Linguagem Python

Aula 2

Características

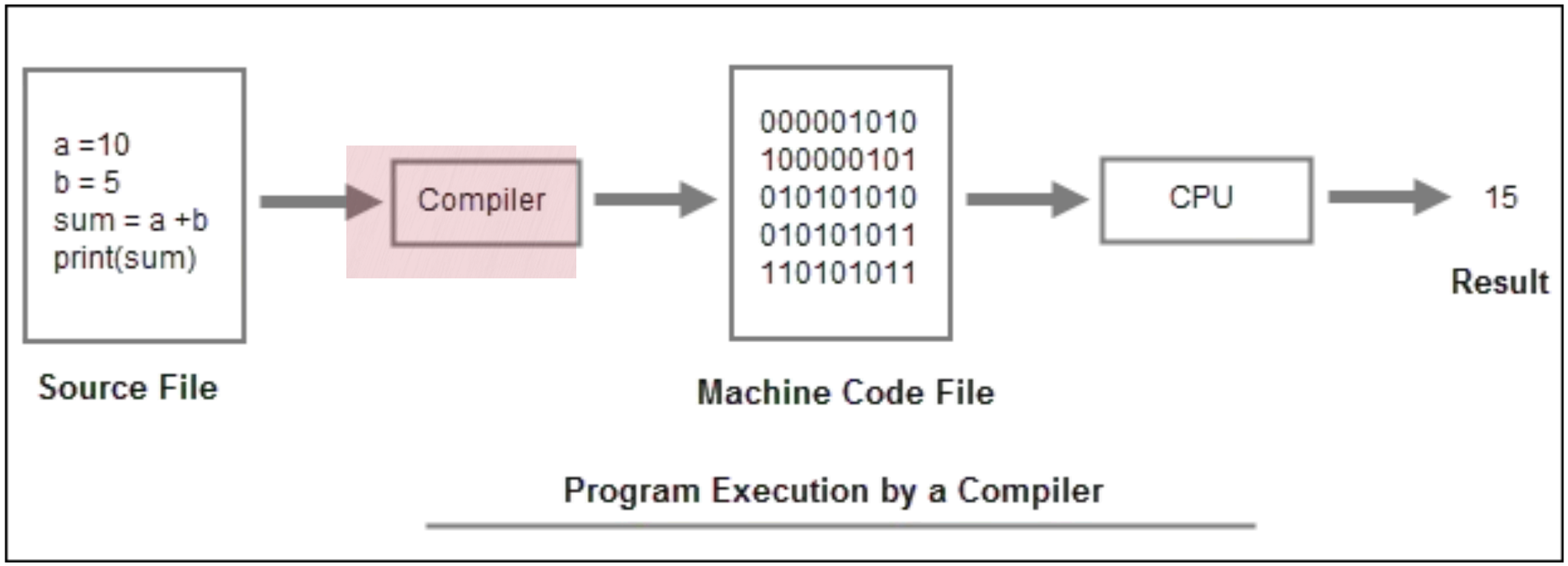
- **Python** é uma linguagem de alto nível, de uso geral, criada por Guido van Rossum, lançada em 1991.
  Versões: Python 1.0 (1994), Python 2.0 (2000), Python 3.0 (2008)

- Conhecido por sua simplicidade, legibilidade e portabilidade.

- É uma linguagem interpretada ("e compilada")

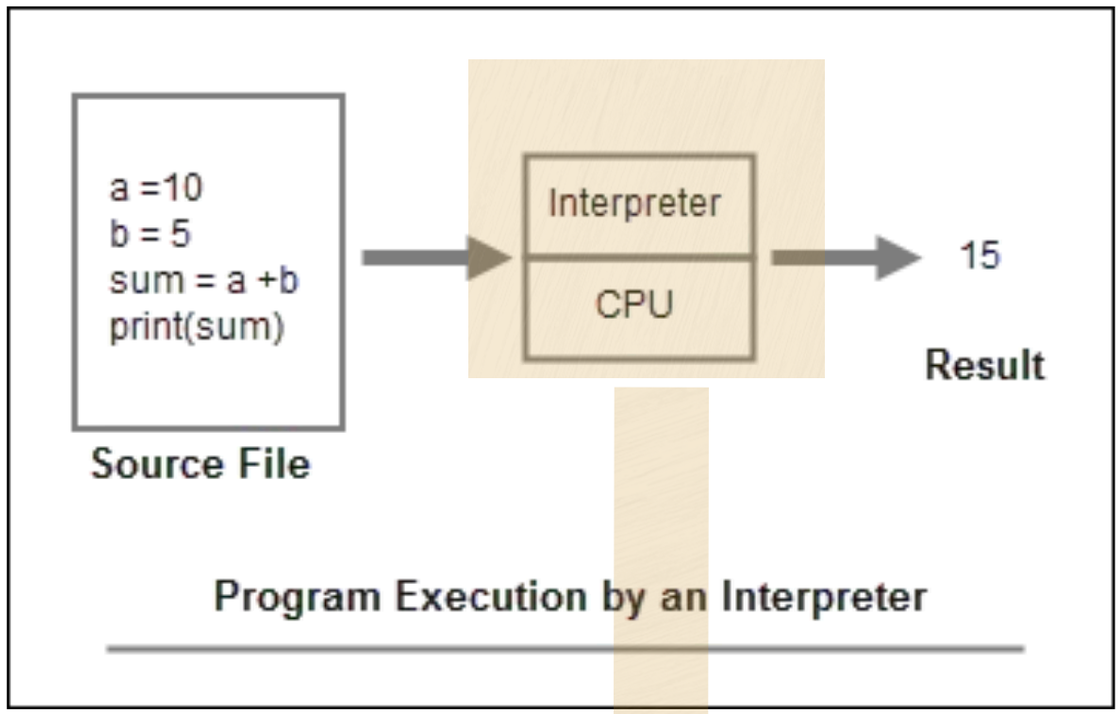
- Pode ser usada para aplicações web, científicas, jogos, sistemas administrativos, etc. [Usada por Google, Dropbox, Youtube, Mozilla, NASA,...]

Linguagens compiladas

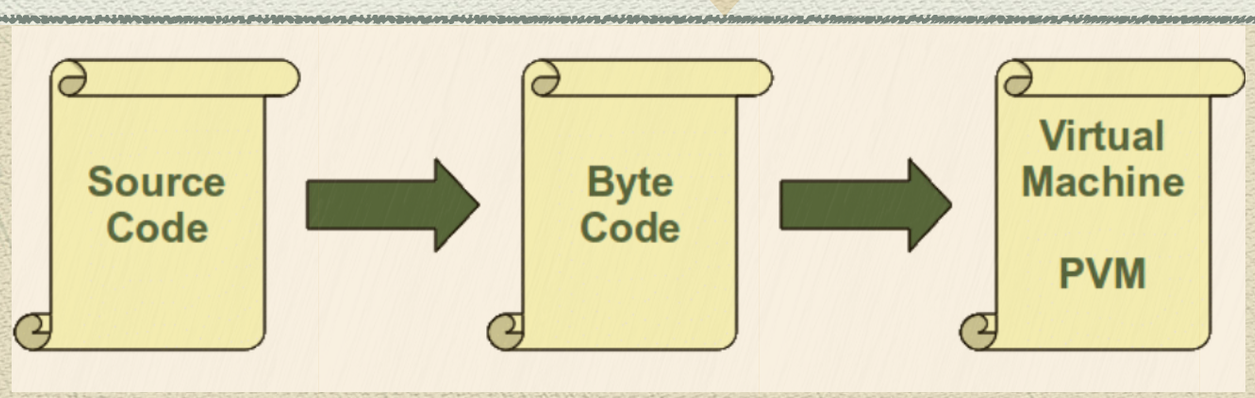


Compilador: programa que traduz um programa numa linguagem de alto nível (código fonte) num código em linguagem de máquina (código binário)

Python - linguagem interpretada



Fluxo quando o código fonte (source code) está em **Python**



Ex. de um programa em Python

```
# arquivo: soma2numeros.py
#  ------------------------
#  Este programa pede dois numeros inteiros (que o usuário deve fornecer),
#  imprime esses numeros, faz a soma e imprime o resultado
#
#  OBS: Veja o alinhamento (indentação) dos comandos
#       Experimente mudar alguns alinhamentos e veja o que acontece
# ..............................

def main():
    A = int(input("Digite o primeiro número:"))
    B = int(input("Digite o segundo número:"))

    print("Primeiro número = ", A)
    print("Segundo número = ", B)

    soma = A + B   # resultado da soma de A e B é guardado na variavel soma
    print("Soma = ", soma)
# ..............................
main()
```

Conceitos básicos

Variáveis

Uma **variável** é um nome que se refere a um objeto (valor). É a maneira de se referir a uma posição de memória usada num programa. É um nome simbólico para um local físico (uma "casa de pombo"). Este lugar físico poderá conter valores como números, textos, etc.

No decorrer da execução do programa, novos valores (de diferentes tipos) podem ser atribuídos às variáveis.

- Em Python não se declara variáveis (ou os seus tipos). Quando precisar, basta pensar num nome (válido) e começar a usá-lo.
- Escolha de nomes válidos para as variáveis (veja a seguir)

Variáveis

Como escolher nomes para as variáveis:

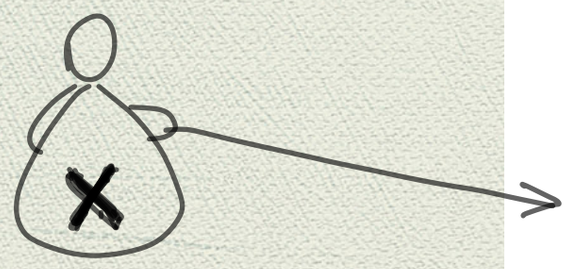
1. Podem conter letras minúsculas (a… z), letras maiúsculas (A…Z), dígitos (0…9), ou o caractere "_" (underscore); **mas não podem começar com um dígito**.
2. Podem ter qualquer comprimento;
3. Não podem ser uma das palavras-chaves (keywords) do Python. (Logo, você saberá quais são.)

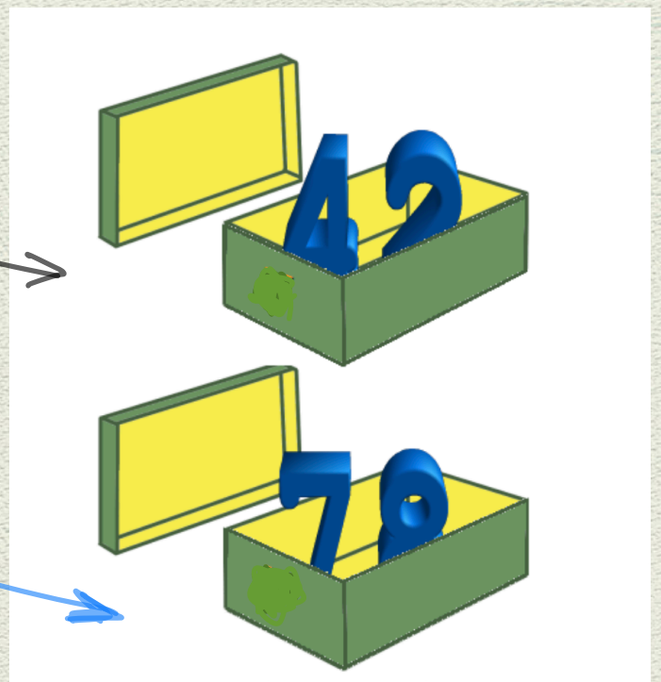
OBS: Letras maíusculas e minúsculas são consideradas distintas! Assim, os nomes max, Max, MAX, MaX são todos distintos. Cuidado para não errar!

- Exemplos corretos: dia, Dia, mes, Mes, Resposta5, Bola7, H2SO4, EP2_mac110, x3, x_3, ano_bissexto, temperatura_media, Preco_Max, Nome, total_sum , numero_aleatorio, conta_pares, lista_nomes, lista_precos, Lista_notas, Pi, delta, epsilon, maximo, peso_min, Ehprimo, pe_de_moleque, _Min_Max,
- Exemplos incorretos: 7Bola, @rroba, bravo! , Argh&%#, lista-de-precos, True, False

Comando de atribuição

$X = 42$





$Y = 78$

$Y = X + Y - 3$

<variavel> = <expressao aritmetica>

Significado:

<variavel> aponta para o valor da <expr. aritmetica>

$Y = 117$

Operadores aritméticos

A tabela a seguir mostra a associatividade das principais operações aritméticas em Python, em ordem decrescente de precedência (da maior para a menor):

Tabela de precedência e associatividade de operadores aritméticos

Operador	descrição	Associatividade
()	parênteses	da esquerda para a direita
**	potência	da direita para a esquerda
+, -	positivo e negativo unário	da direita para a esquerda
*, /, //, %	multiplicação , divisão, divisão inteira e resto	da esquerda para a direita
+, -	soma e subtração	da esquerda para a direita

Tabela dos operadores aritméticos em ordem decrescente de precedência

Expressões típicas com inteiros

expression	*value*	*comment*
99	99	*integer literal*
+99	99	*positive sign*
-99	-99	*negative sign*
5 + 3	8	*addition*
5 - 3	2	*subtraction*
5 * 3	15	*multiplication*
5 // 3	1	*no fractional part*
5 % 3	2	*remainder*
5 ** 3	125	*exponentiation*
5 // 0	*run-time error*	*division by zero*
3 * 5 - 2	13	* * has precedence*
3 + 5 // 2	5	*// has precedence*
3 - 5 - 2	-4	*left associative*
(3 - 5) - 2	-4	*better style*
3 - (5 - 2)	0	*unambiguous*
2 ** 2 ** 3	256	*right associative*
2 ** 1000	107150...376	*arbitrarily large*

Operadores e expressões relacionais

Além de “fazer contas”, o Python permite comparar valores usando os seguintes **operadores relacionais**:

Tabela dos operadores relacionais

Operador	Descrição	Exemplo	Resultado
==	igualdade	2 == 3	False
!=	desigualdade	2 != 3	True
>	maior	3 > 3	False
>=	maior ou igual	3 >= 3	True
<	menor	2 < 3	True
<=	menor ou igual	4 <= 3	False

Operadores relacionais comparam dois valores e o resultado pode ser `False` (falso) ou `True` (verdadeiro). Esse dois valores são chamados de valores **booleanos** em homenagem ao matemático George Boole (https://pt.wikipedia.org/wiki/George_Boole).

Assim como dizemos que as expressões aritméticas são reduzidas a um valor numérico inteiro ou real, as expressões relacionais são reduzidas a um valor booleano (ou seja, `True` ou `False`). As expressões relacionais podem conter expressões aritméticas, como no seguinte exemplo em Python shell:

```
>>> 2 + 3 == 3 + 2 * 1
True
```

Notas de aulas para a disciplina

Introdução à Computação - MAC110/MAC115 - IME-USP

Profas. Nami Kobayashi e Yoshiko Wakabayashi