

## Aula 2

### Lógica de Programação e Algoritmos

Prof. Vinicius Pozzobon Borin

1

### Conversa Inicial

2

- O objetivo desta aula é dar os primeiros passos em programação
- Construção de nossos primeiros algoritmos

3

- Esta aula está estruturada com os seguintes conteúdos:
  - Onde programar em Python?
  - Quais partes compõem um algoritmo computacional?
  - Como mostramos uma mensagem na tela?
  - Como manipulamos dados e variáveis?
  - Como realizamos operações aritméticas?
  - Como lemos informações do teclado?

4

### Ambientes de desenvolvimento

5

- Google Colab – Jupyter Notebook
- <<https://colab.research.google.com/>>



6

- Agora vamos à ferramenta

7

- Podemos usar o Python offline, ou seja, instalado na minha máquina?

8

- Python requer a instalação em ambientes Windows
  - <<https://www.python.org/downloads/>>
- Nativamente instalado em ambientes Linux

9

- Junto ao Python vem instalado o IDLE



IDLE Python Shell 3.8 em ambiente Windows

10

- Ambiente de Desenvolvimento Integrado (*Integrated Development Environment – IDE*)

11

- PyCharm Community Edition
  - <<https://www.jetbrains.com/pt-br/pycharm/>>



12

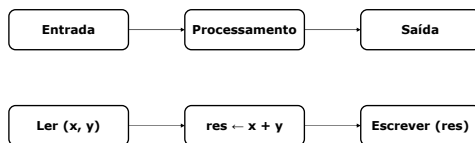
- Visual Studio Code – mantido pela Microsoft
- <<https://code.visualstudio.com/>>



13

## Ciclo de processamento de dados

14



Fonte: Borlin, 2020.

15

## O primeiro programa

- Vamos praticar no Python

16

## Atenção ao programar

- Sempre verifique cada caractere digitado
- Caracteres maiúsculos e minúsculos são distintos
- Veja o exemplo no Google Colab

17

- Abriu aspas? Feche
- Abriu parênteses? Feche
- Cuidado com os espaços

18

Comando de saída

print: comando, instrução, função

Escrever

Pseudocódigo

→

print

Python

Fonte: Borin, 2023.

19

Parênteses

Função

Mensagem

print

(

'

Olá,

mundo!

'

)

Aspas

Fonte: Borin, 2023.

20

Comando de saída

Vamos praticar no Python

21

Parênteses

Função

Cálculo aritmético

print

(

2+3

)

Fonte: Borin, 2023.

22

Vamos praticar no Python

23

Operadores e operações matemáticas

Python	Operação
+	Adição
-	Subtração
*	Multiplicação
/	Divisão (com casas decimais)
//	Divisão (somente a parte inteira)
%	Módulo/resto da divisão
**	Exponenciação ou potenciação

24

- **Atenção à ordem de precedência dos operadores**

$$10 \times \left( \frac{5 + 7}{4} \right)$$

25

- **Vamos praticar no Python**

26

## **Variáveis, dados e seus tipos**

27

## **Dados**

- “Sequência de símbolos quantificados ou quantificáveis” (Puga, 2009, p. 18)
- Valores fornecidos via entrada e manipulados ao longo do programa

28

## **Variável**

- Nome dado a uma região da memória do programa
- Sempre que você invocar o nome de uma variável no programa, seu bloco de memória será automaticamente carregado da RAM

29

## **Atribuição**

- Lê-se: a variável “nota” recebe o dado 8,5

nota = 8.5



*Símbolo de atribuição*

30

- Vamos praticar no Python

31

### Regras para nomes de variáveis

- Nunca inicie o nome de uma variável com um número



32

- Inicie o nome de uma variável com uma letra ou sublinha (underline)

`nota` ♥      `_nota` ♥

33

- Números, letras e sublinhas podem ser empregados à vontade no meio

`nota5` ♥      `_n_o_t_a_` ♥  
`!nota` Ⓢ

34

- Python permite o uso de letras com acentuação
- Não é recomendado

`preço` ♥ Python 3      `preço` Ⓢ Python 2

35

### PEP 8 – Python Enhancement Proposals

- Conjunto oficial de regras e boas práticas do Python:
- <https://peps.python.org/pep-0008/>

`precoTotal` Ⓢ      `preco_total` ♥

36

### Tipos primitivos de dados

- ▀ Numérico (inteiro e ponto flutuante)
- ▀ Caractere
- ▀ Literal/booleana

37

### Variáveis numéricas

- ▀ Quando queremos realizar operações aritméticas
- ▀ Inteiro (int) – números sem casas decimais
- ▀ Ponto flutuante (*float*) – números com casas decimais

38

### Variáveis lógicas/booleanas

- ▀ Variável para um *bit*. Dois estados:
  - True (Verdadeiro) e False (falso)
  - Nível lógico alto ou baixo
  - 1 ou 0

39

- ▀ Lista de operadores lógicos em Python e em pseudocódigo

Python	Operação
==	Igualdade
>	Maior que
<	Menor que
>=	Maior ou igual a
<=	Menor ou igual a
!=	Diferente

40

- ▀ Vamos praticar no Python

41

### Variáveis de cadeias de caracteres (*strings*)

- ▀ Armazenam conjuntos de símbolos encadeados, incluindo acentuação, pontuação etc.

42

7 bits

Tabela ASCII

Dec	Hx	Oct	Chr	Dec	Hx	Oct	Chr
32	20	040	Space	64	40	100	@
33	21	041	!	65	41	101	A
34	22	042	"	66	42	102	B
35	23	043	#	67	43	103	C
36	24	044	\$	68	44	104	D
37	25	045	%	69	45	105	E
38	26	046	&	70	46	106	F
39	27	047	'	71	47	107	G
40	28	050	(	72	48	110	H
41	29	051	)	73	49	111	I
42	2A	052	*	74	4A	112	J
43	2B	053	+	75	4B	113	K
44	2C	054	,	76	4C	114	L

Fonte: ASCII Table, [8,4].

43

Unicode

UTF-8

UTF-16

UTF-32

Fonte: Unicode, [8,4].

44

Variáveis de cadeias de caracteres (*strings*)

String

O

I

a

,

m

u

n

d

o

!

String (representada em ASCII)

79

108

97

44

32

109

117

110

100

111

33

Fonte: Borlin, 2020.

45

Índice do caractere – número inteiro que indica onde o caractere está dentro da *string*

A contagem do índice inicia sempre em zero

46

Variáveis de cadeias de caracteres (*strings*)

String

0

1

2

3

4

5

6

7

8

9

10

→ Índice

→ Conteúdo

O

I

a

,

m

u

n

d

o

!

Fonte: Borlin, 2020.

47

Vamos praticar no Python

48

8



## Manipulações com *strings*

## Concatenação

- ▀ Juntar/somar *strings*
- ▀ Vamos praticar no Python

49

50

## Composição com marcadores de posição

- ▀ Juntar diferentes variáveis e *strings*

Marcador	Tipo
%d ou %i	Números inteiros
%f	Números de ponto flutuante
%s	<i>Strings</i>

- ▀ Vejamos novamente no Python

51

52

## Composição moderna

'Você tirou %d na disciplina de %s'  
% (nota, disciplina)

'Você tirou {} na disciplina de {}'  
.format(nota, disciplina)

- ▀ Vejamos novamente no Python

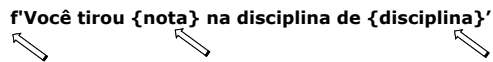
53

54

### Composição com f-string

'Você tirou {} na disciplina de {}'  
.format(nota, disciplina)

f'Você tirou {nota} na disciplina de {disciplina}'



55

- ▀ Vejamos novamente no Python

56

### Fatiamento

- ▀ Podemos recortar/fatiar um pedaço da *string*
- ▀ Vamos praticar no Python

57

### Tamanho (*length*)

- ▀ Podemos descobrir o tamanho da cadeia de caracteres com uma função chamada *len*
- ▀ Vamos praticar no Python

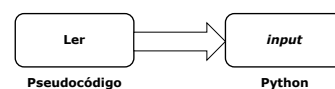
58

### Função de entrada e fluxo de execução do programa

59

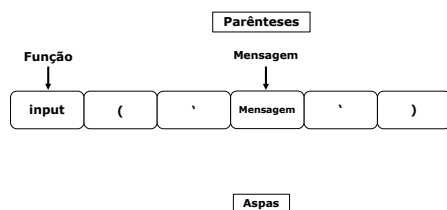
### Comando de entrada

- ▀ *input*: comando, instrução, função



Fonte: Berlin, 2020.

60



Fonte: Borlin, 2020.

61

➤ **Vamos ao Python**

62

### Convertendo dados de entrada (casting)

- O `input` sempre retorna um dado do tipo *string*
- Se quisermos um dado numérico, utilizamos a função `int` ou `float` antes do `input`

63

➤ **Vamos ao Python**

64

### Casting de variáveis

- Ocorre quando existe a conversão de uma variável de um tipo de dado para outro. Em muitas linguagens de programação, é possível converter uma variável de um tipo para outro, desde que essa conversão seja lógica e faça sentido no contexto do programa

65

### Fluxo de execução do programa e o Teste de Mesa

- Como se dá a execução de um programa em Python?
- Vejamos no código

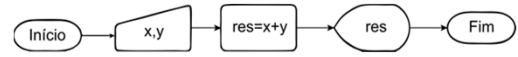
66

### Exercício

- Desenvolva um algoritmo que solicite ao usuário dois números inteiros. Imprima a soma desses dois números na tela

67

### Fluxograma



68