

Introdução, Conceitos Básicos & Python



Professor Me. Luiz Carlos dos Santos Filho

Considerações Iniciais



Acostume-se com a nova forma de pensar:
ALGORITMOS

Não se irrite com a máquina

Só se aprende fazendo !!!

Atenção aos detalhes

Não esqueça das limitações do computador

A linguagem possui regras de escrita rigorosas

Cuidados na Digitação



Letras maiúsculas e minúsculas são diferentes

Aspas são importantes

Parênteses não são opcionais (Python)

Espaços são importantes

Comente seu programa em Python, # é o símbolo para comentários

Conceitos Básicos



Computadores podem executar **tarefas** simples ou complexas em alta velocidade e repetir operações um número indeterminado de vezes.

Tanto as tarefas complexas ou simples são realizadas por meio de **operações** simples.

Para fazê-lo, os computadores precisam ser programados, instruídos “passo a passo” com estas operações.

Conceitos Básicos



Exemplo: Média semestral = Nota da Prova P1 somado com Nota da Prova P2, resultado dividido por 2.

Naturalmente computadores não sabem **conceitos** como média e notas de prova.

Instruções para o computador:

- receba um número que represente a nota P1;
- receba um número que represente a nota P2;
- somar o 1º valor com o 2º valor e dividir o resultado por 2
- armazenar o resultado na memória;
- exibir o resultado

Conceitos Básicos



- receba um número que represente a nota P1;
- receba um número que represente a nota P2;
- somar o 1º valor com o 2º valor e dividir o resultado por 2;
- exibir o resultado

Tradução das instruções para a linguagem Python:

```
# Calculo da média entre duas notas
P1=float(input("Entre com a Nota P1: "))
P2=float(input("Entre com a Nota P2: "))
Media=((P1+P2)/2)
print("A média das notas é = ",Media)
```

Esta sequência de instruções pode ser chamada de algoritmo e pode ser traduzida numa linguagem que o computador compreenda.

Esta sequência de instruções é um programa (conjunto de comandos conhecidos = Instruction List)

Conceitos Básicos



Linguagem Materna
Humana

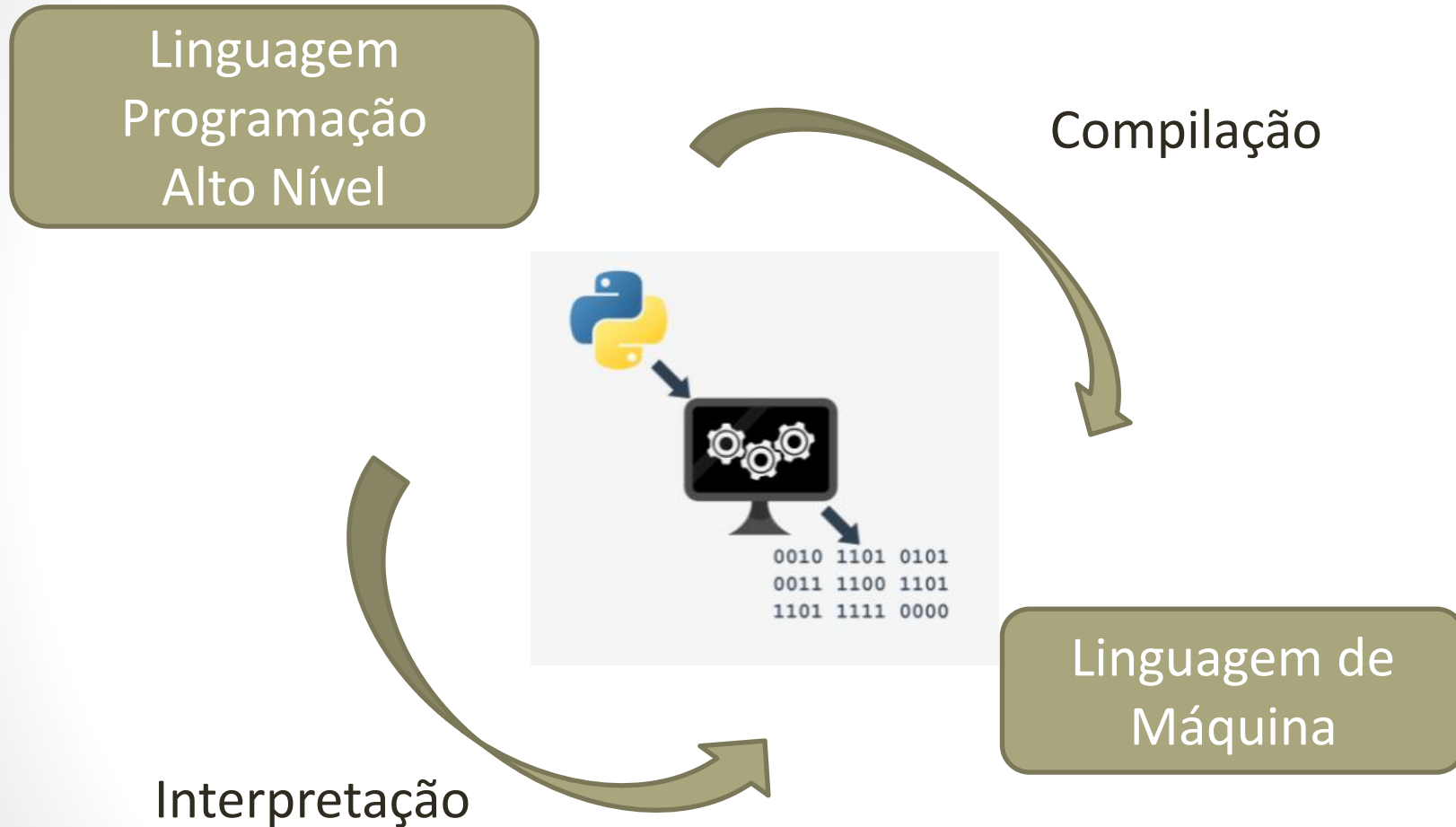


Linguagem de
Máquina



Linguagem de Programação de Alto Nível
Source Code – Código Fonte

Conceitos Básicos



Fonte: Cisco Networking Academy e Python Academy

Conceitos Básicos

COMPILAÇÃO - o source program é traduzido uma vez (no entanto, este ato deve ser repetido sempre que modificar o source code) obtendo um ficheiro (por exemplo, um ficheiro .exe se o código se destinar a ser executado no MS Windows) contendo o machine code; agora pode distribuir o ficheiro por todo o mundo; o programa que executa esta tradução chama-se compilador ou tradutor;

INTERPRETAÇÃO - você (ou qualquer utilizador do código) pode traduzir o source program cada vez que este tem de ser executado; o programa que executa este tipo de transformação chama-se intérprete, pois interpreta o código cada vez que se pretende executá-lo; também significa que não pode simplesmente distribuir o source code tal como está, porque o utilizador final também precisa do intérprete para o executar.

Fonte: Cisco Networking Academy e Python Academy

Conceitos Básicos

Interpretador

- ☐ Verifica as linhas do código fonte no que diz respeito as regras da linguagem
- ☐ Read-Check-Execute
- ☐ Pode executar parte do programa antes de apresentar o erro
- ☐ Mensagens de erro podem não ser totalmente claras

Python é uma linguagem interpretada

Python



Guido van Rossum (Haarlem, 31 de janeiro de 1956 [1]) é um matemático e programador de computadores holandês, mais conhecido por ser o autor da linguagem de programação Python (1989).

Python



Na década de 1970, a BBC tinha um programa de TV popular do qual van Rossum era um grande fã chamado Monty Python's Flying Circus, ou apenas Monty Python para os íntimos.

Python

Objetivos do Python

Em 1999, Guido van Rossum definiu os seus objetivos para o Python:

- uma linguagem **fácil e intuitiva**, tão poderosa como a dos principais concorrentes;
- de **open source**, para que qualquer pessoa possa contribuir para o seu desenvolvimento;
- código que seja tão **compreensível** como o inglês simples;
- **adequado para tarefas quotidianas**, permitindo tempos de desenvolvimento curtos.

Python

Porque não Python?

Apesar da popularidade crescente de Python, ainda existem alguns nichos onde o Python está ausente, ou raramente é visto:

- **programação de baixo nível** (por vezes chamada programação "close to metal"): se quiser implementar um condutor ou motor gráfico extremamente eficaz, não utilizaria Python;
- **aplicações para dispositivos móveis**: embora este território ainda esteja à espera de ser conquistado pelo Python, é muito provável que um dia tal venha a acontecer.

Python – Ambiente de Desenvolvimento

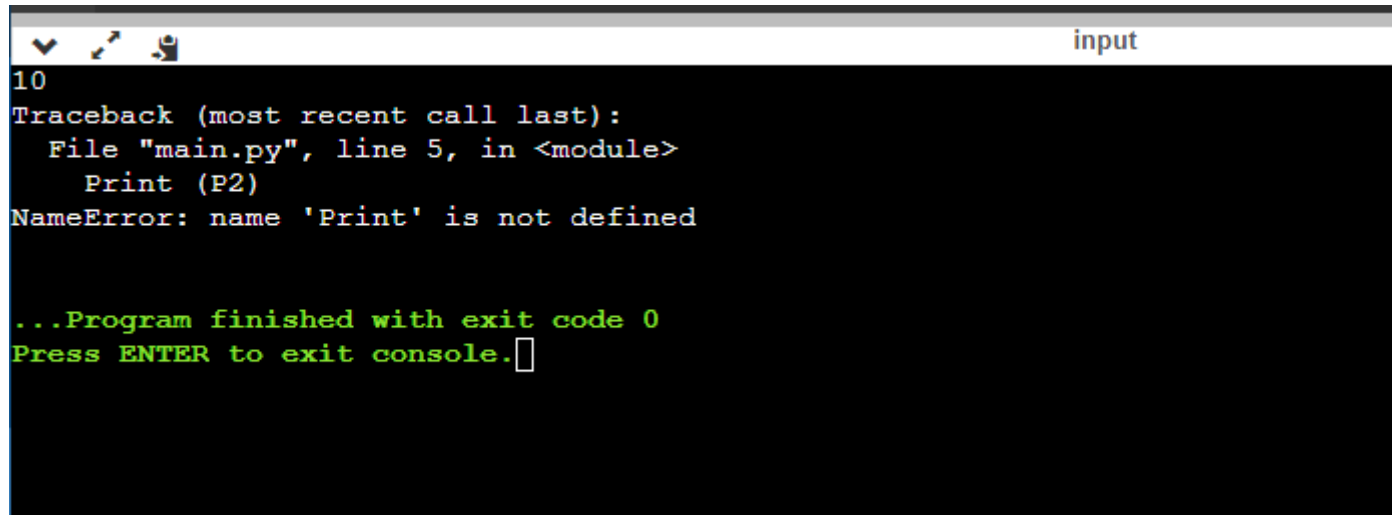
IDLE – Integrated Development and Learning Environment



[GDB online Debugger | Compiler - Code, Compile, Run, Debug online C, C++ \(onlinegdb.com\)](https://onlinegdb.com)

Python

Tratamento de Erro



The screenshot shows a Python console window titled 'input'. The output is as follows:

```
10
Traceback (most recent call last):
  File "main.py", line 5, in <module>
    Print (P2)
NameError: name 'Print' is not defined

...Program finished with exit code 0
Press ENTER to exit console.
```

- o **traceback** (que é o caminho que o código percorre através de diferentes partes do programa - pode ignorá-lo por agora, uma vez que está vazio num código tão simples);
- a **localização do erro** (o nome do ficheiro contendo o erro, o número da linha e o nome do módulo); nota: o número pode ser enganador, uma vez que o Python normalmente mostra o local onde primeiro se notam os efeitos do erro, não necessariamente o erro em si;
- o **conteúdo da linha errada**; nota: a janela do editor IDLE não mostra os números das linhas, mas mostra a localização atual do cursor no canto inferior direito; use-a para localizar a linha errada num source code longo;
- o **nome do erro** e uma breve explicação.

Python

Outros ambientes de Desenvolvimento



Visual Studio Code



Introdução Python

Variáveis

- Espaços de memória que designamos com uma nome e armazenamos valores.
- Estes valores podem ser modificados e utilizados em qualquer ponto do programa.
- Atribuímos um valor a uma variável por meio do sinal de igualdade



Introdução Python

Regras para os nomes das Variáveis

- Iniciar com uma letra (maiúscula ou minúscula) ou (_)
- Pode conter números e o símbolo sublinha (_)
- Não pode ser igual a nenhuma palavra reservada

Nome	Válido	Observações
A1 ou a1	Sim	Iniciar com letras Maiúsculas/Minúsculas
1A	Não	Iniciar com número
Minha_Variável	Sim	É permitido (_) e pode ter acento
_B	Sim	É permitido iniciar com (_)
Salário Médio	Não	Não pode conter espaços

Introdução Python

Tipos de Dados

- Em Python não declaramos a variável, ou seja, não dizemos antecipadamente qual o tipo de dado que ela pode receber. No Python, assim que atribuímos um valor a variável ela passa a ser do tipo do dado que foi atribuído a ela.
- Tipos e exemplo:
 - `Minha_Variável = 3` será do tipo `int` – números inteiros
 - `Minha_Variável = 3.0` será do tipo `float` - números reais
 - `Minha_Variável = 3+2j` será do tipo `complex` – números complexos
 - `Minha_Variável = True` será do tipo `bool` – booleano
 - `Minha_Variável = [1,2,3]` será do tipo `list` - lista
 - `Minha_Variável = "Python"` será do tipo `str` – texto (string)
 - `Minha_Variável = "3"` também será do tipo `str` (string)

Introdução Python

Operadores Aritméticos

Operação	Resultado
$x + y$	Soma de x e y
$x - y$	Diferença de x e y
$x * y$	Produto de x e y
x / y	Quociente de x e y
$x // y$	Resultado inteiro da divisão de x e y
$x \% y$	Resto de x / y
$x ** y$	x elevado a y

Em Python o separador decimal é o ponto

Introdução Python

Prioridade em expressões aritméticas

Prioridade:

1º) ()

2º) potenciação e radiciação

3º) multiplicação, divisão e módulo (resto da divisão)

4º) soma e subtração

Expressões com operadores de **mesma prioridade** são realizadas da esquerda para a direita.

Exceção da exponenciação que é feita da **direita para a esquerda (Python)**.

Exemplo:

`Exp = 2**2**3`

primeiro é realizada a operação `2**3` que resulta 8 e depois `2**8 = 256`

Introdução Python

Funções Matemáticas

Operação	Resultado
<code>abs(x)</code>	Valor absoluto x
<code>int(x)</code>	Converte x para inteiro
<code>float(x)</code>	Converte x para real
<code>complex(re, im)</code>	Número complex com a parte real re , parte imaginária im , im padrão é zero.
<code>c.conjugate()</code>	conjugado do número complexo c
<code>divmod(x, y)</code>	A tupla $(x // y, x \% y)$
<code>pow(x, y)</code>	x elevado a y

Introdução Python

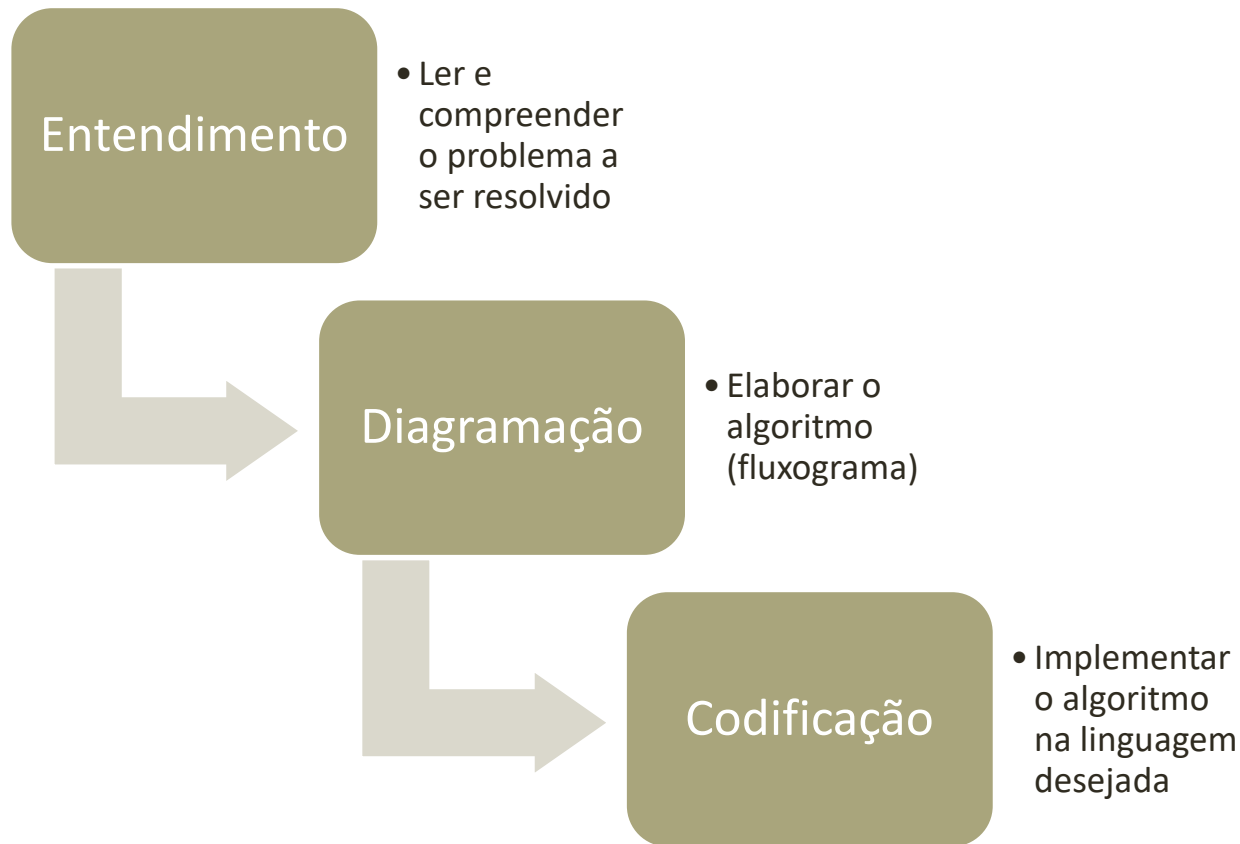
Iniciando a programação – estrutura sequencial



Algumas boas práticas profissionais:

- Um problema nunca vem totalmente definido e especificado – aborde-o com atenção e profundidade. Busque as informações que faltam.
- Faça um cabeçalho para seus programas.
- Comente seus programas – você não trabalha sozinho.
- Preocupe-se com a estética e bom censo para facilitar a leitura do código.
- Seu usuário é seu cliente – trabalhe para dar inteligibilidade e clareza nas comunicações de entrada e saída com o usuário.

Relembrando – passos para resolução de um problema




Introdução Python

ENTRADA de DADOS

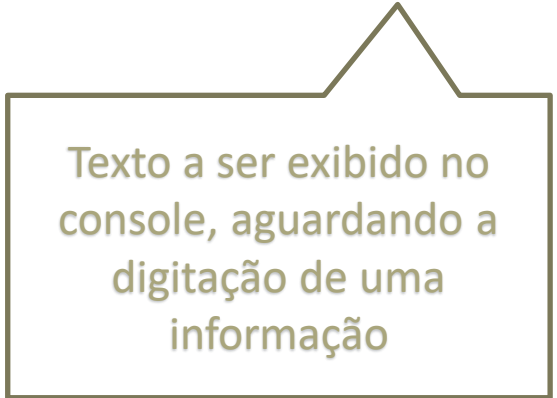
input – sempre recebe o valor **como texto (string)**

- input(“Entre com a informação a ser registrada: ”)
- Podemos atribuir o valor capturado pelo console em uma variável:

```
V1 = input("Entre com a informação a ser registrada =")
```



Variável que vai
receber a
informação



Texto a ser exibido no
console, aguardando a
digitação de uma
informação

Introdução Python

Entrada de Dados - Exemplos

```
V1 = input("Entre com a informação a ser registrada = ")
```

Preparado para receber string – caracteres
alfanuméricos

```
V1 = int ( input("Entre com a informação a ser registrada = ") )
```

Preparado para receber somente números
inteiros

```
V1 = float ( input("Entre com a informação a ser registrada = ") )
```

Preparado para receber somente números
reais (inteiros e decimais)

Introdução Python

Saída de Dados - print

Função print – algumas técnicas

1. Básico , textos e variáveis separados por vírgula
2. Marcadores de posição
3. Método .format
4. F-String

Neste material utilizaremos as três primeiras , a F-String é uma técnica mais recente e mais sofisticada, vamos estudá-la mais a frente no curso.

Introdução Python

Saída de Dados - print

Função print - exemplo

```
V1 = input("Entre com a informação a ser registrada =")
```

```
print("\n")           # \n comando para pular linha
```

```
print (V1)            # imprime o valor da variável V1
```

Parâmetro sep da função print:

```
print('Dia', 'Mês', 'Ano', sep='/')
```

```
print('ontem', 'Hoje', 'Amanhã', sep='-')
```

```
print("B", "n", "n", ".", sep='a')
```

Obs: se nada informarmos o padrão é um espaço em branco

O parâmetro “sep”
junta as
informações que
estão separados
pelas vírgulas
conforme o que for
especificado nele

Introdução Python

Saída de Dados

print com parâmetro “End”

Exemplo com fim de linha sem nenhum caractere

```
print('Vamos estudar Na ', end='')  
print('Python Academy')
```

Abre e fecha aspas,
ou seja, nenhum
caractere

Exemplo com fim de linha igual à ->

```
print('As rosas são', end=' -> ')  
print('Vermelhas')
```

O parâmetro “end”
junta as instruções
prints conforme o
que for
especificado nele

Exemplo com fim de linha igual à :

```
print("Quantidade", end=': ')  
print(40)
```

Obs: se nada informarmos o padrão é uma quebra de linha (\n)

Introdução Python

Saída de Dados - print

07/03/2024

```
1 print (''  
2 Quantidade de Pessoas   Diária   Diária  
3 no Apartamento         tipo 1(R$) tipo2 (R$)  
4      1      20,00      25,00  
5      2      28,00      34,00  
6      3      35,00      42,00  
7      4      42,00      50,00  
8      5      48,00      57,00  
9      6      53,00      63,00 ''')  
10
```

Podemos usar aspas triplas para organizar o print de tabelas

```
Quantidade de Pessoas   Diária   Diária  
no Apartamento         tipo 1(R$) tipo2 (R$)  
      1      20,00      25,00  
      2      28,00      34,00  
      3      35,00      42,00  
      4      42,00      50,00  
      5      48,00      57,00  
      6      53,00      63,00
```

```
...Program finished with exit code 0  
Press ENTER to exit console.
```

Introdução Python

Marcador de posição

É usado na composição de textos e variáveis.

O símbolo **%** é utilizado para indicar a impressão do conteúdo de uma variável, indica que temos um marcador de posição.

Exemplo:

```
v1=int(input("Digite o primeiro valor "))
v2=int(input("Digite o segundo valor "))
soma = v1 + v2
print("Soma de %i com %i = %i " % (v1,v2,soma) )
```

Variáveis a serem inseridas nos marcadores de posição na ordem em que aparecem

A alternativa seria:

```
print("Soma de ",v1,"com e",v2,"=",soma)
```

Marcador	Tipo
%d ou %i	Números inteiros
%s	Strings
%f	Números decimais (float)

Introdução Python

Exemplo

- 1) Faça o algoritmo e depois escreva um código que receba dois números pelo teclado, faça a soma e depois imprima no console este valor.
- 2) Altere o programa para realizar as operações abaixo utilizando estes dois números:

Soma de x e y

Diferença de x e y

Produto de x e y

Quociente de x e y

Divisão com resultado inteiro de x / y

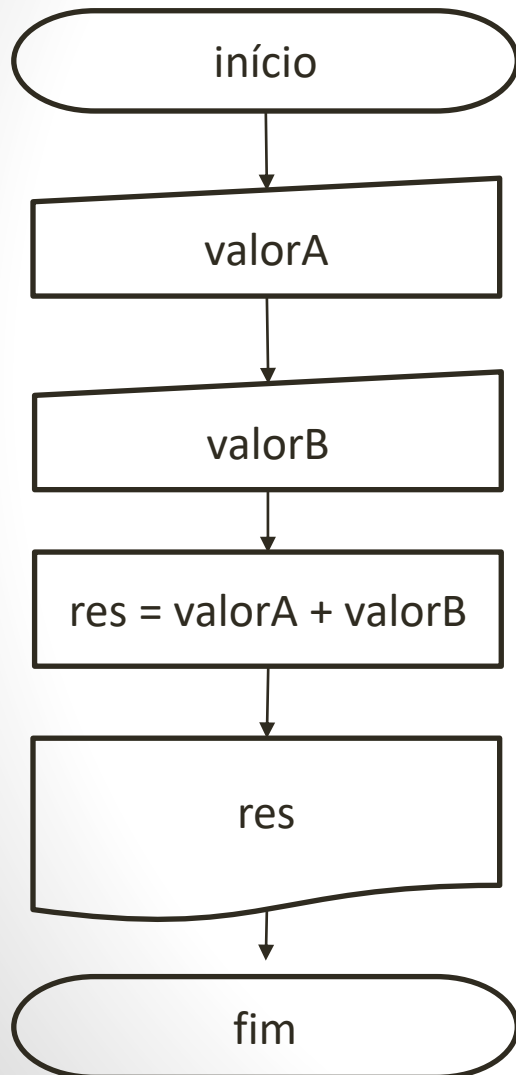
Resto da divisão x / y

x elevado a y

Introdução Python

Exemplo

Fluxograma



Python

```
valorA = int(input("Digite o 1º valor"))
valorB = int(input("Digite o 2º valor"))
res = valorA + valorB
print("Resultado = ", res)
#Usando marcador de posição
print("Resultado = %i ", %(res))
```

Introdução Python

Exemplo

```
v1=int(input("Digite o primeiro valor "))
v2=int(input("Digite o segundo valor "))
soma = v1 + v2
subtrai = v1 - v2
multiplica = v1 * v2
divide = v1 / v2
x,y = divmod(v1, v2)
w = pow (v1,v2)

print(" A soma de %i com %i = %i " % (v1,v2,soma))
print(" A subtração de %i com %i = %i " % (v1,v2,subtrai))
print(" A multiplicação de %i por %i = %i " % (v1,v2,multiplica))
print(" A divisão de %i por %i = %i " % (v1,v2,divide))
print(" A parte inteira de %i dividido por %i é igual a %i e o resto da
divisão é igual a %i " % (v1,v2,x,y))
print("%i elevado a %i = %i " % (v1,v2,w))
```

Introdução Python

Marcador de posição

Formatação de números decimais:

Quando trabalhamos com Floats (marcador **f**) podemos definir a quantidade de casas decimais informando-a entre a **%** e o **f**:
O Python fará arredondamento.

Uma alternativa é usar a função `round()`

Introdução Python

Marcador de posição

```
##Empresa
##Nome do Programador
##Data
##Programa: Somando valores
v1=float(input("Digite o primeiro valor: "))
v2=float(input("Digite o segundo valor: "))
soma = (v1 + v2)
print("A soma de R$ %.2f com R$ %.2f é igual a R$ %.2f " % (v1,v2,soma))
```

```
Digite o primeiro valor: 45.8765
Digite o segundo valor: 50.7864
A soma de R$ 45.88 com R$ 50.79 é igual a R$ 96.66

...Program finished with exit code 0
Press ENTER to exit console.█
```

Usa **2** casas decimais
(arredondamento)

Exemplo de
saída

Introdução Python

Usando a função Round

```
##Empresa
##Nome do Programador
##Data
##Programa: Somando valores
v1=float(input("Digite o primeiro valor: "))
v2=float(input("Digite o segundo valor: "))
v1=round(v1,2) # arredondamos o valor e registramos na mesma variável
v2=round(v2,2)
soma = v1 + v2
print("A soma de R$",v1, "com R$ ",v2, " é igual a R$ ", soma)
```

Usa 2 casas decimais (arredondamento)

```
Digite o primeiro valor: 45.8765
Digite o segundo valor: 50.7864
A soma de R$ 45.88 com R$ 50.79 é igual a R$ 96.66

...Program finished with exit code 0
Press ENTER to exit console.
```

Exemplo de
saída

Introdução Python

Usando o método .format

```
##Empresa
##Nome do Programador
##Data
##Programa: Somando valores
v1=float(input("Digite o primeiro valor: "))
v2=float(input("Digite o segundo valor: "))
soma = v1 + v2
print("A soma de R$ {:.2f} com R$ {:.2f} é igual a R$ {:.2f}".format(v1,v2,soma))
```

Usa **2** casas decimais (arredondamento)
O formato {:.2f} é passado para variável dentro do .format , na sequência em que aparecem

```
Digite o primeiro valor: 45.8765
Digite o segundo valor: 50.7864
A soma de R$ 45.88 com R$ 50.79 é igual a R$ 96.66

...Program finished with exit code 0
Press ENTER to exit console. █
```

Exemplo de saída

Bibliografia

MENEZES, N. N. C., Introdução a Programação com Python: algoritmos e lógica de programação para iniciantes. Editora Novatec, 3ª edição, 2019.

FORBELLONE, A. L. V., EBERSPACHER, H. F., Lógica de Programação: A construção de algoritmos e estruturas de dados, São Paulo: Pearson, 2013.

