input()  
5 if str(senhaDigitada) == senhaRecuperada:  
6     print('Acesso permitido.')  
7     if str(senhaDigitada) == '12345':  
8         print('Tente criar uma senha mais complexa. Por favor...')  
9 else:  
10    print('Acesso negado.')
```

# Convertendo valores

- As funções int(), str() e float() servem para converter valores entre tipos diferentes de tipo compatível
- `idade = int('34')`

# Comentários

- Código pode ficar grande e complexo rapidamente
- É uma boa prática comentar trechos relevantes do código para facilitar a leitura no futuro
- Em python, comentários iniciam com o caractere #
- Qualquer coisa após o # é ignorada pelo interpretador
- Usamos comentários para “desabilitar” linhas temporariamente

# Valores booleanos

- O tipo booleano é aquele cujos valores podem ser apenas verdadeiro e falso
- True ou False no caso do Python
- Fique atento à grafia, iniciam com letra maiúscula e não tem aspas

# Operadores de comparação

Operador	Significado
<code>==</code>	Igual a
<code>!=</code>	Diferente de
<code>&lt;</code>	Menor que
<code>&gt;</code>	Maior que
<code>&lt;=</code>	Menor ou igual que
<code>&gt;=</code>	Maior ou igual que

# Operadores booleanos

- São operadores que trabalham com valores booleanos
- Expressões que incluem esses operadores são avaliadas para valores booleanos
- Principais operadores
  - AND
  - OR
  - NOT

# Tabelas verdade

AND		Resultado
True	True	True
True	False	False
False	True	False
False	False	False

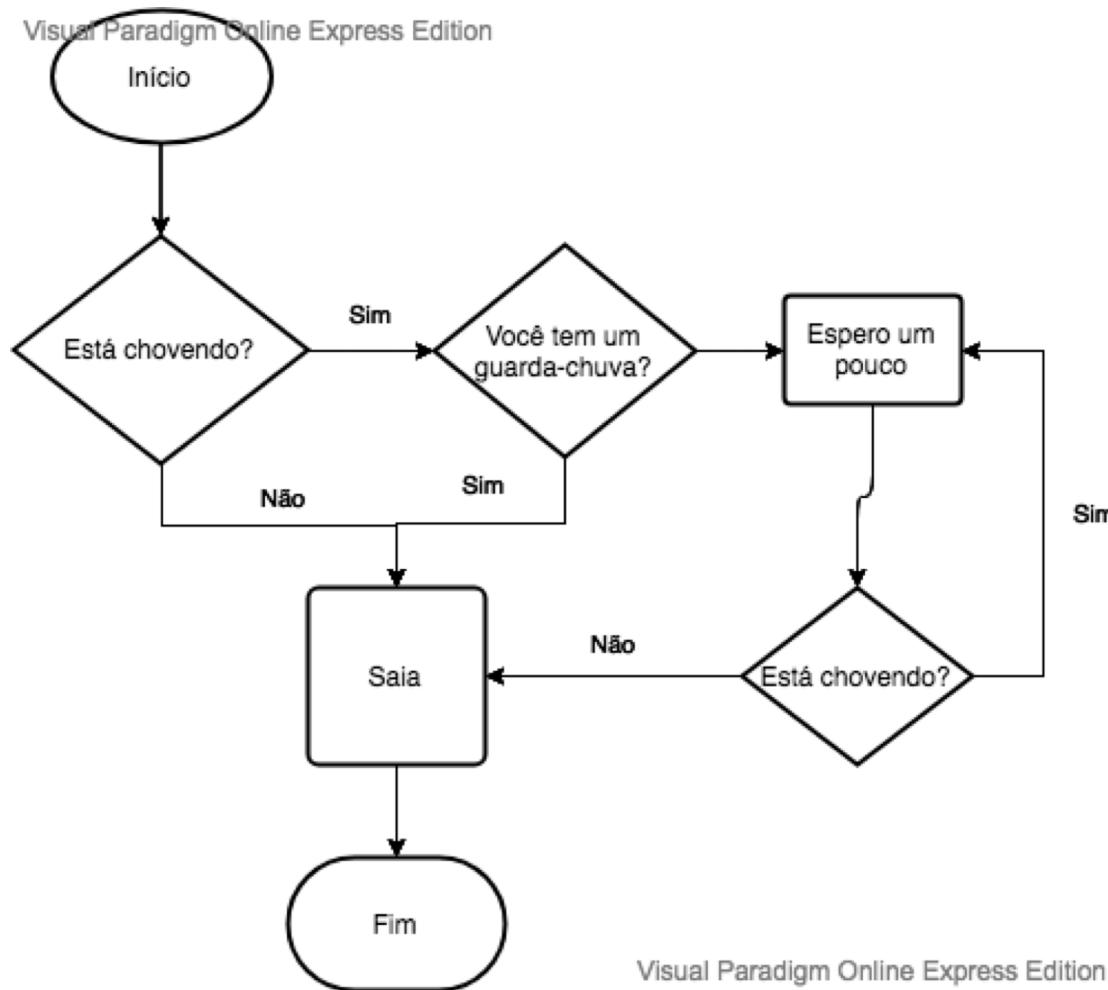
OR		Resultado
True	True	True
True	False	True
False	True	True
False	False	False

NOT	Resultado
True	False
False	True

# Misturando operadores booleanos e de comparação

- Operadores de comparação são avaliados como valores booleanos, logo podem ser usados em expressões com operadores booleanos:
  - $(4 < 5)$  and  $(5 < 6)$
  - $(4 < 5)$  or  $(5 > 6)$
- Precedência
  - not > and > or

# Controle de fluxo



# Elementos do controle de fluxo

- Podemos usar instruções de controle de fluxo para determinar quais trechos de código devem executar em determinadas situações
- Condições determinam se um trecho de código deve executar
- Condições nada mais são do que expressões booleanas
- Trechos de código podem ser agrupados em blocos controlados pelas estruturas de controle
- Observar a indentação do código para definir o bloco

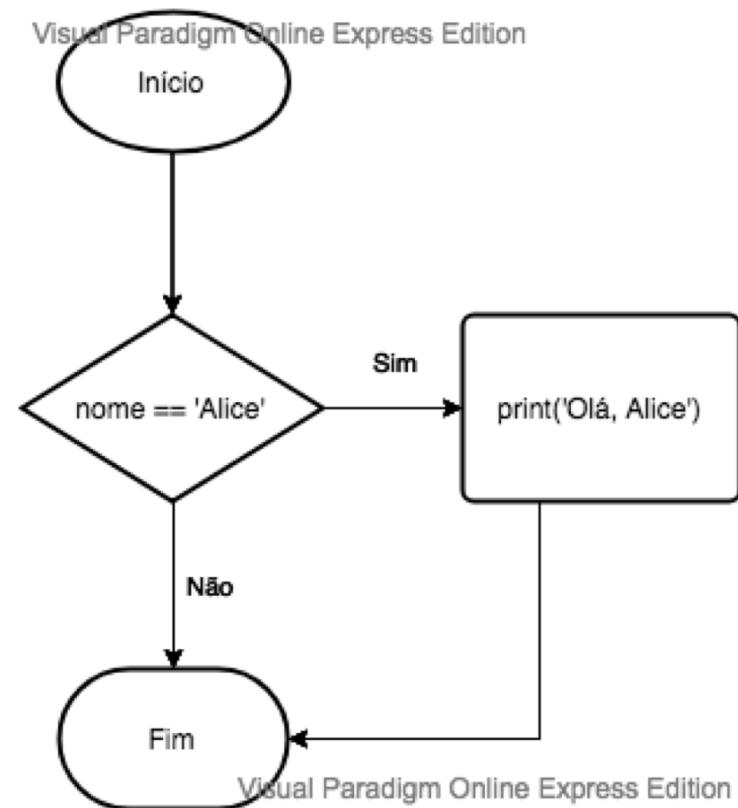
# Exemplo de um programa com fluxo de controle

```
1 arquivoSenha = open('SecretPasswordFile.txt')
2 senhaRecuperada = arquivoSenha.read()
3 print('Digite sua senha:')
4 senhaDigitada = input()
5 if str(senhaDigitada) == senhaRecuperada:
6     print('Acesso permitido.')
7     if str(senhaDigitada) == '12345':
8         print('Tente criar uma senha mais complexa. Por favor...')
9 else:
10    print('Acesso negado.'
```

# Instrução *if*

- Também chamado de comando de seleção
- Seu bloco de código será executado se sua condição for avaliada para True

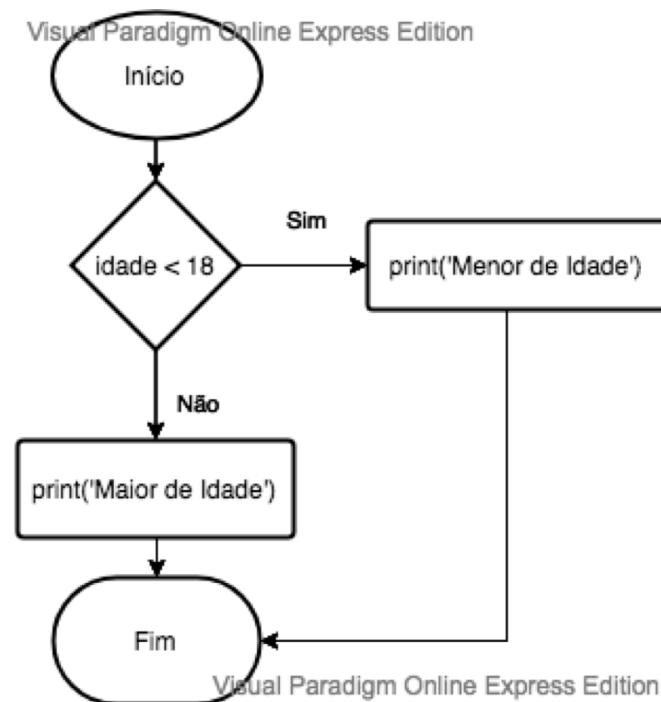
```
if nome == 'Alice':  
    print('Olá, Alice')  
  
if idade < 18:  
    print('Menor de idade.')
```



# Instrução else

- Instrução opcional após o bloco do comando if
- Define um bloco que será executado apenas quando a condição do seu bloco if for False

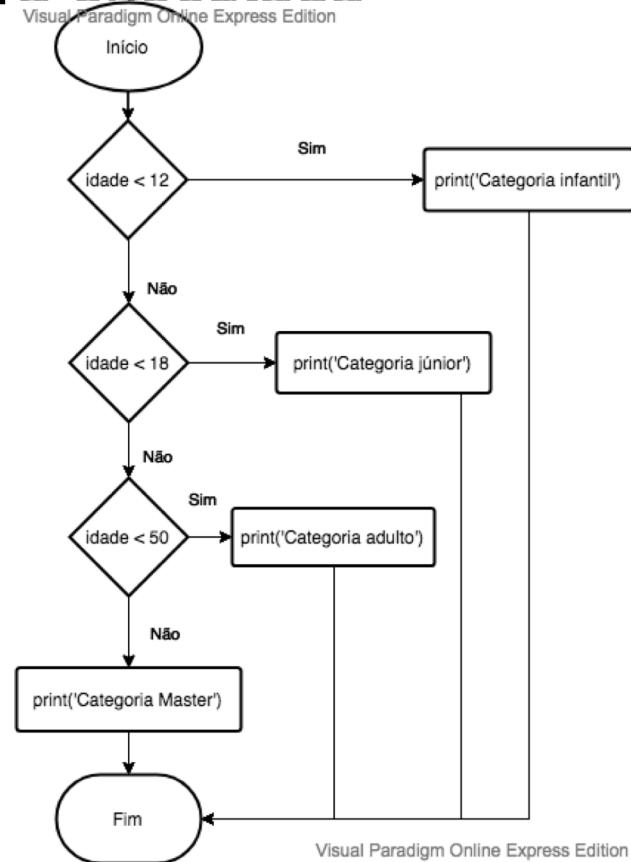
```
if idade < 18:  
    print('Menor de idade.')  
else:  
    print('Maior de idade.')
```



# Instrução *elif*

- Cria uma sequencia de condições que são checadas de cima para baixo até que uma seja verdadeira
- A primeira condição verdadeira será executada

```
if idade < 12:  
    print('Categoria infantil')  
elif idade < 18:  
    print('Categoria júnior')  
elif idade < 50:  
    print('Categoria adulto')  
else:  
    print('Categoria master')
```



# Exercícios

- 2) Solicite do usuário o ano em que ele nasceu e retorne a sua idade
- 3) Solicite a altura do usuário em centímetros e seu peso em kg. Calcule seu IMC e informe-o em que faixa ele está:
  - Menor que 18.5 , abaixo do peso
  - Entre 18.5 e 24.9, peso normal
  - Entre 25 e 29.9, sobrepeso
  - Maior que 30, está “fofinho”