

Organização de programas em Python



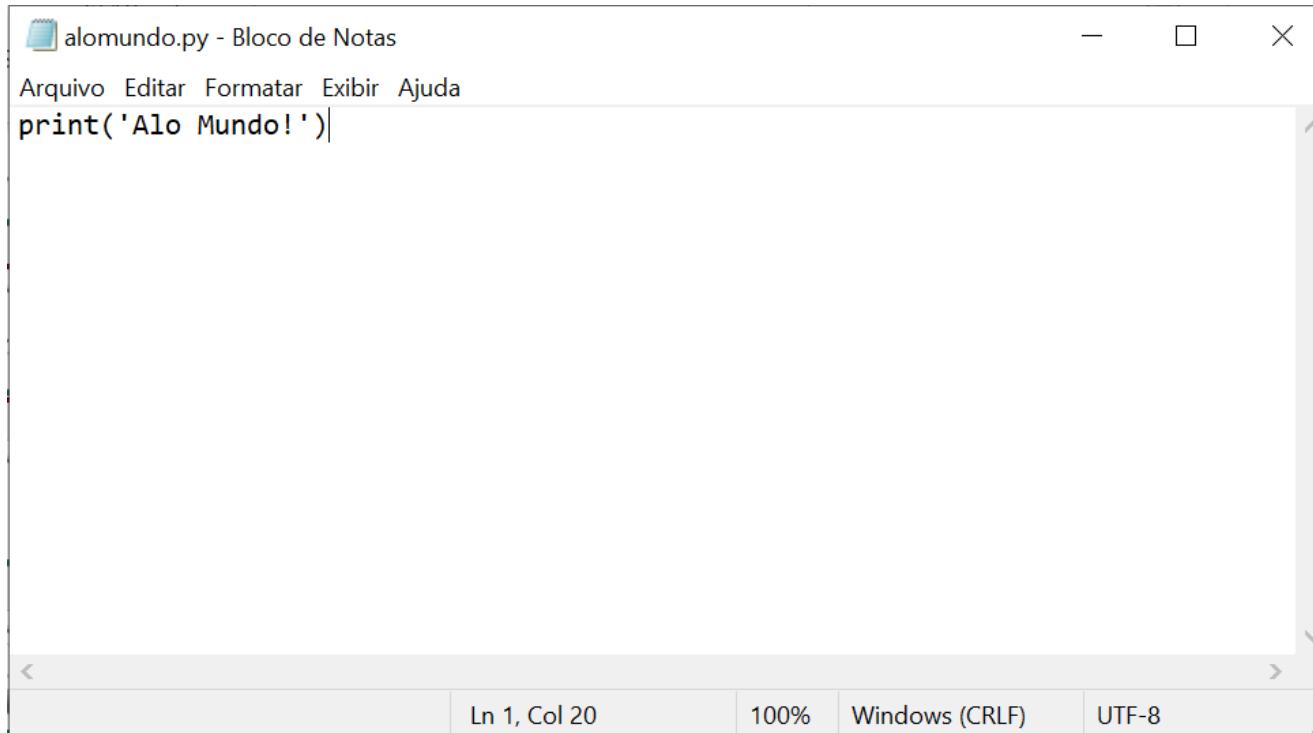
Vamos programar em Python!

Mas...

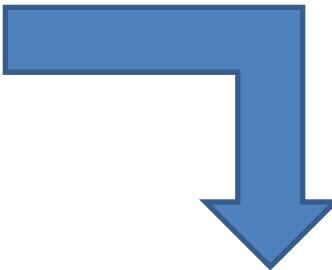
- Como um programa é organizado?
- Quais são os tipos de dados disponíveis?
- Como variáveis podem ser declaradas?
- Como atribuir valores às variáveis?
- Como entrada e saída básica de dados podem ser feitas?

Vamos começar com um exemplo...

Primeiro passo: escrever o programa!



The screenshot shows a Windows Notepad window titled "alomundo.py - Bloco de Notas". The menu bar includes "Arquivo", "Editar", "Formatar", "Exibir", and "Ajuda". The main text area contains the Python code: `print('Alo Mundo!')`. The status bar at the bottom shows "Ln 1, Col 20", "100%", "Windows (CRLF)", and "UTF-8".



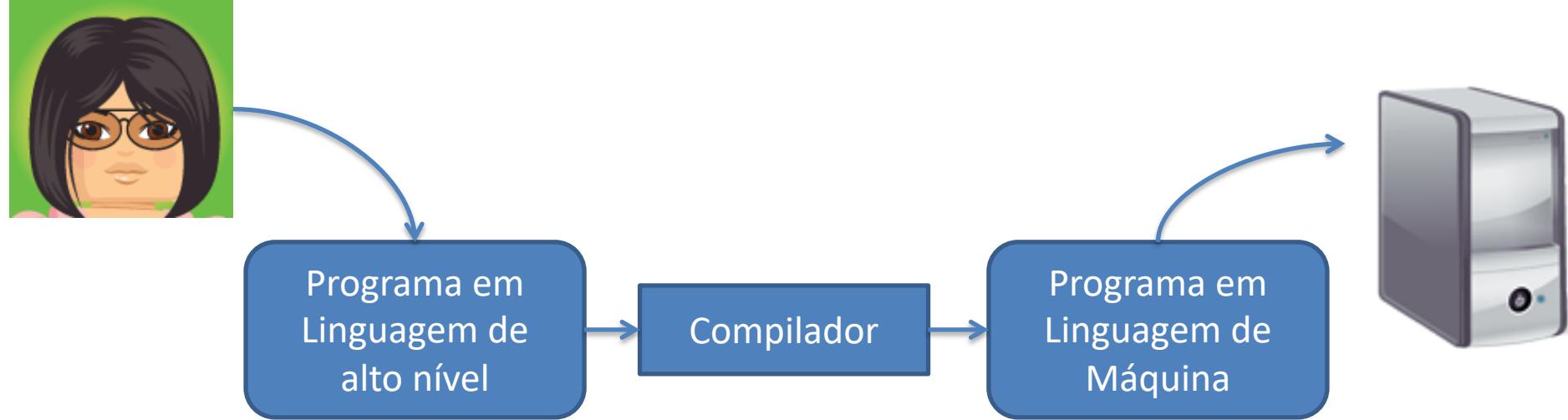
alomundo.py

Mas o computador não conhece Python!!!

- O computador só entende binário
 - Linguagem de zeros e uns
 - 0100100111010101001010101, entendeu?
- Precisamos traduzir o programa Python para binário

Compilação

- Na maioria das linguagens, antes de executar um programa, é necessário compilar o programa
- O compilador gera um arquivo “executável”
 - Esse novo arquivo é o que será de fato executado



Python é uma linguagem interpretada

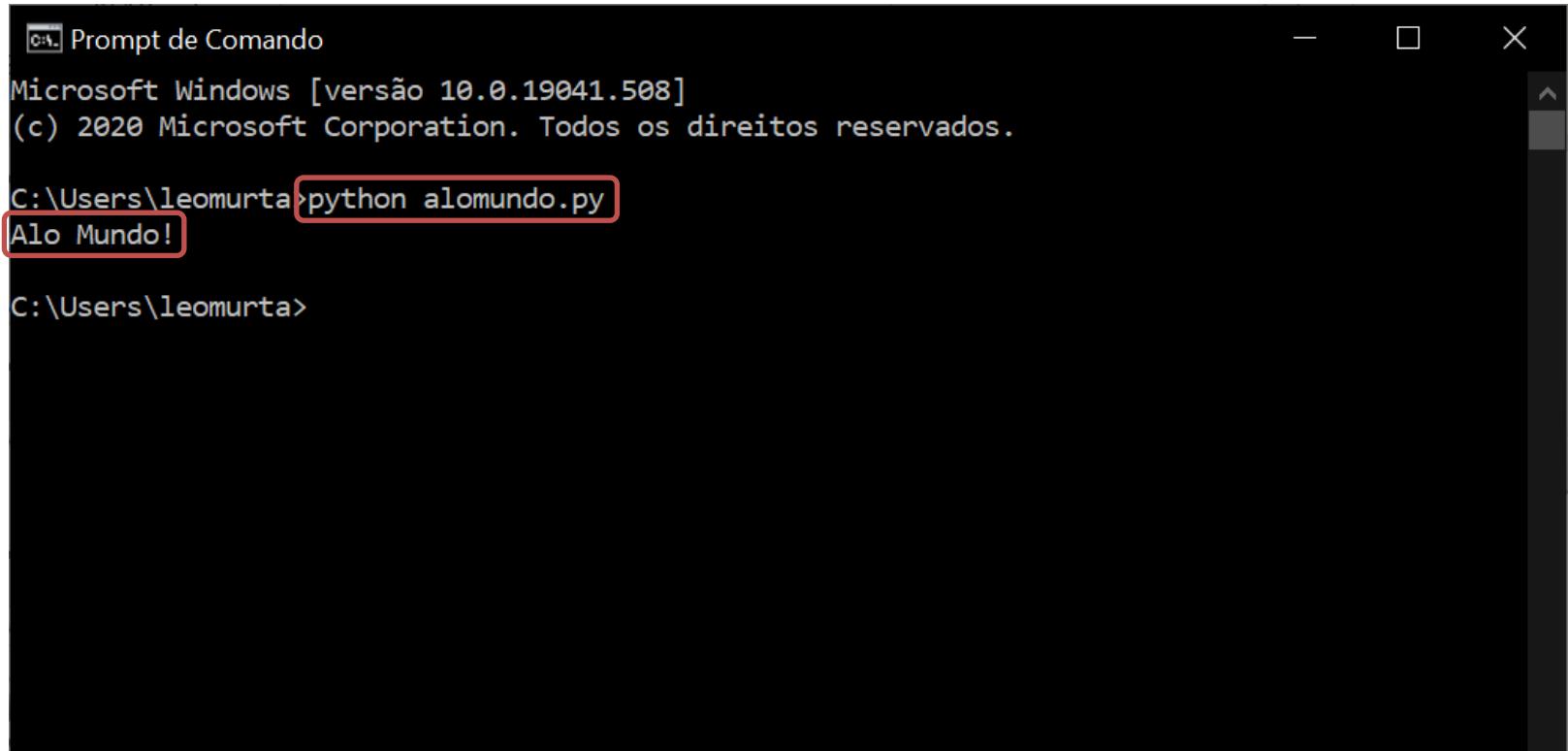
- Não é necessário compilar o código Python
- O interpretador Python vai lendo o código fonte, traduzindo para linguagem de máquina e executando ao mesmo tempo

Instalação do Interpretador Python

- Download do Python mais recente
 - <http://www.python.org/downloads/>
 - Usar as configurações padrões



Execução



```
C:\ Prompt de Comando
Microsoft Windows [versão 10.0.19041.508]
(c) 2020 Microsoft Corporation. Todos os direitos reservados.

C:\Users\leomurta>python alomundo.py
Alo Mundo!

C:\Users\leomurta>
```

VAMOS FAZER JUNTOS?

Notepad x IDE

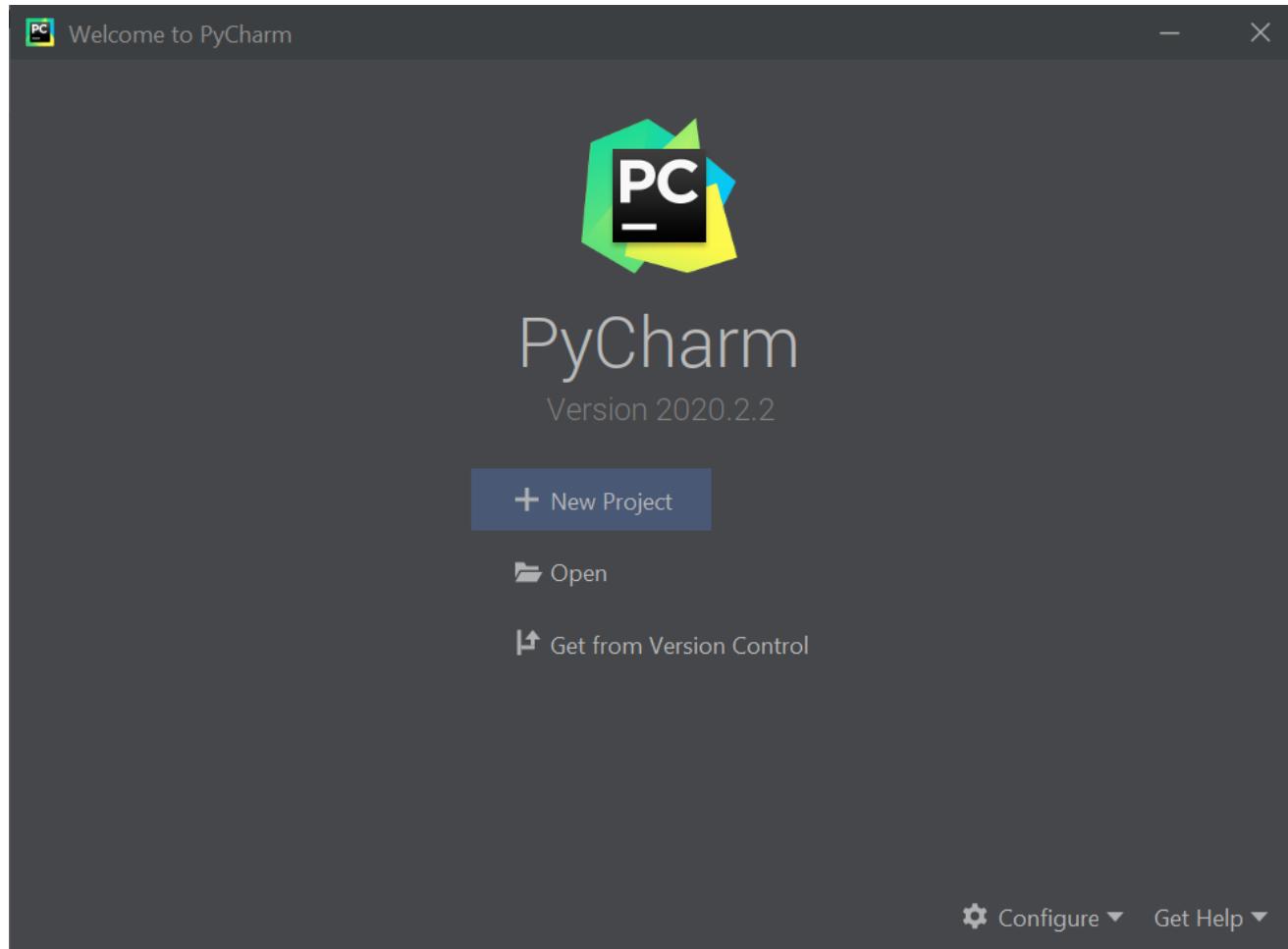
- Dificuldades do Notepad
 - Editor básico, sem ajuda para programar
 - Execução externa
- *Integrated Development Environment (IDE)*
 - Colore o código
 - Autocompleta o código
 - Verifica a sintaxe ao digitar
 - Permite executar o código de forma integrada
 - Etc.

Instalação do PyCharm

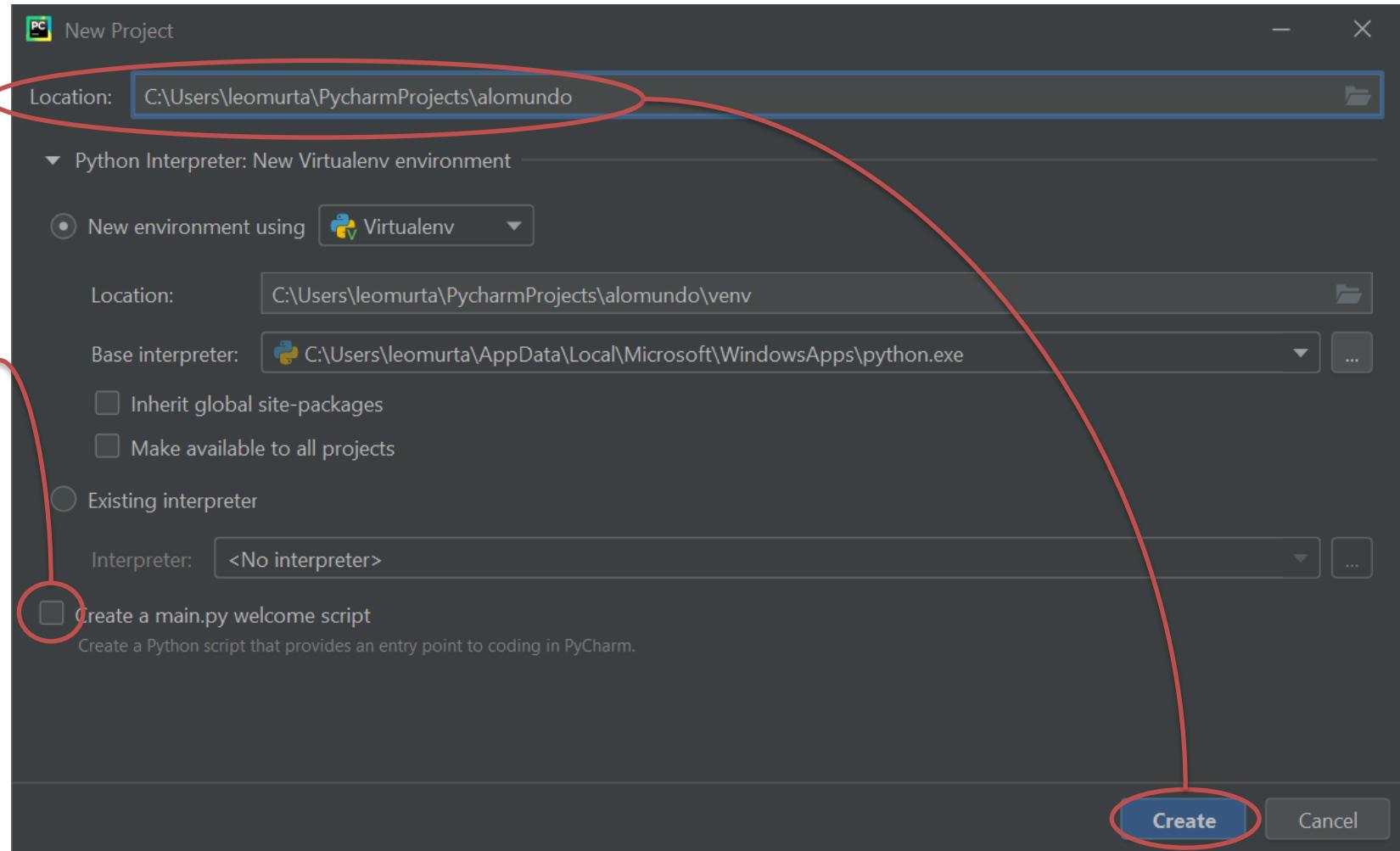
- Usaremos o PyCharm nas aulas, mas os alunos podem optar por qualquer outra IDE ou editor
- Download do PyCharm
 - <https://www.jetbrains.com/pycharm/download/>
 - Versão *Community* (gratuita)
 - Usar as configurações padrões



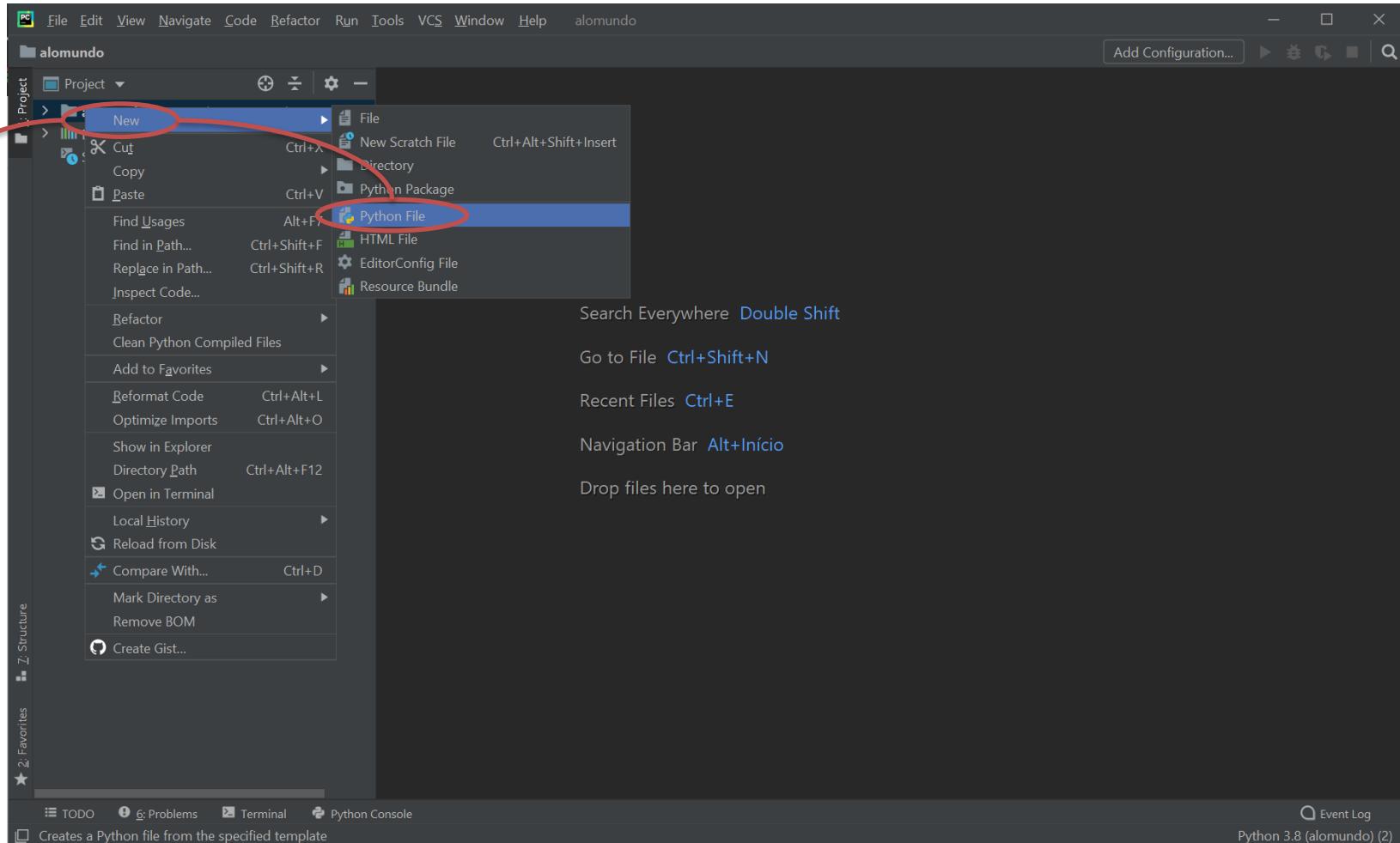
Criando um projeto no PyCharm...



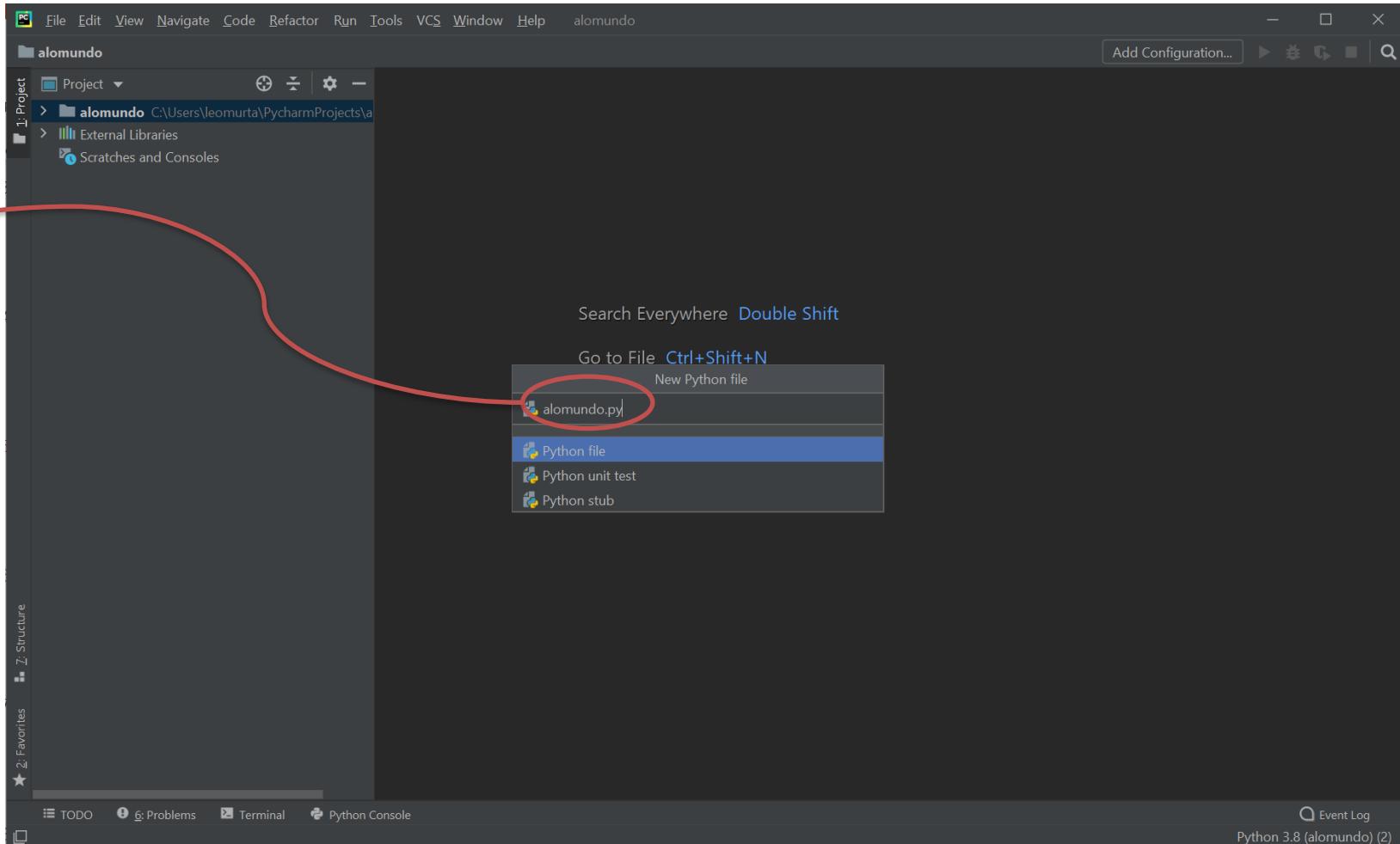
Criando um projeto no PyCharm...



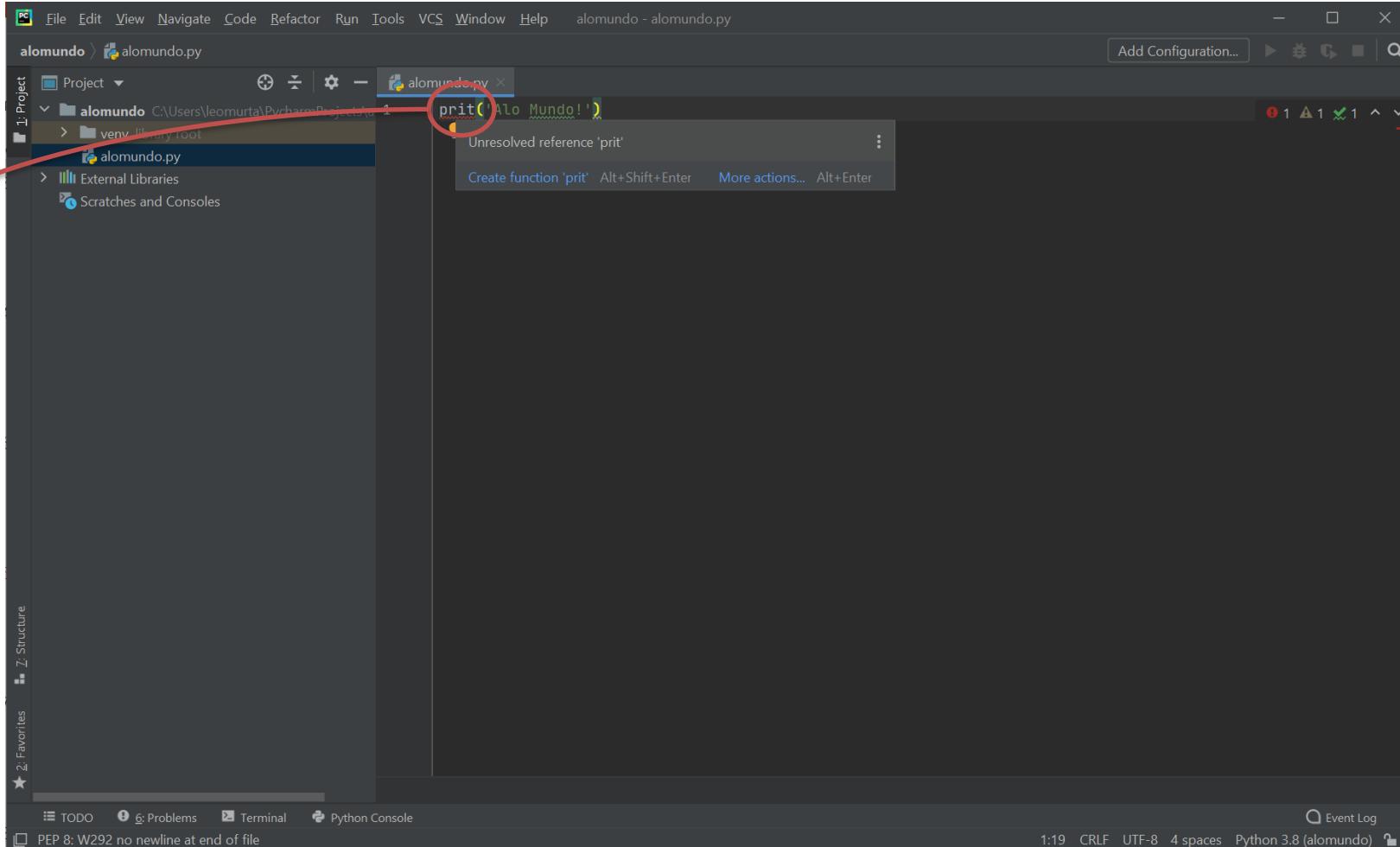
Criando um Arquivo Python no Projeto



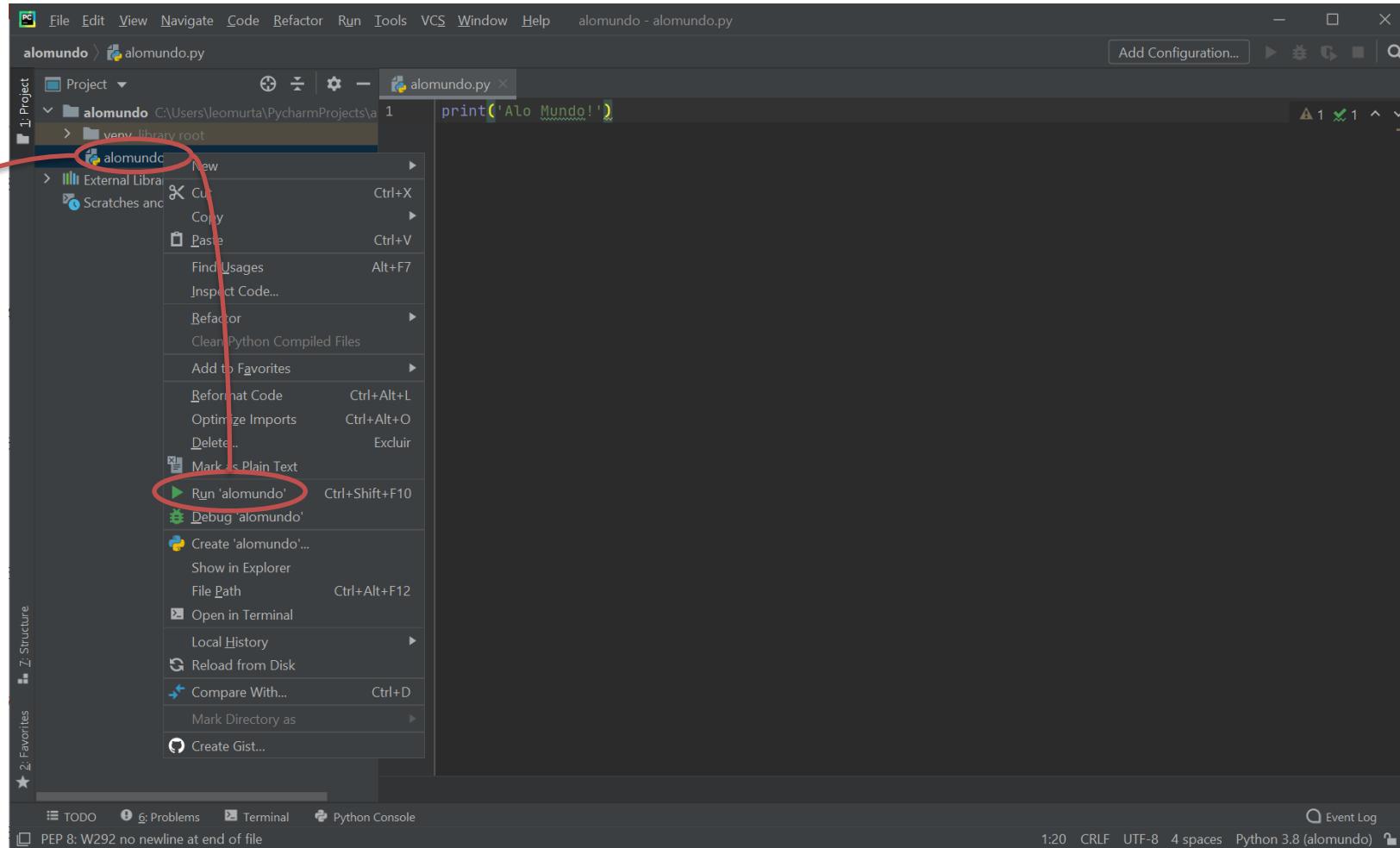
Criando um Arquivo Python no Projeto



Escrevendo o programa no PyCharm...



Executando o programa no PyCharm...



Clicar com o
botão da
direita sobre
o arquivo do
programa

Selecionar
Run
'alomundo'

Escrevendo e executando o programa no PyCharm...

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and the current file alomundo - alomundo.py. The left sidebar displays the Project structure with a folder named 'alomundo' containing 'venv' and 'alomundo.py'. The main editor window shows the Python code:

```
print('Alo Mundo!')
```

The bottom terminal window shows the execution of the script and its output:

```
C:\Users\leomurta\PycharmProjects\alomundo\venv\Scripts\python.exe C:/Users/leomurta/PycharmProjects/alomundo/alomundo.py
Alo Mundo!
```

A red box highlights the output 'Alo Mundo!' in the terminal. The status bar at the bottom right indicates the time as 1:20, encoding as CRLF, character set as UTF-8, 4 spaces, Python 3.8 (alomundo), and an event log count of 1.

VAMOS FAZER JUNTOS?

Regras básicas

- A sequência dos comandos é importante
 - O Python lê de cima para baixo, como nós
- Em casos de códigos alternativos ou que se repetem, podemos criar blocos de comandos
 - Blocos devem ser criados usando indentação (com espaços ou tab)

Comentários

- Comentários são trechos do programa voltados para a leitura por humanos, e ignorados pelo interpretador
- Começam com o símbolo **#**
 - Tudo na linha após **#** é ignorado pelo interpretador
- Use comentários para documentar seu código e fazer com que ele seja fácil de entender por outras pessoas

Atribuição de valores

- Em Python, o operador de igualdade (`=`) é usado para atribuir valores às variáveis
- É equivalente ao símbolo de atribuição (`←`) que usávamos no pseudocódigo
- Sempre na forma: **variável = valor ou expressão**
 - A expressão do lado direito é processada
 - O valor gerado é atribuído à variável

Exemplo de programa em Python

```
# Este programa calcula a área de um triangulo retângulo
altura = 15
base = 3
area = (altura * base) / 2
print(area)
```

Quais são os tipos de dados disponíveis?

- Em Python, toda variável tem um tipo
- Com isso, o computador pode saber quais operações são permitidas
- Os tipos podem ser divididos em três grupos
 - Tipos numéricos (inteiro, real, etc.)
 - Tipos textuais (caractere e string)
 - Tipo lógico (booleano)
- Os tipos são definidos dinamicamente, pelo próprio Python
 - Não é preciso dizer de que tipo é cada variável

Exemplo de variáveis lógicas (boolean)

x = True

y = False

Exemplo de variáveis textuais (string)

```
nome = 'Maria'  
sobrenome = "Silva"  
letra = 'A'  
texto = 'Alo Mundo'
```

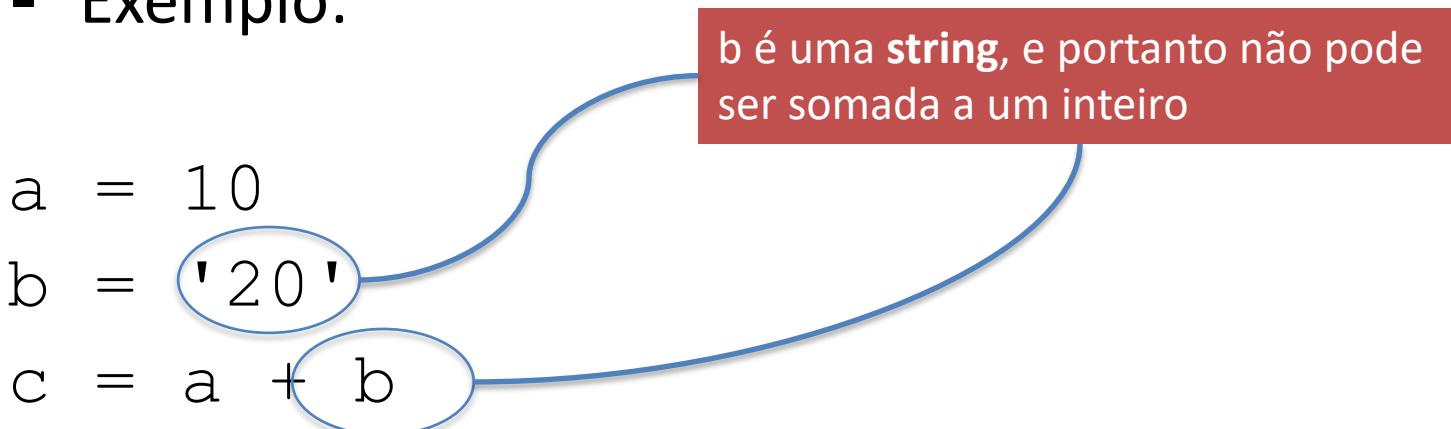
Tipagem Dinâmica

a = -5	→ inteiro
b = 10	→ inteiro
c = 200	→ inteiro
d = -12312312	→ inteiro
e = 345092834	→ inteiro
f = 2.5	→ float
g = 0.6023e24	→ float
h = 0.4e-3	→ float

- Tipo é determinado automaticamente pelo Python no momento da criação da variável

Tipagem Forte

- Uma vez que uma variável tenha um valor de um tipo, ele não pode ser usado como se fosse de outro tipo
- Exemplo:



The diagram shows a Python code snippet with three assignments: `a = 10`, `b = '20'`, and `c = a + b`. The variable `b` is highlighted with a blue oval, and the entire assignment `c = a + b` is also highlighted with a blue oval. A red callout box points to the `b` assignment with the text: "b é uma **string**, e portanto não pode ser somada a um inteiro".

```
a = 10
b = '20'
c = a + b
```

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

TypeError: unsupported operand type(s) for +: 'int' and 'str'

Regras para nomes de variáveis

- Os nomes de variáveis devem respeitar algumas regras
 - São sensíveis a caixa
 - Podem ter tamanho ilimitado (mas evite abusos)
 - Devem começar com letra ou underline (_)
 - Outros caracteres podem ser letras, números ou underline
 - Não podem ter espaço nem acentos
 - Não podem ser uma palavra reservada da linguagem

Entrada de dados

- Para entrada de dados, usamos **input**
- É possível informar um texto que aparecerá impresso na tela para que o usuário saiba que o programa está esperando a entrada de um valor

```
nome = input('Digite o nome do aluno: ')
print(nome)
```

Input lê dados como string

- Você pode usar o comando `type` para saber o tipo que o Python atribuiu a uma variável

```
altura = input('Digite a altura do triangulo: ')
print(type(altura))
```

...

Mudança de tipo

- Usar `int()` ou `float()` para converter o tipo para numérico

```
altura = int(input('Digite a altura do triangulo: '))
print(type(altura))
```

...

Saída de dados

- Para saída de dados, usamos **print**

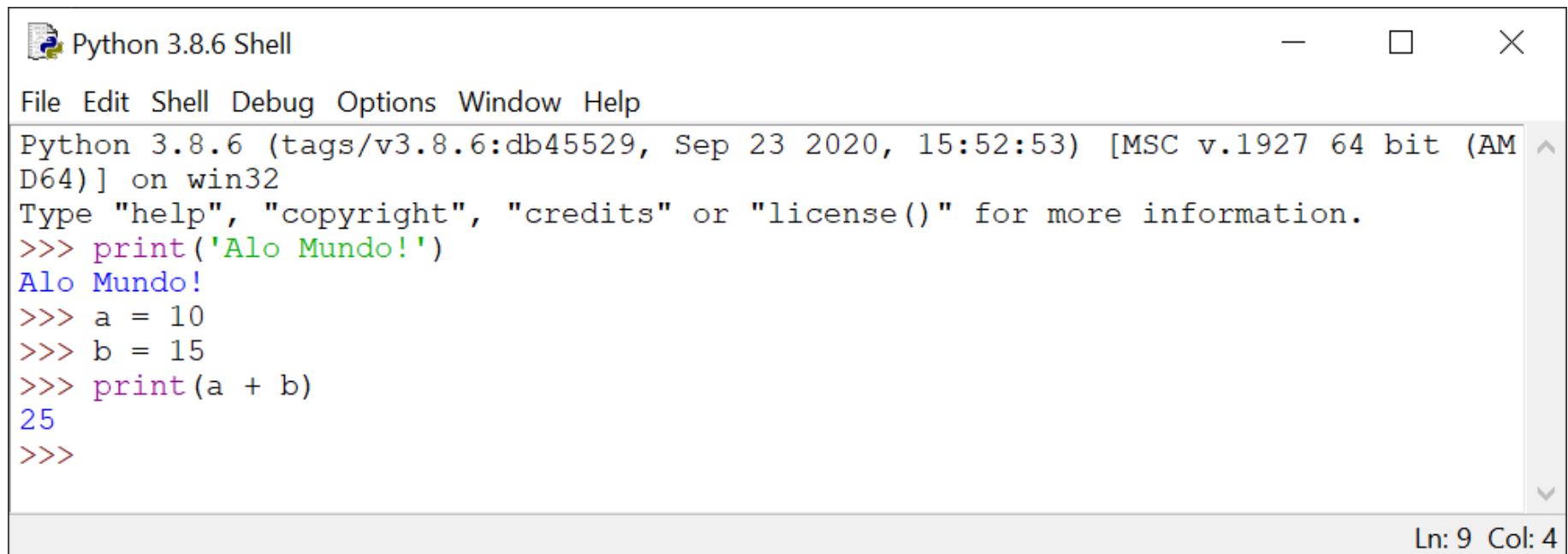
```
print('Prog é muito legal')
print(123)
altura = 10
print(altura)
print('Vamos pular uma linha \n')
print('O nome do aluno é', nome)
```

Voltando ao exemplo de programa em Python

```
altura = int(input('Digite a altura do triangulo: '))
base = int(input('Digite a base do triangulo: '))
area = (base * altura) / 2
print('A área do triangulo é', area)
```

IDLE

- Python também fornece uma interface interativa para execução de pequenas sequencias de comandos
- Basta rodar IDLE ou chamar *python* no prompt



The screenshot shows the Python 3.8.6 Shell window. The title bar reads "Python 3.8.6 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the following text:

```
Python 3.8.6 (tags/v3.8.6:db45529, Sep 23 2020, 15:52:53) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('Alo Mundo!')
Alo Mundo!
>>> a = 10
>>> b = 15
>>> print(a + b)
25
>>>
```

The status bar at the bottom right indicates "Ln: 9 Col: 4".

Exercícios

- Qual a saída do programa abaixo?

```
x = 1.0
y = 2.0
z = 3.0

x = -x
y = y - 1
z = z + x
z = z + x - y
print('x =', x, ', y =', y, ', z =', z)
```

Exercícios

1. Faça um programa que leia o nome, a idade, a altura, o peso e a nacionalidade do usuário e escreva essas informações na forma de um parágrafo de apresentação
2. Faça um programa que exiba o perímetro de uma circunferência a partir do seu raio
3. Faça um programa que leia dois pontos num espaço bidimensional e calcule a distância entre esses pontos

Exercícios

4. Faça um programa que informe a distância em quilômetros de um raio para o observador
 - O observador deve informar o tempo (em segundos) transcorrido entre ver o raio e ouvir o trovão
 - Assuma que a velocidade do som é 340 m/s

Exercícios

5. Faça um programa para, a partir de um valor informado em centavos, indicar a menor quantidade de moedas que representa esse valor
 - Considere moedas de 1, 5, 10, 25 e 50 centavos, e 1 real
 - Exemplo: para o valor 290 centavos, a menor quantidade de moedas é 2 moedas de 1 real, 1 moeda de 50 centavos, 1 moeda de 25 centavos, 1 moeda de 10 centavos e 1 moeda de 5 centavos

Referências

- Slides preparados em conjunto com Vanessa Braganholo e Aline Paes

Organização de programas em Python

