



Ativando venv (pycharm)

Terminal

Local



```
(.venv) PS C:\Users\jonatas.lopes\PycharmProjects\projeto_pyhton> deactivate  
PS C:\Users\jonatas.lopes\PycharmProjects\projeto_pyhton>  
PS C:\Users\jonatas.lopes\PycharmProjects\projeto_pyhton> .\.venv\Scripts\Activate  
(.venv) PS C:\Users\jonatas.lopes\PycharmProjects\projeto_pyhton>
```

MUITO IMPORTANTE = estar na pasta do projeto, onde já tem a pasta .venv



pasta do projeto



.venv



@fpftech.educacional

Relembrando tipagem

python

Copiar

```
# Inteiro
numero = 10
print(numero, type(numero)) # Output: 10 <class 'int'>

# Float
pi = 3.14159
print(pi, type(pi)) # Output: 3.14159 <class 'float'>

# String
nome = "Python"
print(nome, type(nome)) # Output: Python <class 'str'>

# Booleano
verdadeiro = True
print(verdadeiro, type(verdadeiro)) # Output: True <class 'bool'>
```

(opcional) python possui type hint

python

Copiar

```
# Inteiro com type hint
numero: int = 10
print(numero, type(numero)) # Output: 10 <class 'int'>

# Float com type hint
pi: float = 3.14159
print(pi, type(pi)) # Output: 3.14159 <class 'float'>

# String com type hint
nome: str = "Python"
print(nome, type(nome)) # Output: Python <class 'str'>

# Booleano com type hint
verdadeiro: bool = True
print(verdadeiro, type(verdadeiro)) # Output: True <class 'bool'>
```



@fpftech.educacional

Input()

Escreva() = print()

Como fazer o leia() no python?

python

 Copiar

```
variavel = input("Mensagem ou prompt para o usuário: ")
```

Exemplo Simples de Uso:

python

 Copiar

```
nome = input("Digite seu nome: ")  
print("Olá,", nome)
```

Por padrão, TODO o input é do tipo string.



Input() convertendo pra tipagem

Se por padrão, TODO o input é do tipo string... Como converter?

Exemplo com Conversão:

python

 Copiar

```
idade = int(input("Digite sua idade: "))  
print("Você tem", idade, "anos.")
```

No exemplo acima, está sendo convertido o valor para int()



Argumentos nomeados x argumentos posicionais

```
1 def soma_argumento_posicional(n1, n2):  
2     return n1 + n2  
3  
4 print(soma_argumento_posicional(1,2))
```

Ln: 4, Col: 38

 Run  Share  \$ Command Line Arguments

3

```
1 def soma_argumento_nomeado(n1, n2):  
2     return n1 + n2  
3  
4 print(soma_argumento_nomeado(n2=2, n1=1))
```

Ln: 3, Col: 1

 Run  Share  \$ Command Line Arguments

3



Função com valores padrão

```
1 def soma_argumento_nomeado(n1=0, n2=0):  
2     return n1 + n2  
3  
4 print(soma_argumento_nomeado(n2=2))
```

Ln: 4, Col: 36

 Run

 Share

\$

Command Line Arguments


2



@fpftech.educacional

print()

Print() também pode receber argumentos extras =

sep (separador)

end (final)

Esses 2 argumentos são nomeados, ou seja, não são posicionais.

Sintaxe do print()

python

 Copiar

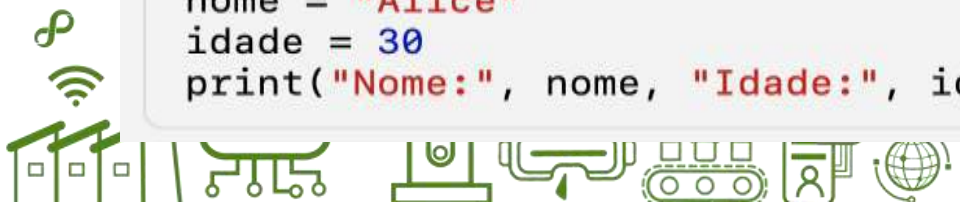
```
print(objeto1, objeto2, ..., sep=' ', end='\n')
```

Exemplo com Vários Objetos e Parâmetros:

python

 Copiar

```
nome = "Alice"  
idade = 30  
print("Nome:", nome, "Idade:", idade, sep=", ", end=".\\n")
```



F strings

Você pode usar a formatação de strings para criar saídas mais sofisticadas e legíveis.

Exemplo Usando F-strings (a partir do Python 3.6):

python

 Copiar

```
nome = "Carlos"  
idade = 25  
print(f"Meu nome é {nome} e eu tenho {idade} anos.")
```

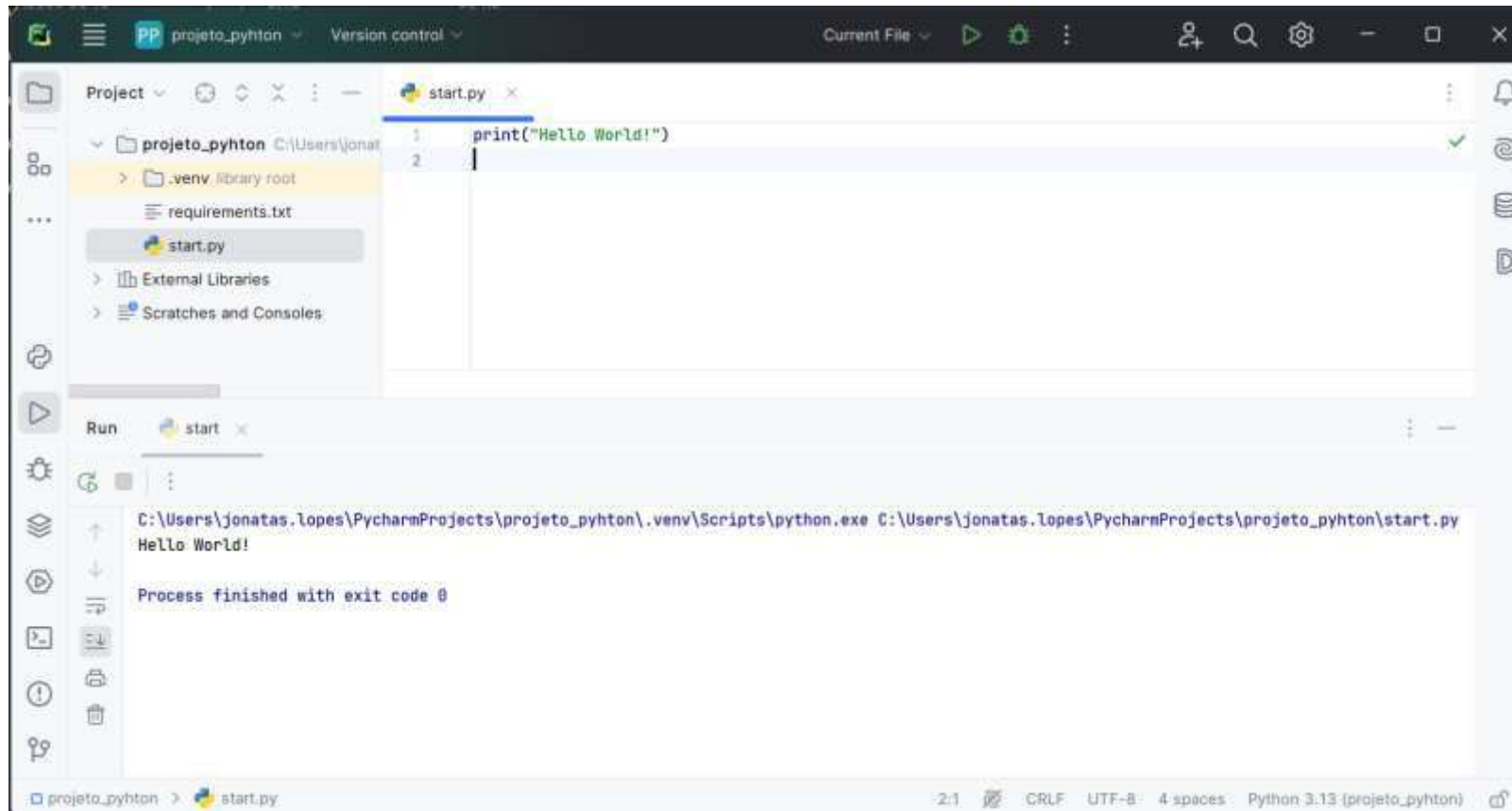
```
1 nome = "Carlos"  
2 idade = 25  
3 print("Meu nome é {} e eu tenho {} anos.".format(nome, idade))
```

```
1 nome = "Carlos"  
2 idade = 25  
3 print("Meu nome é {pess} e eu tenho {num} anos.".format(pess=nome, num=idade))
```

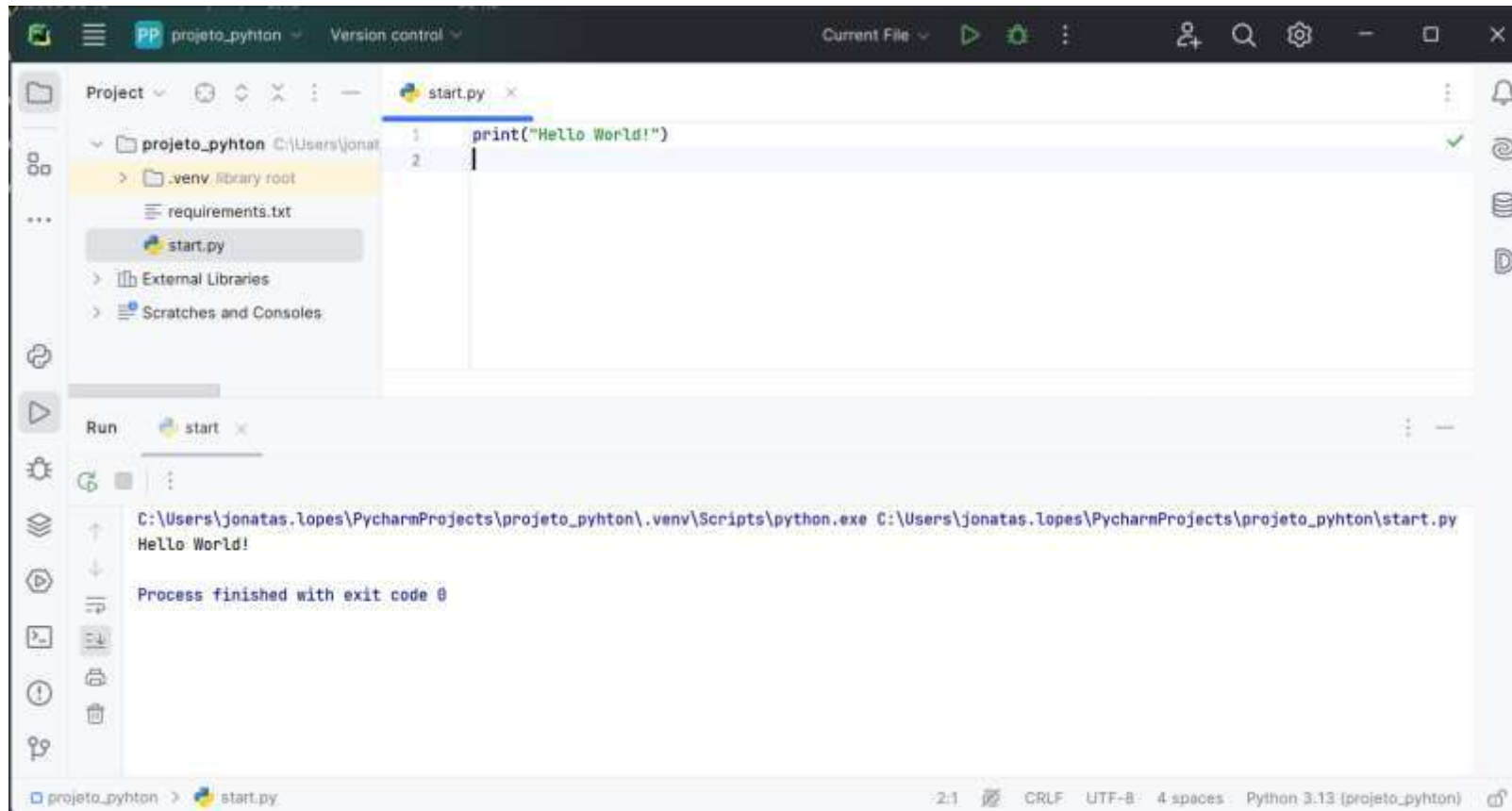


Botões IDE pycharm

- Diretorio de pastas
- Tabs de diferentes arquivos
- Área de código
- Botões controle janela auxiliar
- Janela auxiliar (debugar, terminal, git, etc)



Botões da IDE



- Botão pra rodar o programa/arquivo
- Output do programa/arquivo
- “finished with code 0” significa finalizou ok.
- Console python
(pode emular algumas coisas em python)
- Terminal
(pode escrever alguns códigos como pip install etc)










Welcome X

Visual Studio Code

Editing evolved






Start




-  New File...
-  Open File...
-  Open Folder...
-  Clone Git Repository...
-  Connect to...

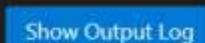
Recent

- python_codes_run C:\Users\jonatas.lopes
- bowling C:\Users\jonatas.lopes\Downloads\Unity_projects
- My project C:\Users\jonatas.lopes\Downloads\Unity_projects
- unity_planets_proj C:\Users\jonatas.lopes
- Essentials Project C:\Users\jonatas.lopes\Downloads
- More...

Walkthroughs

-  **Get Started with VS Code**
Customize your editor, learn the basics, and start coding
-  **Learn the Fundamentals**
-  **GitHub Copilot** Updated
-  **Get Started with C# Dev Kit** Updated
-  **Get Started with Python Development** Updated
- More...

 No access to GitHub Copilot found. You are currently logged in as jonsqlopes.  

Source: GitHub Copilot 

☒ Show welcome page on startup

FileEditSelectionViewGoRunTerminalHelppython_codes_run

python_codes_run

000extractor_handling.ipynb0 CREATE_saptags_4metadata.pytesting_new_extractors.ipynbUntitled-1game.py 1 XSettings

EXPLORER

PYTHON_CODES_RUN

.vscode

venv

0 CREATE schema and translations.py

0 CREATE_saptags_4metadata.py

0 MODIFY_excel_extractor_sap_data.py

000 SET_confest from landing.py

0 SET_timestamp_in_confest.py

0expoAmazonia_symplo_handling.ipynb

000extractor_handling.ipynb

B57d028d-3acb-4a19-9dcb-6d2d848e64b0_2...

b889f29a-39ef-47bc-a06c-d9dbe7050827_20...

browserAiScript_copy.ipynb

browserAiScript.ipynb

cdcef507-af64-41d6-ab0a-b37b89f19781_202...

change_keys_dict.py

change_sap_column_names.py

change_sap_tag_names.py

chat_dl_test_actions.py

chat_dl_test_dados_confirmacao_pedido.py

chat_dl_test_sap_models.py

chat_dl_test_test_base.py

chat_dl_test.py

check_if_data_value_in_keys.py

check_in_a_not_b.py

convert_excel_to_db.py

creating_dict_datetimes.py

debugging_bronze_browseai.py

duckdb_polars_test.ipynb

duckdb_infer_schema.py

estrut_append_unids_carga_cabecalho_nota.txt

game.py

lower_dict_keys.py

mat_texto_mat_caract_json

ollama_API.ipynb

python_print_columns_length.py

python_print_sap_dict.py

python_print_table_tags.py

run_duplicate_names.py

run_sql_duplicates.py

runPython.ru

OUTLINE

TIMELINE

game.py > ...

```
1 import pygame
2 import sys
3 import random
4
5 # Initialize Pygame
6 pygame.init()
7
8 # Constants
9 WIDTH, HEIGHT = 600, 400
10 FPS = 60
11 GRAVITY = 0.5
12 JUMP_SPEED = 10
13
14 # Colors
15 WHITE = (255, 255, 255)
16 BLACK = (0, 0, 0)
17
18 # Bird class
19 class Bird(pygame.sprite.Sprite):
20     def __init__(self):
21         super().__init__()
22         self.image = pygame.Surface((30, 30))
23         self.image.fill(WHITE)
24         self.rect = self.image.get_rect()
25         self.rect.center = (WIDTH // 4, HEIGHT // 2)
26         self.velocity = 0
27
28     def update(self):
29         self.velocity += GRAVITY
30         self.rect.y += self.velocity
31
32         # Keep the bird on the screen
33         if self.rect.bottom > HEIGHT:
34             self.rect.bottom = HEIGHT
35             self.velocity = 0
36         elif self.rect.top < 0:
37             self.rect.top = 0
```

PROBLEMS1

OUTPUT

DEBUG-CONSOLE

TERMINAL

JUPYTER

PORTS

POLYGLOT NOTEBOOK

Filter

Tasks

Ln 1, Col 1

Spaces: 4

UTF-8

CRLF

Python

Go Live

Prettier

Operações aritméticas

python

 Copiar

```
# Adição
a = 10
b = 5
resultado_adicao = a + b
print("Adição:", resultado_adicao) # Output: Adição: 15

# Subtração
resultado_subtracao = a - b
print("Subtração:", resultado_subtracao) # Output: Subtração: 5

# Multiplicação
resultado_multiplicacao = a * b
print("Multiplicação:", resultado_multiplicacao) # Output: Multiplicação: 50

# Divisão
resultado_divisao = a / b
print("Divisão:", resultado_divisao) # Output: Divisão: 2.0

# Divisão Inteira
resultado_divisao_inteira = a // b
print("Divisão Inteira:", resultado_divisao_inteira) # Output: Divisão Inteira: 2

# Módulo (Resto da divisão)
resultado_modulo = a % b
print("Módulo:", resultado_modulo) # Output: Módulo: 0

# Exponenciação
resultado_exponenciacao = a ** b
print("Exponenciação:", resultado_exponenciacao) # Output: Exponenciação: 100000
```



Operações relacionais

python

 Copiar

```
# Definindo valores
a = 10
b = 5

# Maior que
resultado_maior = a > b
print("Maior que:", resultado_maior) # Output: Maior que: True

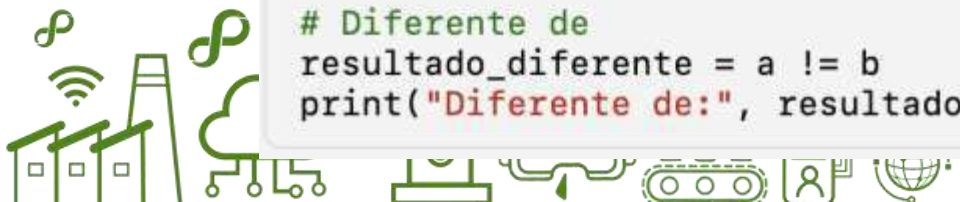
# Menor que
resultado_menor = a < b
print("Menor que:", resultado_menor) # Output: Menor que: False

# Maior ou igual a
resultado_maior_ou_igual = a >= b
print("Maior ou igual a:", resultado_maior_ou_igual) # Output: Maior ou igual a: True

# Menor ou igual a
resultado_menor_ou_igual = a <= b
print("Menor ou igual a:", resultado_menor_ou_igual) # Output: Menor ou igual a: False

# Igual a
resultado_igual = a == b
print("Igual a:", resultado_igual) # Output: Igual a: False

# Diferente de
resultado_diferente = a != b
print("Diferente de:", resultado_diferente) # Output: Diferente de: True
```



python

 Copiar

```
# Definindo variáveis
```

```
a = 10
```

```
b = 5
```

```
c = 7
```

```
# Exemplo sem parênteses usando 'and' e operadores relacionais
```

```
resultado = a > b and b < c
```

```
print("Resultado de a > b and b < c:", resultado) # Output: True
```

```
# Exemplo sem parênteses usando 'or' e operadores relacionais
```

```
resultado = a < b or c >= b
```

```
print("Resultado de a < b or c >= b:", resultado) # Output: True
```

```
# Exemplo sem parênteses usando 'not' para inverter o resultado
```

```
resultado = not a == b
```

```
print("Resultado de not a == b:", resultado) # Output: True
```



python

 Copiar

```
# Definindo variáveis
x = [1, 2, 3]
y = x # y e x referenciam o mesmo objeto
z = [1, 2, 3] # z é um objeto diferente, apesar de ter o mesmo conteúdo

# Comparação de identidade com o operador 'is'
resultado_is = x is y
print("Resultado de x is y:", resultado_is) # Output: True

# Comparação de identidade com um objeto diferente
resultado_is_not = x is z
print("Resultado de x is z:", resultado_is_not) # Output: False

# Usando 'is' com operadores lógicos
resultado_logico = x is y and z is not y
print("Resultado de x is y and z is not y:", resultado_logico) # Output: True
```



@fpftech.educacional

Obrigado!



    @fpftech.educacional