



Prof. Me. Massaki de O. Igarashi

massaki.igarashi@anchieta.br



Agenda desta aula!

01

- Fluxogramas (Continuação)

02

- Variáveis

03

- Operadores

04

- Sistema / Computador







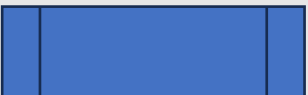
05

- Analogia (Computação x Cozinha)

06

- Spoiler da próx. aula

Norma ISO 5807 – Simbologia Fluxograma

SÍMBOLO	SIGNIFICADO	DESCRIÇÃO
	Terminal (<i>Terminator</i>)	Definição de Início e Fim do fluxo lógico de um programa. Também usado na definição de sub-rotinas de procedimento ou de função.
	Entrada Manual (<i>Manual Input</i>)	Representa a entrada manual de dados , normalmente efetuada em um teclado conectado diretamente ao console do computador.
	Processamento (<i>Process</i>)	Representa a execução de uma operação ou grupo de operações que estabelecem o resultado de operação lógica ou matemática.
	Exibição (<i>Display</i>)	Representa a execução da operação de saída visual de dados em um monitor de vídeo conectado ao console do computador.
	Decisão (<i>Decision</i>)	Este símbolo representa o uso de desvios condicionais para outros pontos do programa de acordo com determinadas situações.
	Preparação (<i>Preparation</i>)	Representa a modificação de instruções ou grupo de instruções existentes em relação à ação de sua atividade subsequencial.
	Processo Predefinido (<i>Predefined Process</i>)	Definição de um grupo de operações estabelecidas como uma sub-rotina de processamento anexa ao diagrama de blocos

MANZANO, José Augusto Navarro G.; OLIVEIRA, Jayr Figueiredo de. **Algoritmos - Lógica para Desenvolvimento de Programação de Computadores**. [Digite o Local da Editora]: Editora Saraiva, 2019. *E-book*. ISBN 9788536531472. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788536531472>. Acesso em: 10 mar. 2024.

Mais Exercícios para praticar em casa!

5) Criar a narrativa, o pseudocódigo e o fluxograma para um programa que receba o raio (r) de uma esfera, e em seguida calcule e exibir o seu volume (v) desta esfera.

Narrativa:

Atribuir $PI = 3.14159265358979323846$

Ler r

Calcular $v = (4/3.0) * PI / (r * r * r)$

Exibir r

Pseudocódigo:

Início:

Real: r, v

Real $PI = 3.14159265358979323846$

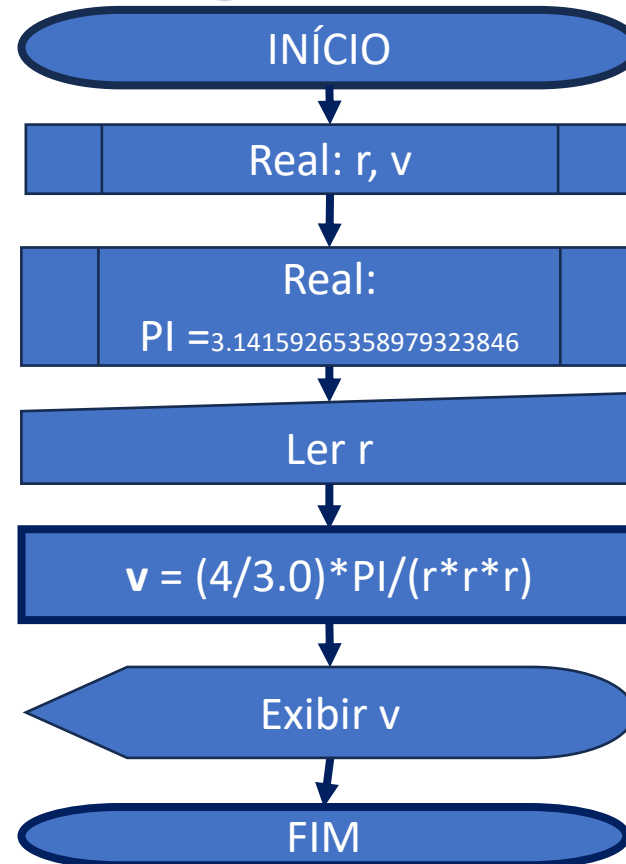
Ler r

Calcular $v = (4/3.0) * PI / (r * r * r)$

Exibir v

FIM

Fluxograma:



Mais Exercícios para praticar em casa!

6) Criar a narrativa, o pseudocódigo e o fluxograma para um programa que receba um ângulo (g) em graus, e converta este valor para radianos (r).

Narrativa:

Atribuir $\text{PI} = 3.14159265358979323846$

Ler g

Calcular $r = g * \text{PI} / 180.0$

Exibir r

Pseudocódigo:

Início:

Real: g, r

Real $\text{PI} = 3.14159265358979323846$

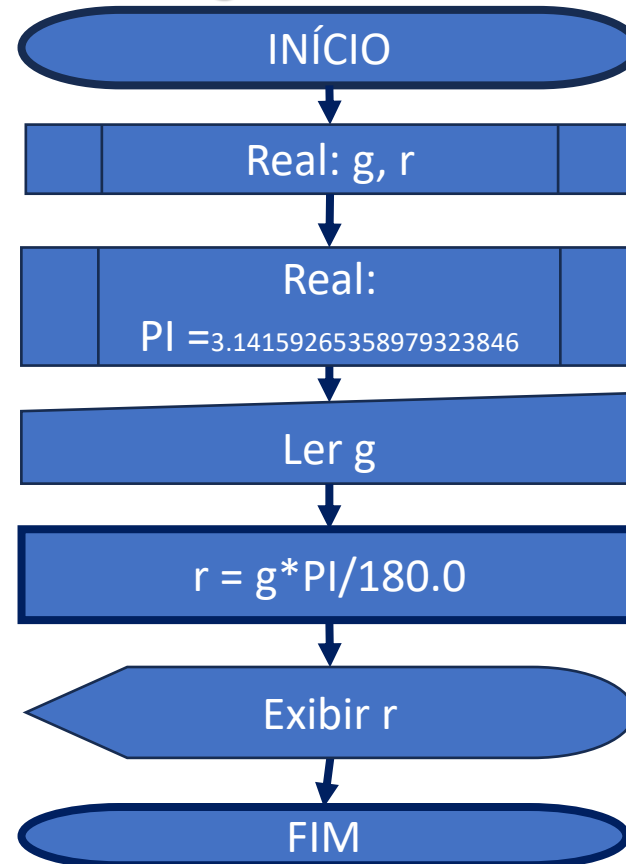
Ler g

Calcular $r = g * \text{PI} / 180.0$

Exibir r

FIM

Fluxograma:



Mais Exercícios para praticar em casa!

7) Criar a narrativa, o pseudocódigo e o fluxograma para um programa que receba o valor de um ângulo em radianos, e converta este valor para graus.

Narrativa:

Atribuir $\text{PI} = 3.14159265358979323846$

Ler r

Calcular $g = r * 180.0 / \text{PI}$

Exibir g

Pseudocódigo:

Início:

Real: g, r

Real $\text{PI} = 3.14159265358979323846$

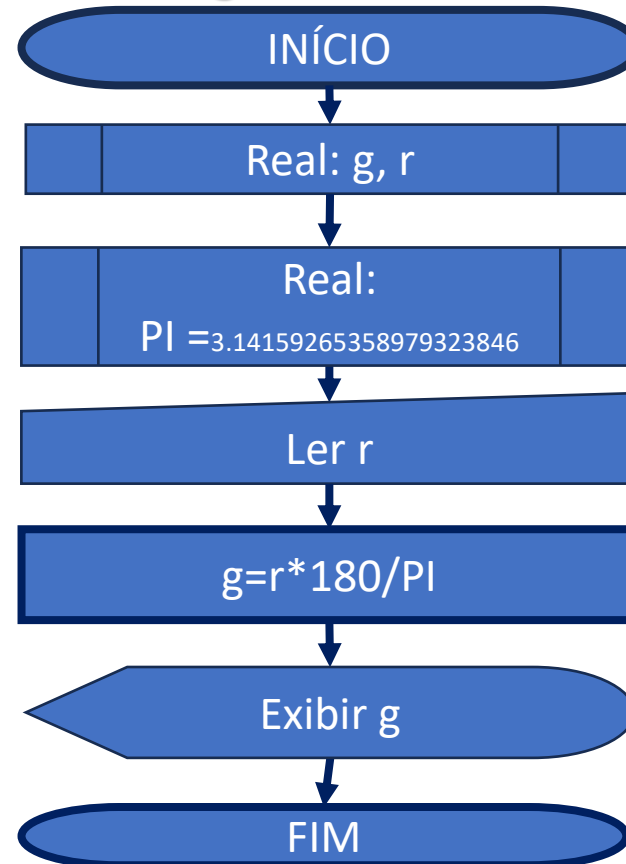
Ler r

Calcular $g = r * 180.0 / \text{PI}$

Exibir g

FIM

Fluxograma:



Mais Exercícios para praticar em casa!

8) Criar a narrativa, o pseudocódigo e o fluxograma para um programa que receba ângulo em radianos (Rad), e converta este valor para grados(Gr). Lembrar que 400 grados equivalem a 2π .

Narrativa:

Atribuir $PI = 3.14159265358979323846$

Ler Rad

Calcular $Gr = 400 * Rad / (2 * PI)$

Exibir Gr

Pseudocódigo:

Início:

Real: Rad, Gr

Real $PI = 3.14159265358979323846$

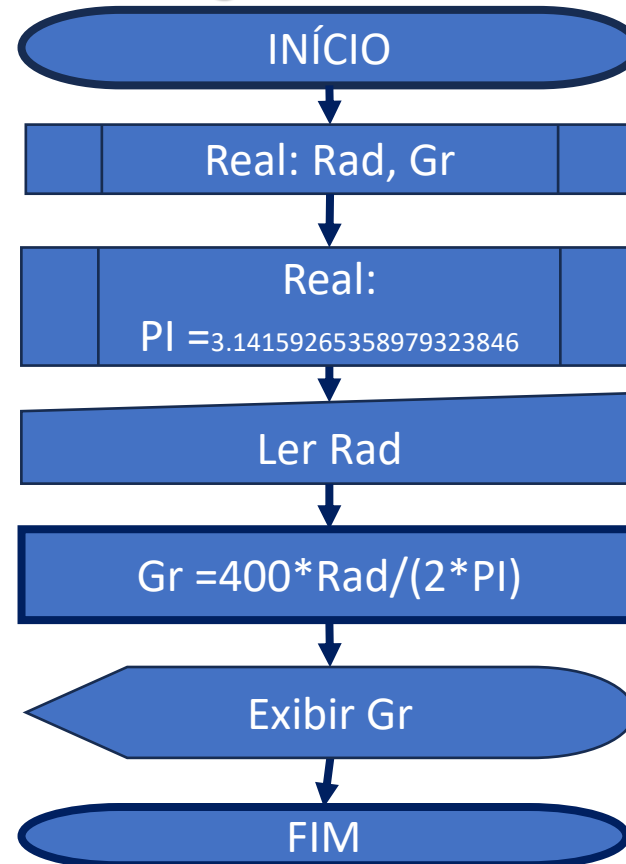
Ler Rad

Calcular $Gr = 400 * Rad / (2 * PI)$

Exibir Gr

FIM

Fluxograma:



IDENTIFICADORES: *VARIÁVEIS*

Nas últimas aulas aprendemos sobre algoritmo e suas formas de representação, inclusive sobre a **função do algoritmos de representar uma solução para um problema** no mundo real escrito em linguagem que o computador compreende. Com esse mesmo pensamento, podemos entender que cada item tratado nesse problema pode ser representado por uma **variável**.

IDENTIFICADORES: *VARIÁVEIS*

Pode-se dizer que **você certamente irá lidar com variáveis na maior parte do seu tempo como profissional de programação**, mas as variáveis não costumam vir sozinhas; elas costumam aparecer com **operadores aritméticos e relacionais**, principalmente quando o programa tem a função de processar algum cálculo específico ou algum sinal de entrada para tomar uma determinada decisão.

IDENTIFICADORES: *VARIÁVEIS*

Em programação, um **identificador** é um nome atribuído a uma **variável** de memória, uma estrutura de dados, uma classe, um objeto, um procedimento, uma função, um comando ou palavra reservada da linguagem.

Variável é tudo que está **sujeito a variações**, que é **incerto, instável ou inconstante**. Quando se fala de **computadores**, é preciso ter em mente que o volume de dados a serem tratados é grande e diversificado. Dessa forma, os **dados a serem processados são bastante variáveis**.

Todo dado a ser armazenado na memória de um computador deve ser previamente identificado segundo seu tipo, ou seja, primeiramente é necessário saber o tipo do dado para depois fazer seu armazenamento adequado. **Armazenado o dado, ele pode ser utilizado e processado a qualquer momento**.



Regras para identificadores:

Um identificador pode ter até 32 caracteres de comprimento.

❖ O 1º caractere deve ser uma letra do alfabeto.

❖ Os demais caracteres podem ser letras, números ou sinal *underscore* (`_`)

❖ Não usar sinais de pontuação, caracteres acentuados, cedilha (ç) ou espaço em branco.

❖ Exemplos de identificadores: **Nome, Nome_Cliente, Nome_Completo, Data, Data_Emissao, Mensagem, Aviso, 1aSemana, Endereco, a, b, c, ..., X, Y, Z.**

VARIÁVEIS

Uma variável representa um contêiner ou espaço na memória física ou virtual de um computador, onde diferentes tipos de dados (valores) são armazenados durante a execução de um programa. A cada variável é atribuído um nome descritivo ou identificador que se refere ao valor salvo.

Elas são importantes para o funcionamento de programas e aplicações que lidam com cálculos, condições, repetições e qualquer outro dado mutável durante o seu funcionamento.

OPERADORES

Operadores aritméticos: são símbolos especiais que realizam operações matemáticas básicas

Operadores Aritméticos

OPERAÇÃO	SÍMBOLO	CÓDIGO EM PYTHON	CÓDIGO EM C++	DESCRIÇÃO
ADIÇÃO	+	$x + y$	$x + y$	Efetua a soma $x + y$
SUBTRAÇÃO	-	$x - y$	$x - y$	Efetua a subtração $x - y$
MULTIPLICAÇÃO	*	$x * y$	$x * y$	Efetua a multiplicação $x * y$
DIVISÃO	/	x / y	x / y	Efetua a divisão x / y
DIVISÃO INTEIRA	//	$x // y$	Operador inexistente	Efetua a divisão truncada $x // y$ Ou seja, $5 // 2 = 2$, pois trunca o ponto flutuante 0.5
POTÊNCIA	**	$x ** y$	Operador inexistente	Eleva um número x a um expoente y (x^y)
RESTO DA DIVISÃO	%	$x \% y$	$x \% y$	Obtém o resto da divisão entre dois inteiros

OPERADORES

Operadores Relacionais

OPERAÇÃO	SÍMBOLO	CÓDIGO EM PYTHON	CÓDIGO EM C++	DESCRIÇÃO
Menor que	<	<code>x < y</code>	<code>x < y</code>	Verdadeiro se x Menor y
Maior que	>	<code>x > y</code>	<code>x > y</code>	Verdadeiro se x Maior y
Igual	==	<code>x == y</code>	<code>x == y</code>	Verdadeiro se x Igual a y
ATRIBUIÇÃO	=	<code>X = 2</code>	<code>X = 2</code>	Atribui um valor ou resultado de cálculo à uma constante
Diferente	!=	<code>x != y</code>	<code>x != y</code>	Verdadeiro se x Diferente de y
Menor igual	<=	<code>x <= y</code>	<code>x <= y</code>	Verdadeiro se x Menor ou Igual a y
Maior Igual	>=	<code>x >= y</code>	<code>x >= y</code>	Verdadeiro se x Maior ou Igual a y

O símbolo **=** é considerado operador de atribuição; usado para atribuir valores **para constantes OU atribuir cálculos**

Exemplo:
`X = 2`
`Y = 3`
`soma = X + Y`

IDENTIFICADORES: *VARIÁVEIS*

As operações com os operadores aritméticos básicos podem ser feitas de forma a armazenar os cálculos realizados primeiramente em variáveis e em seguida exibí-los. Isto é um procedimento muito comum na computação; já que as variáveis tem seus valores armazenados em posições da memória RAM do computador.

Mas antes de sairmos processando e fazendo uso de variáveis e operadores, é preciso fazer uma pausa para responder duas perguntas:

O que é um sistema?

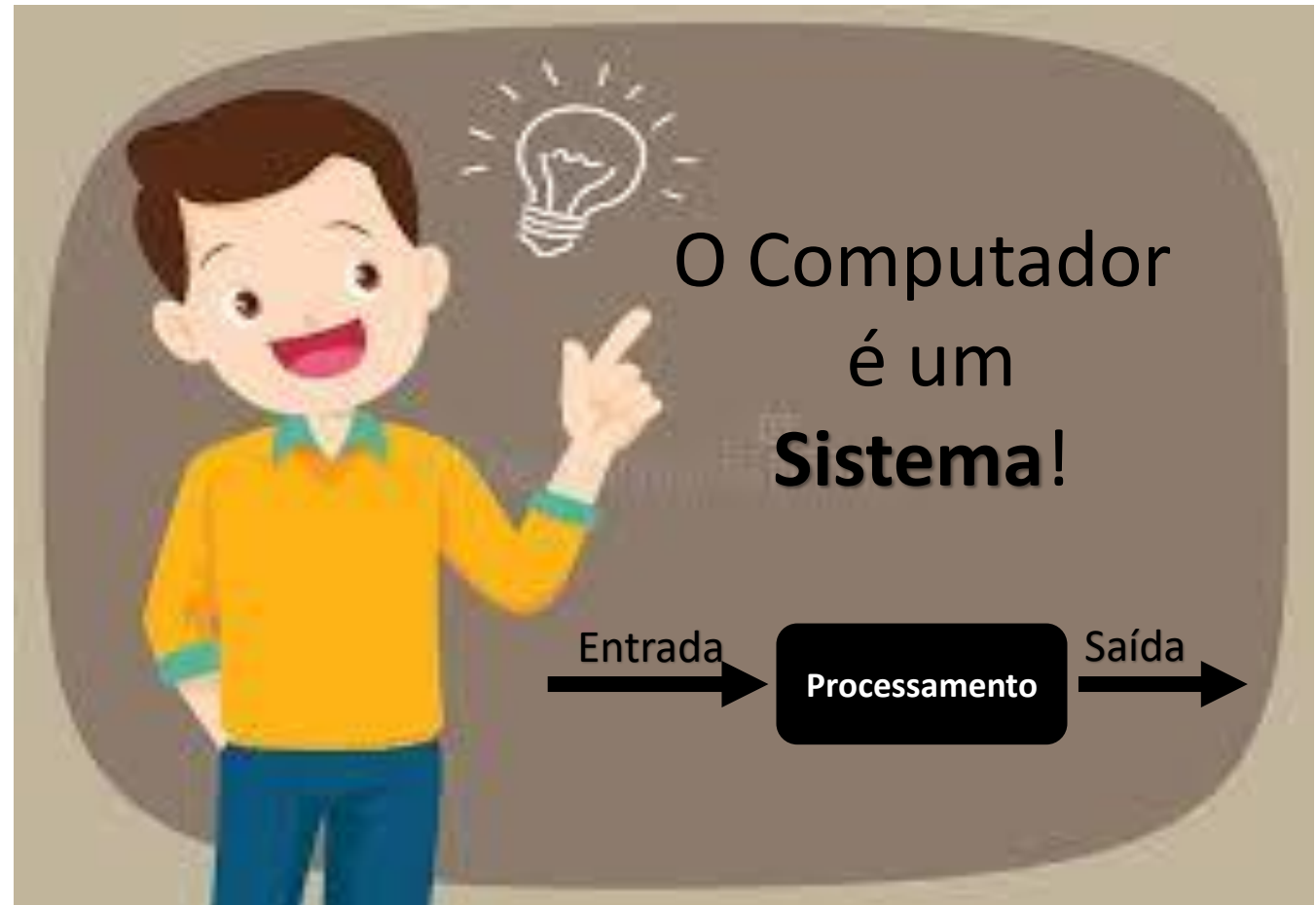
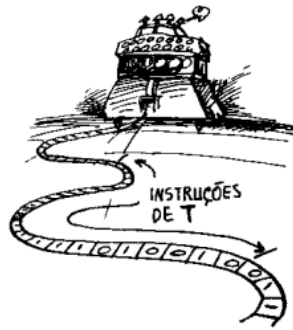
O computador é um sistema?

Introdução: Você sabe o que é um Sistema?

Uma super “dica”!

Toda linguagem de programação tem:

1. Comandos de ENTRADA
2. OPERADORES MATEMÁTICOS
3. Comandos de CONDIÇÃO
4. Comandos de REPETIÇÃO
5. Comandos de SAÍDA



Introdução: O Sistema processando um programa

Exemplo 01: Crie um código em linguagem Python p/ Solicitar 2 valores a e b e em seguida calcular e exibir o resultado.

Entrada(s)

```
a = input("Digite a: ")  
b = input("Digite b: ")
```

Processamento

```
soma = float(a) + float(b)
```

Saída

```
print(soma)
```

1º Passo: Criar a Narrativa ou pseudo-código

Ler valor a

Ler valor b

Calcular $soma = a + b$

imprimir soma

2º Passo: criar o Algoritmo do programa

início:

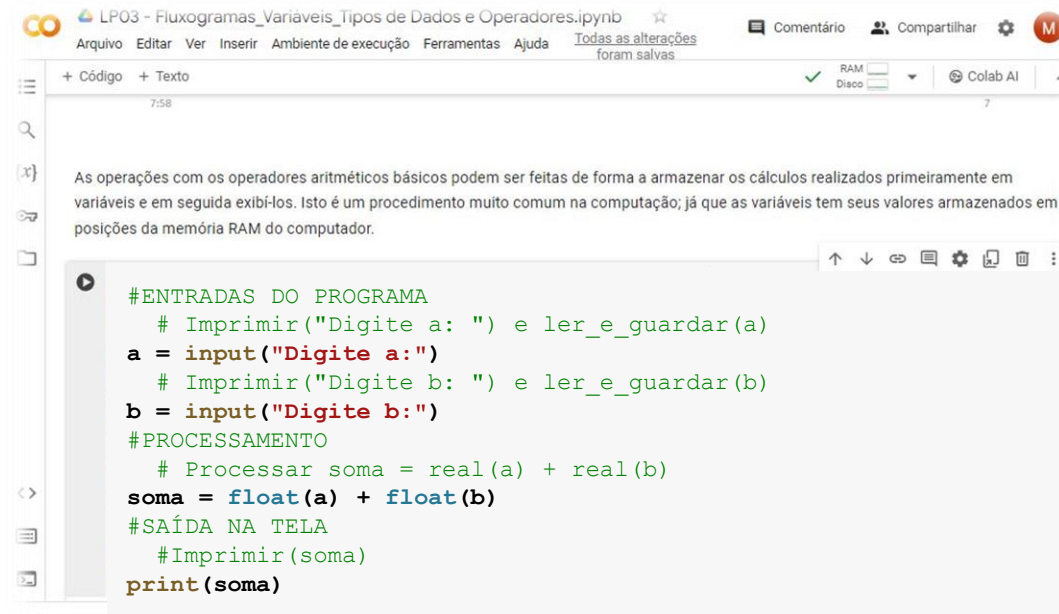
imprimir("Digite a: ") e ler_e_guardar(a)

imprimir("Digite b: ") e ler_e_guardar(b)

Processar $soma = real(a) + real(b)$

Imprimir(soma)

Fim



```
LP03 - Fluxogramas_Variáveis_Tipos de Dados e Operadores.ipynb  
Arquivo Editar Ver Inserir Ambiente de execução Ferramentas Ajuda Todas as alterações foram salvas  
+ Código + Texto  
7:58  
As operações com os operadores aritméticos básicos podem ser feitas de forma a armazenar os cálculos realizados primeiramente em variáveis e em seguida exibi-los. Isto é um procedimento muito comum na computação; já que as variáveis tem seus valores armazenados em posições da memória RAM do computador.  
#ENTRADAS DO PROGRAMA  
# Imprimir("Digite a: ") e ler_e_guardar(a)  
a = input("Digite a:")  
# Imprimir("Digite b: ") e ler_e_guardar(b)  
b = input("Digite b:")  
#PROCESSAMENTO  
# Processar soma = real(a) + real(b)  
soma = float(a) + float(b)  
#SAÍDA NA TELA  
#Imprimir(soma)  
print(soma)
```



Leia com a Câmera
do Smartphone

https://bit.ly/ANC01_LP03

EXEMPLO 02

Narrativa ou pseudo-código

Ler valor a Ler valor b

Calcular soma= $a+b$

Calcular subtracao= $a-b$

Calcular multiplicacao= $a*b$

Calcular divisao= a/b

e divisão Inteira = $a//b$

Calcular potencia= $a**b$

Calcular resto= $a\%b$

imprimir soma

imprimir Subtração

imprimir Multiplicação

imprimir Divisão e Divisão inteira

imprimir Potência

imprimir Resto



**Crie o algoritmo
a partir da
Narrativa ao lado**

```
+ Código + Texto

Exemplo 02) Crie um código de programa em linguagem Python para Solicitar dois
valores a e b e em seguida calcular e exibir o resultado de:

• Soma
• Subtração
• Multiplicação
• Divisão
• Divisão Inteira
• Potência
• Resto

[ ] #ENTRADAS DO PROGRAMA
a = float(input("Digite a:"))
b = float(input("Digite b:"))
#PROCESSAMENTOS
soma = a + b
subtracao = a - b
multiplicacao = a * b
divisao = a / b
potencia = a ** b
resto = a % b
#SAÍDAS NA TELA
print("Soma = " + str(soma))
print("Subtração = " + str(subtracao))
print("Multiplicação = " + str(multiplicacao))
print("Divisão = " + str(divisao))
print("a elevado a b = " + str(potencia))
print("Resto da divisao de a por b = " + str(resto))

7s conclusão: 15:39
```

TIPOS DE DADOS

Saber sobre variável e operadores apenas não basta; você precisa saber sobre tipos de dados para programar!

Tipificar é apenas uma forma de classificar um valor. Um caractere é diferente de um número inteiro e um número inteiro é diferente de um número decimal; que também difere de um booleano. A maioria das linguagens de programação lida com 4 tipos primitivos de dados:

Tipos de dados

Inteiro (do inglês, **integer**)

int → Tipo para definir números inteiros: **8** / **80**

Decimal ou ponto flutuante (do inglês, **float** point)

float → Tipo para definir números decimais: **3.14** / **5.50**

Caracteres (do inglês **String**)

str → Tipo para definir caracteres: **"Exemplo de String"** / **"12"**

Booleano (do inglês, **Bool**)

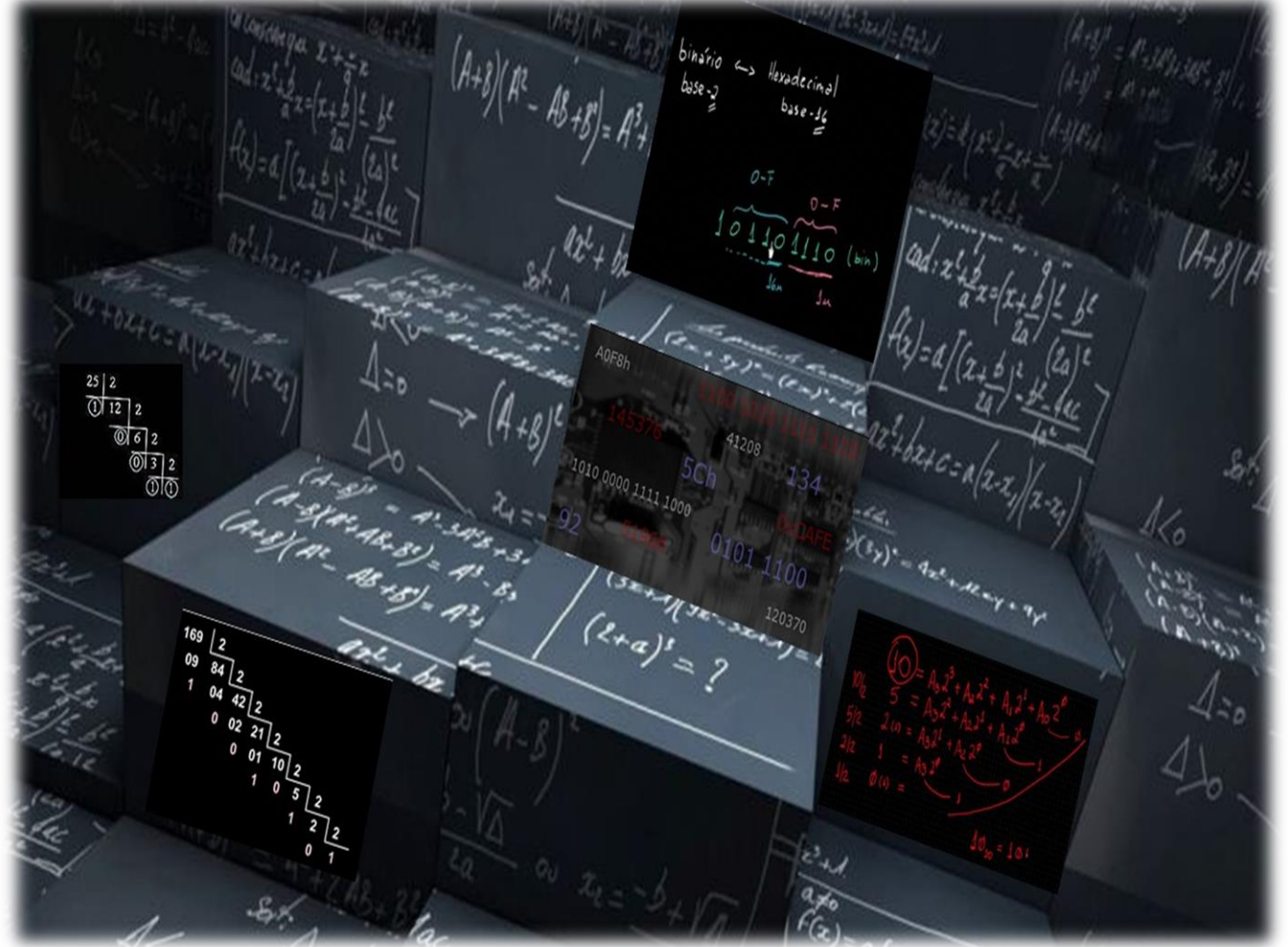
bool → Tipo para definir valores booleanos: **True** / **False**

O tipo de dados
Caracter(String)
sempre virá
entre aspas!

PRÓXIMA AULA



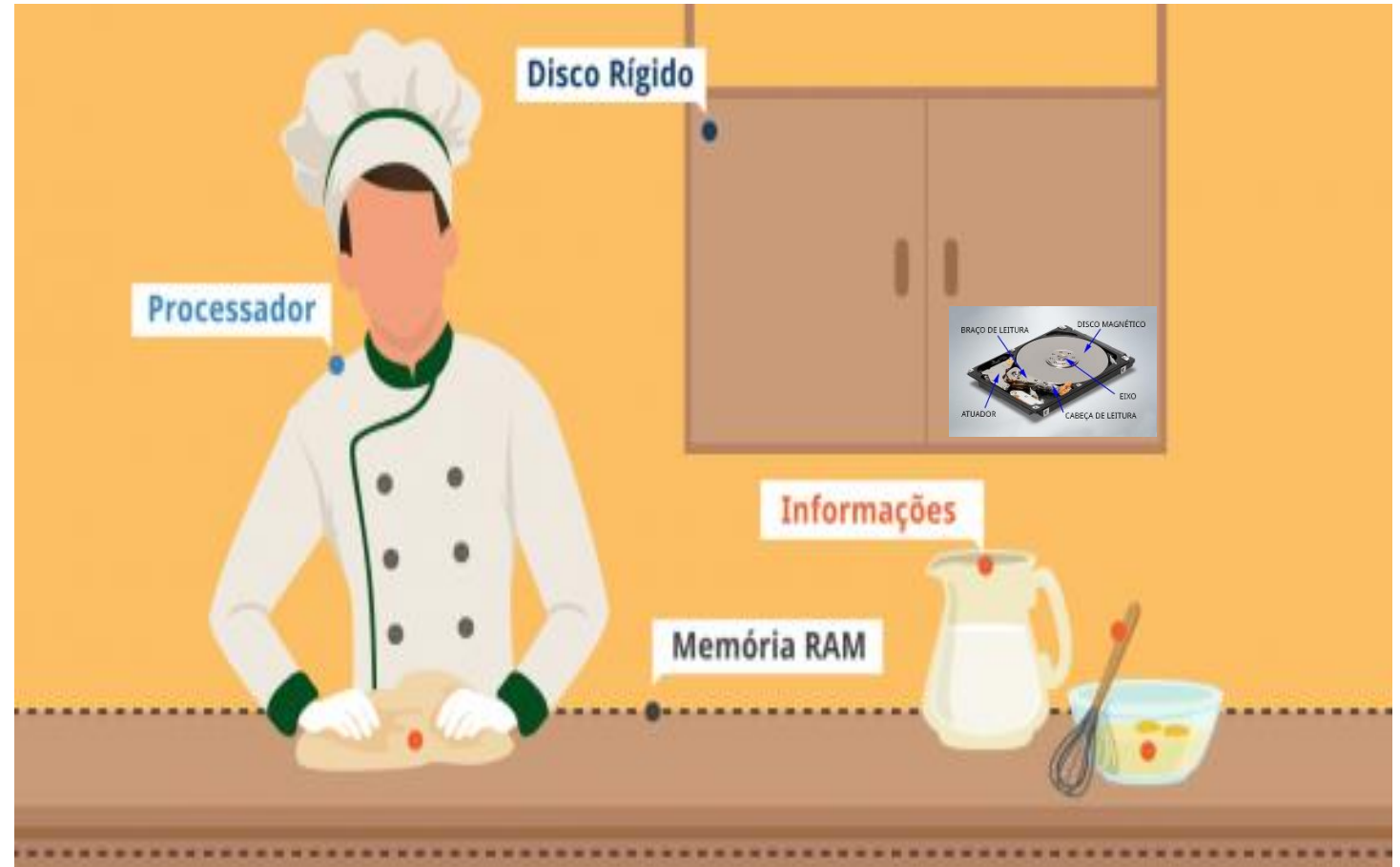
Aprenderemos sobre as bases numéricas e as conversões numéricas



ANALOGIA

Computador x Cozinha

*Mas antes,
vamos fazer
um pouco de
analogia!*



<https://tudosobrehospedagemdesites.com.br/memoria-ram-hospedagem/>

ANALOGIA



A memória RAM é como uma prateleira de dispensa da cozinha; neste caso, cada parte da prateleira é como uma posição de memória e os dados podem ser armazenados individualmente em potes, que nesta analogia entenderemos como armazenados nas variáveis.



DADOS ENDEREÇOS

50H
00H
25H

00H
01H
02H
03H
04H
05H
06H
07H
...

Endereços → posição onde os dados ou informações serão colocados, geralmente expressos em números hexadecimais. Cada endereço pode conter apenas uma informação.

Conteúdo ou dado → informação presente em cada posição de memória, para nosso estudo consideraremos os dados sempre como 8 bits ou 1 byte. Geralmente também são expressos em números hexadecimais.

Computador x Cozinha



Assim como os potes recebem nomes individuais para suas específicas funções na dispensa da cozinha, as variáveis também recebem nomes!

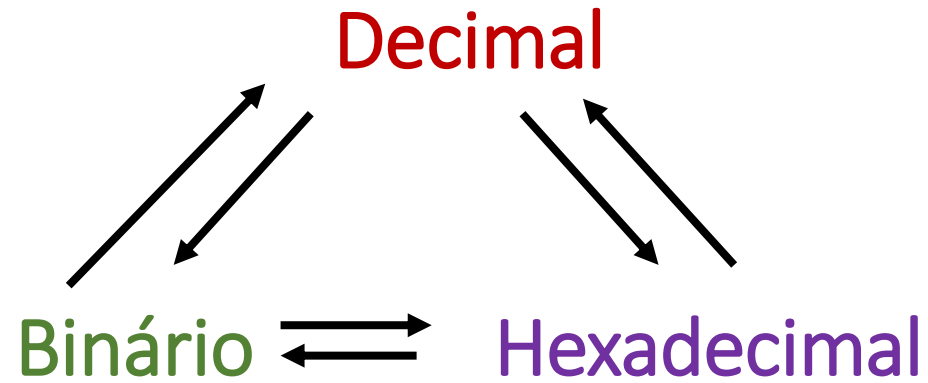
BITS E BYTES

0x80000000		0x100	
S:0x8000xxxx	Data (Binary: 2 bytes)		Charact
...0000	0b1101100111101111	0b1111101011111111
...0004	0b1101100111101111	0b1111101011111111
...0008	0b1101100111101111	0b1111101011111111
...000C	0b1101100111101111	0b1111101011111111
...0010	0b1111011101110111	0b0110111101111010	w.ZO
...0014	0b1111011101110111	0b0110111101111010	w.ZO
...0018	0b1111011101110111	0b0110111101111010	w.ZO
...001C	0b1111011101110111	0b0110111101111010	w.ZO
...0020	0b1010111101111111	0b1111001001111100	8 . .
...0024	0b1010111101111111	0b1111001001111100	8 . .
...0028	0b1010111101111111	0b1111001001111100	8 . .
...002C	0b1010111101111111	0b1111001001111100	8 . .
...0030	0b1100111111111111	0b1110111110101011
...0034	0b1100111111111111	0b1110111110101011
...0038	0b1100111111111111	0b1110111110101011

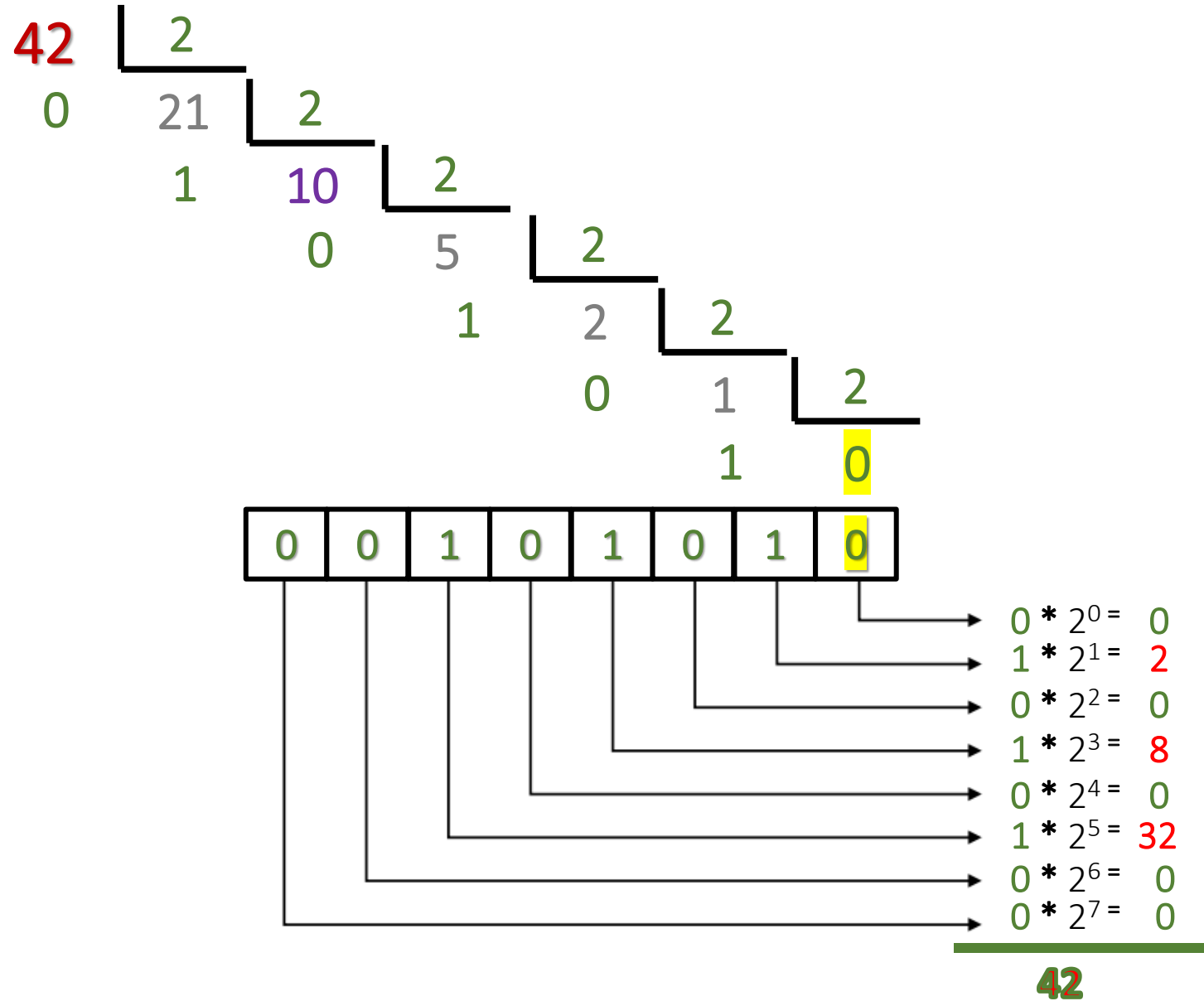
8 BITS = 1 BYTES

DECIMAL → HEXADECIMAL → BINÁRIO

BINÁRIO	HEXADECIMAL	DECIMAL
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15



Decimal \longleftrightarrow Binário



Decimal ↔ Hexadecimal

$$\begin{array}{r}
 495 \quad | \quad 16 \\
 \hline
 15 \quad 30 \quad | \quad 16 \\
 \hline
 14 \quad 1 \quad = 1EF
 \end{array}$$

DECIMAIS	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
HEXADECIMAIS	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

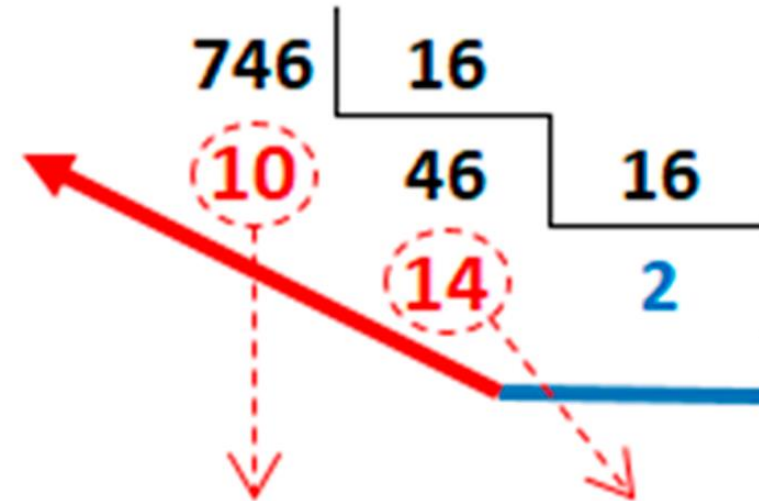
=1EF

$$\begin{array}{r}
 \begin{array}{l} \text{└─┐} \\ \text{└─┐} \\ \text{└─┐} \end{array}
 \begin{array}{l}
 15 * 16^0 = 15 \\
 14 * 16^1 = 224 \\
 01 * 16^2 = 256 \\
 \hline
 495
 \end{array}
 \end{array}$$

Decimal \rightleftharpoons HEXADECIMAL

Transcrevendo o hexadecimal:

2EA₁₆



Decimais	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimais	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

EXERCÍCIO DESAFIO

9) Criar a narrativa, o pseudocódigo e o fluxograma para um programa que solicite ao usuário um valor decimal de duas casas decimais e seja capaz de converter este valor para binário. E se puder ir além, tente também fazer um programa em linguagem python para este desafio.

Narrativa:

Fluxograma:

Pseudocódigo:

REFERÊNCIAS BIBLIOGRÁFICAS

MANZANO, José Augusto Navarro G.; OLIVEIRA, Jayr Figueiredo de. **Algoritmos - Lógica para Desenvolvimento de Programação de Computadores**. [Digite o Local da Editora]: Editora Saraiva, 2019. *E-book*. ISBN 9788536531472. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788536531472>. Acesso em: 10 mar. 2024.

GOOGLE COLABORATORY:

<https://colab.research.google.com/notebooks/intro.ipynb>