

# APRENDOENDO PYTHON COM RPG !

OFICINA DE INTRODUÇÃO A PROGRAMAÇÃO



RAPHAEL SANTOS & VINÍCIUS SARINHO

# QUEM NÓS SOMOS?

VINÍCIUS MEIRA SARINHO



- 7º SEMESTRE DA UNIFAN  
CURSANDO ENGENHARIA  
DE COMPUTAÇÃO
- PROGRAMAÇÃO SÓLIDA EM  
C E EM PYTHON/GODOT
- APRENENDENDO MAIS SOBRE  
DESENVOLVIMENTO DE  
JOGOS/DESING
- ESTAGIANDO ROBÓTICA  
EDUCACIONAL PARA  
CRIANÇAS  
NEURODIVERGENTES

# PROJETOS:



# QUEM NÓS SOMOS?

RAPHAEL SANTOS



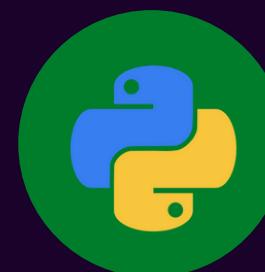
- 8º SEMESTRE DA UNIFAN CURSANDO ENGENHARIA DE COMPUTAÇÃO
- PROGRAMAÇÃO SÓLIDA EM JAVA, PYTHON E C
- APRENDENDO MAIS SOBRE DESENVOLVIMENTO DE JOGOS/DESING
- ESTAGIEI EM DESENVOLVIMENTO E MANUTENÇÃO DE MAQUINAS

# PROJETOS:

The screenshot shows the 'PREENCHER DADOS' (Fill Data) screen of the AutoSavvy Park application. On the left is a sidebar with icons for Home, Payment, Seats, Client List, Cost per Minute, and Cashier. The main area has four input fields: 'Placa' (License Plate), 'Modelo' (Model), 'Nome' (Name), and 'Contato' (Contact). Below these is a blue button labeled 'Adicionar Veículo +' (Add Vehicle +).



# MAPA DA JORNADA!



1. Entender o que é programar e por que Python é nossa escolha.



3. Conhecer os comandos essenciais do Python.



2. Preparar nosso ambiente de criação (VSCode).



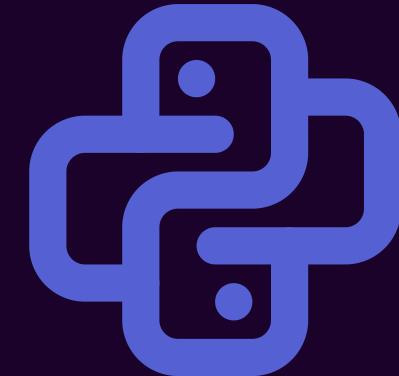
4. Construir nosso RPG, passo a passo.

# O QUE É PROGRAMAR?

Programar é **dar instruções extremamente precisas** para o computador executar uma tarefa. Pense nisso como uma receita de poção mágica:

- Você precisa dos ingredientes certos (números, textos).
- Na ordem exata (passo 1, passo 2).
- Se você errar um detalhe, a poção... ou o programa... não funciona!

# POR QUE PYTHON?



**Sintaxe Simples:** É quase como ler em inglês. O foco é na lógica, não em símbolos complexos. Java: `System.out.println("Olá")`; Python: `print("Olá")`



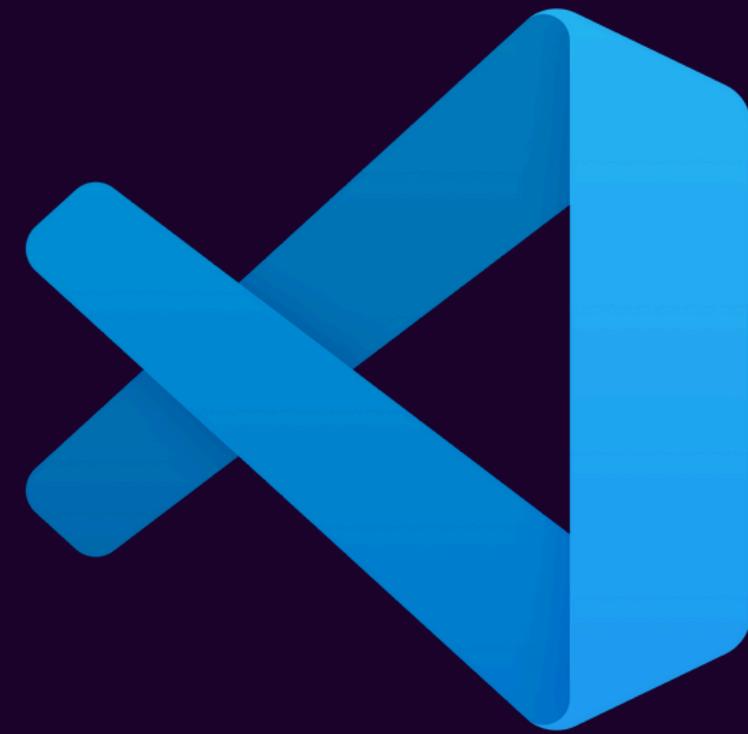
**Extremamente Versátil:** Usado para criar sites (Instagram), analisar dados (Netflix), em ciência (NASA) e, claro, jogos!



**Comunidade Gigante:** Se tiver uma dúvida, alguém no mundo já teve e a resposta está online. Você nunca está sozinho na sua jornada.

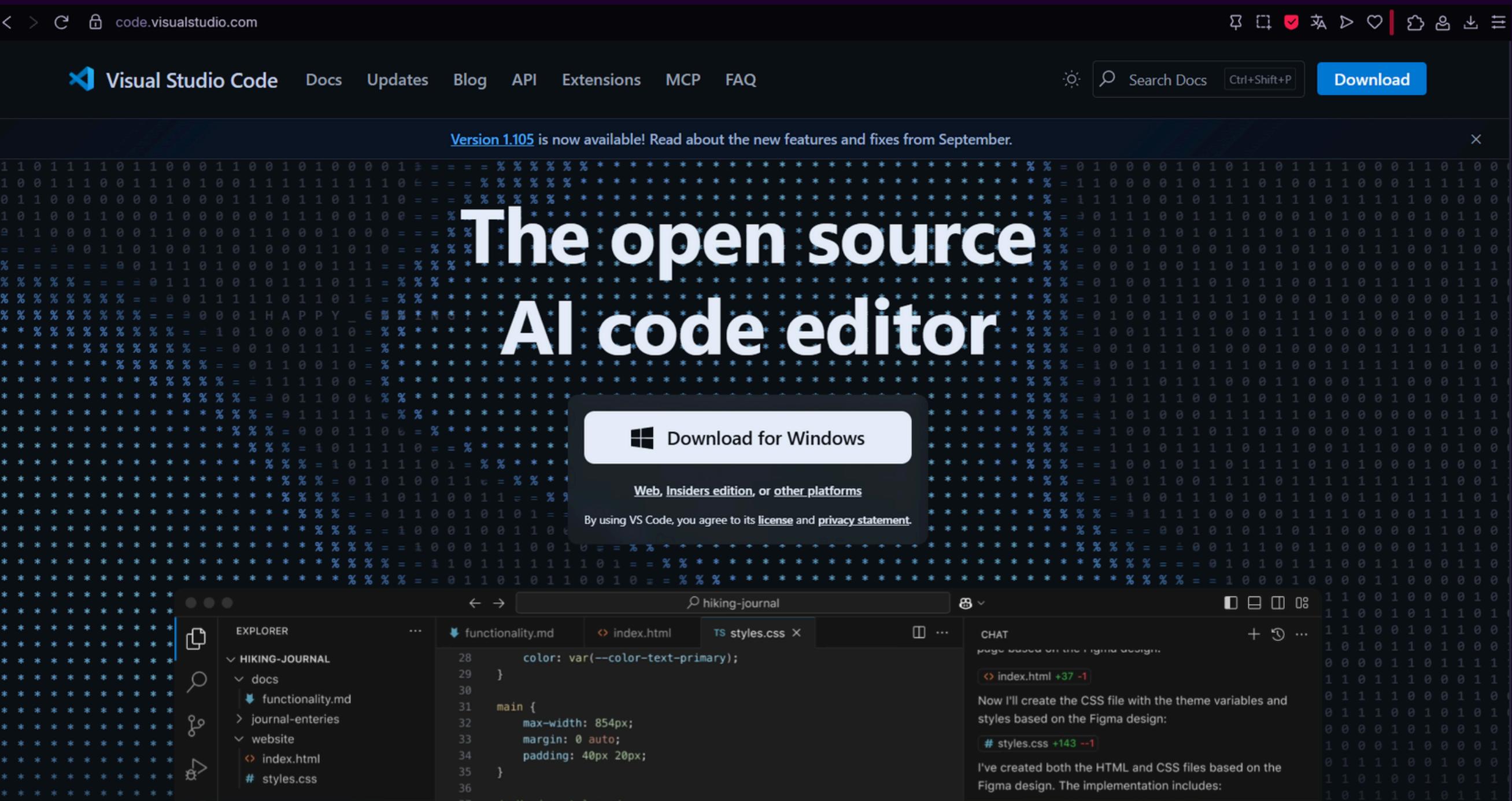


# Nossas Ferramentas de Aventura!



VISUAL CODE

# Nossas Ferramentas de Aventura!



The screenshot shows the homepage of the Visual Studio Code website ([code.visualstudio.com](https://code.visualstudio.com)). The page features a large, central text area with a pixelated background of binary code. The text reads "The open source AI code editor". Below this, there are download links for "Windows" and "Web, Insiders edition, or other platforms". A note at the bottom states: "By using VS Code, you agree to its [license](#) and [privacy statement](#)". The Visual Studio Code interface is visible at the bottom, showing the Explorer, Editor, and Chat panels. The Chat panel contains a message from a user named "Figma" about creating CSS files based on Figma designs.

Version 1.105 is now available! Read about the new features and fixes from September.

The open source  
AI code editor

Download for Windows

Web, Insiders edition, or other platforms

By using VS Code, you agree to its [license](#) and [privacy statement](#).

EXPLORER

HIKING-JOURNAL

docs

journal-entries

website

index.html

styles.css

functionality.md

index.html

TS styles.css

color: var(--color-text-primary);

}

main {

max-width: 854px;

margin: 0 auto;

padding: 40px 20px;

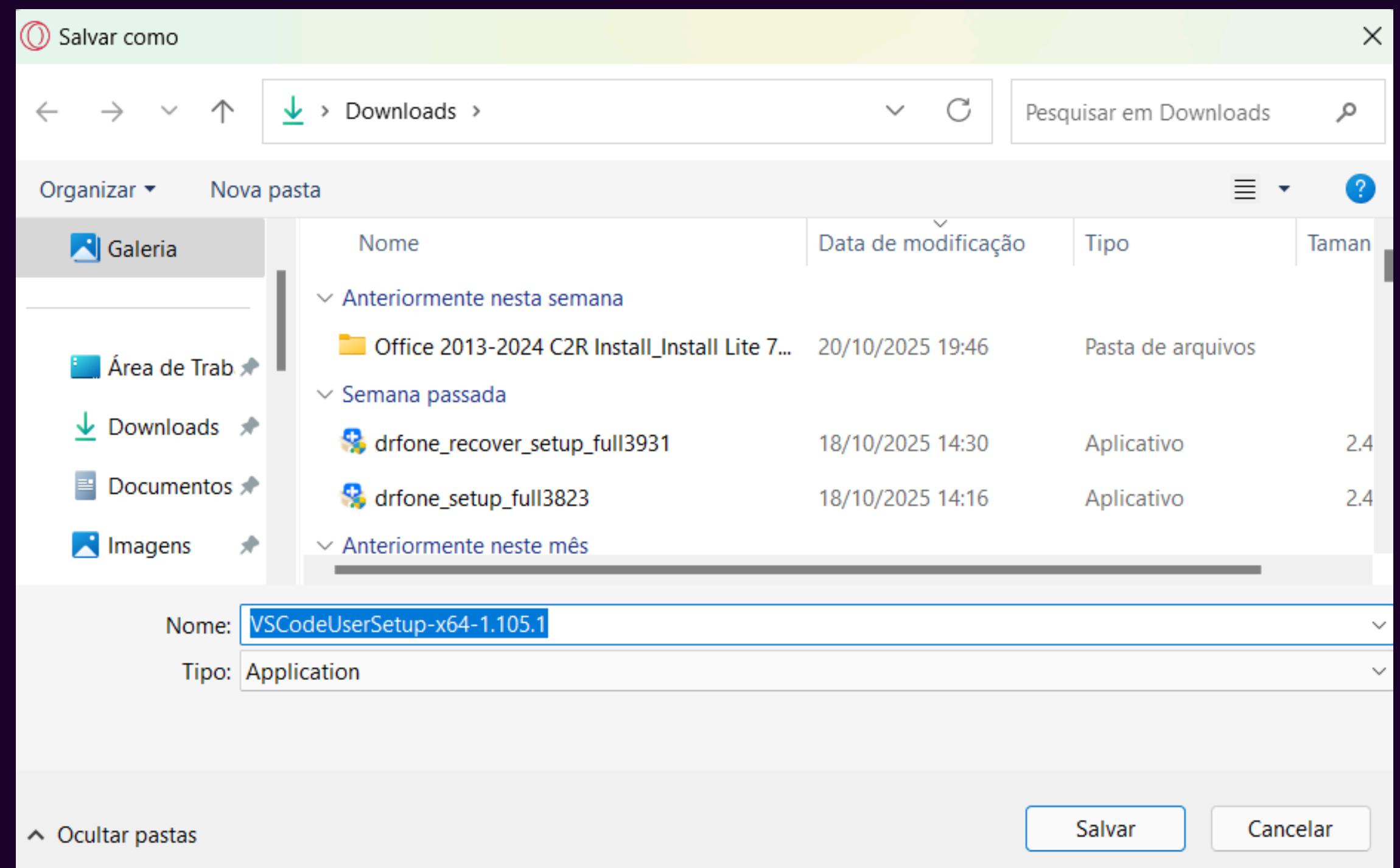
}

Now I'll create the CSS file with the theme variables and styles based on the Figma design:

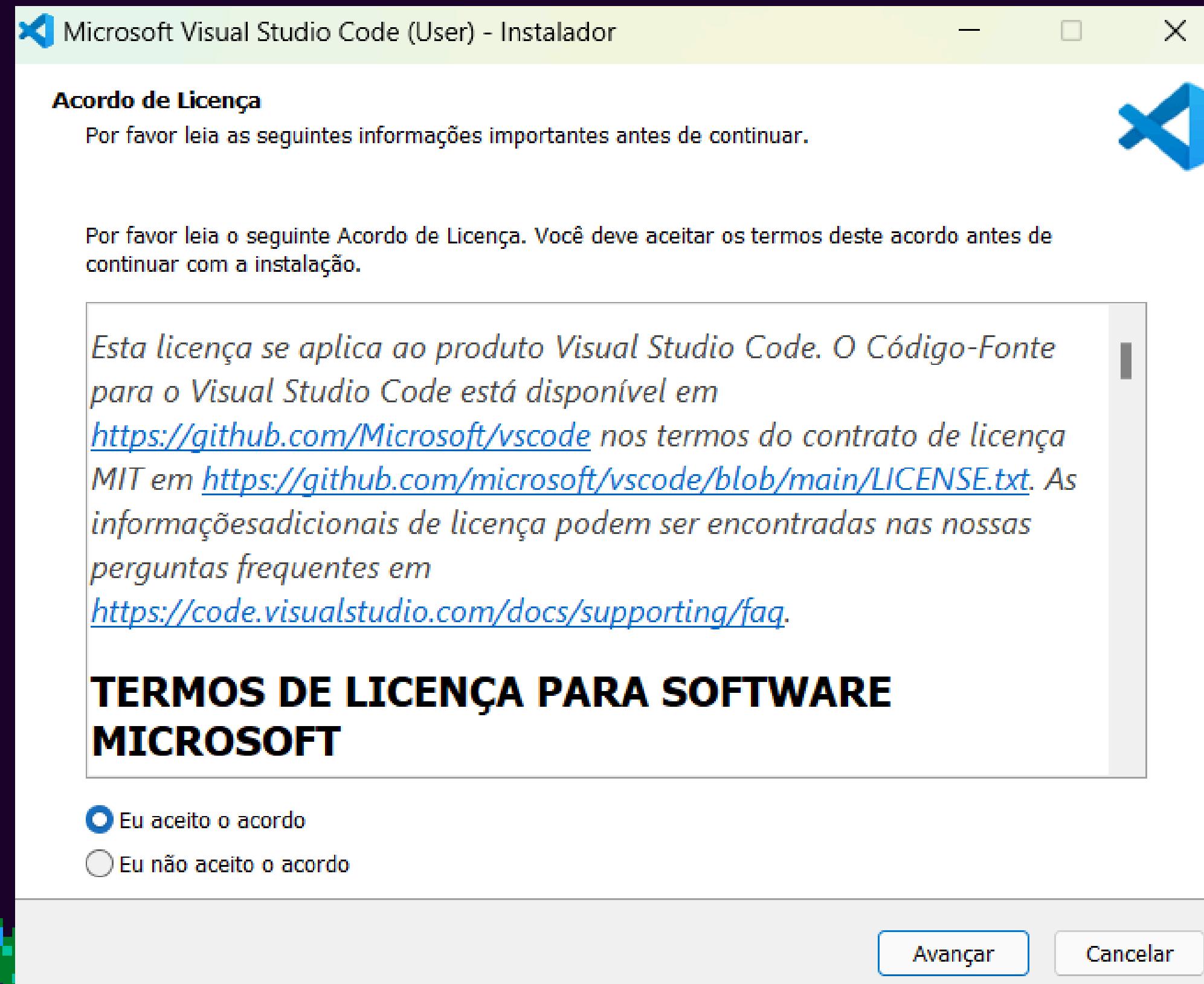
# styles.css +143 -1

I've created both the HTML and CSS files based on the Figma design. The implementation includes:

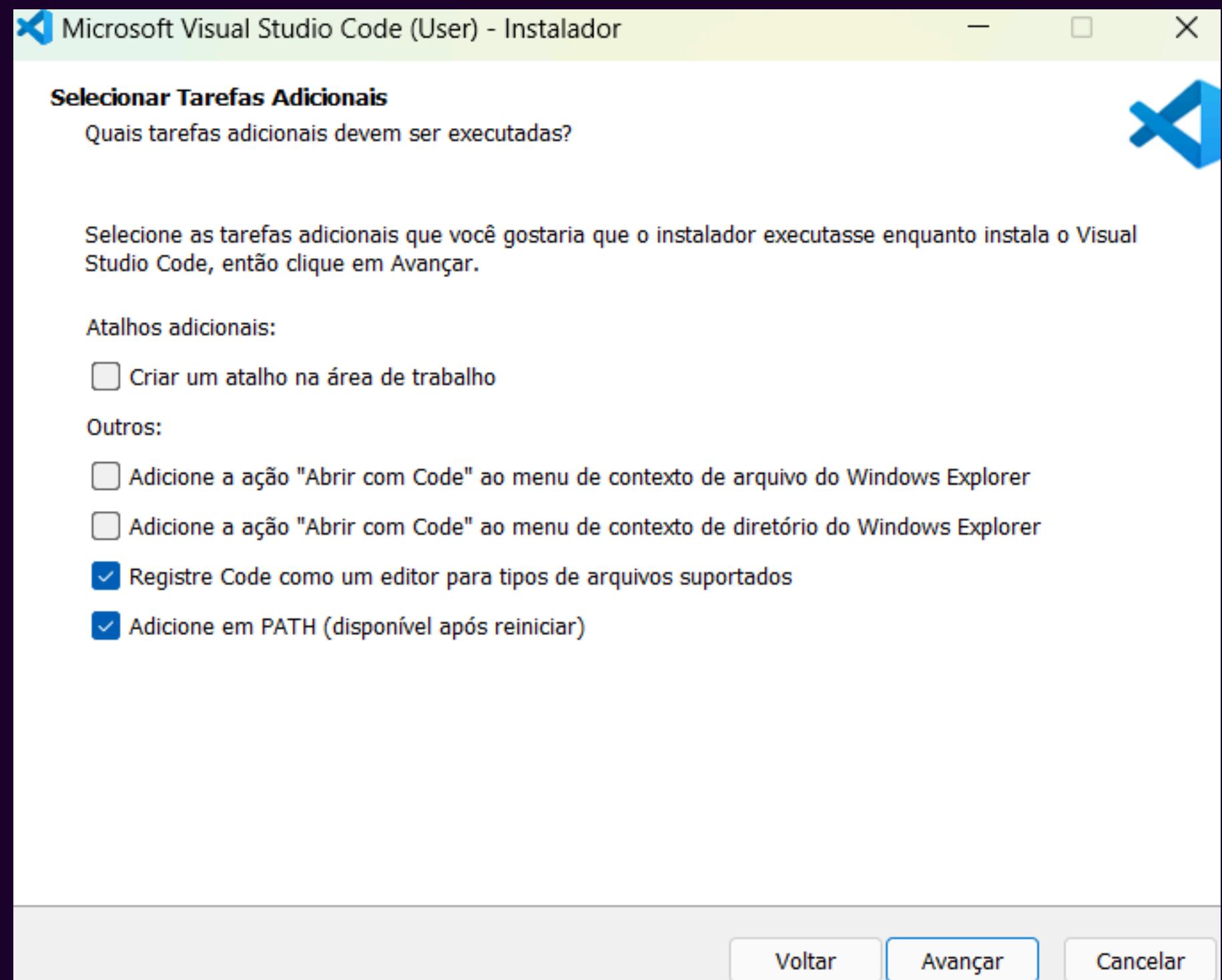
# Nossas Ferramentas de Aventura!



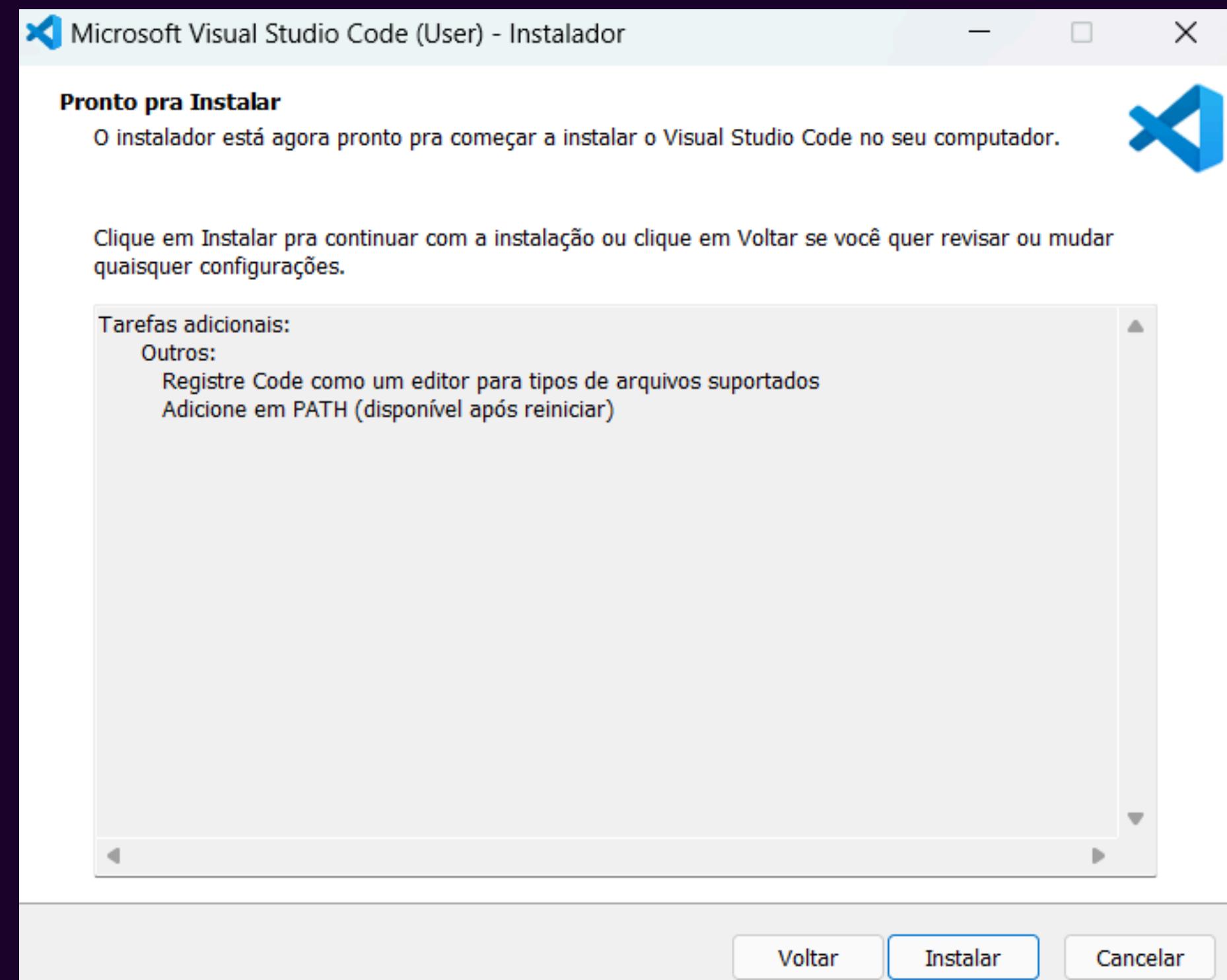
# Nossas Ferramentas de Aventura!



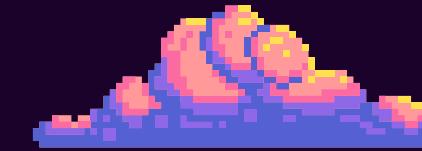
# Nossas Ferramentas de Aventura!



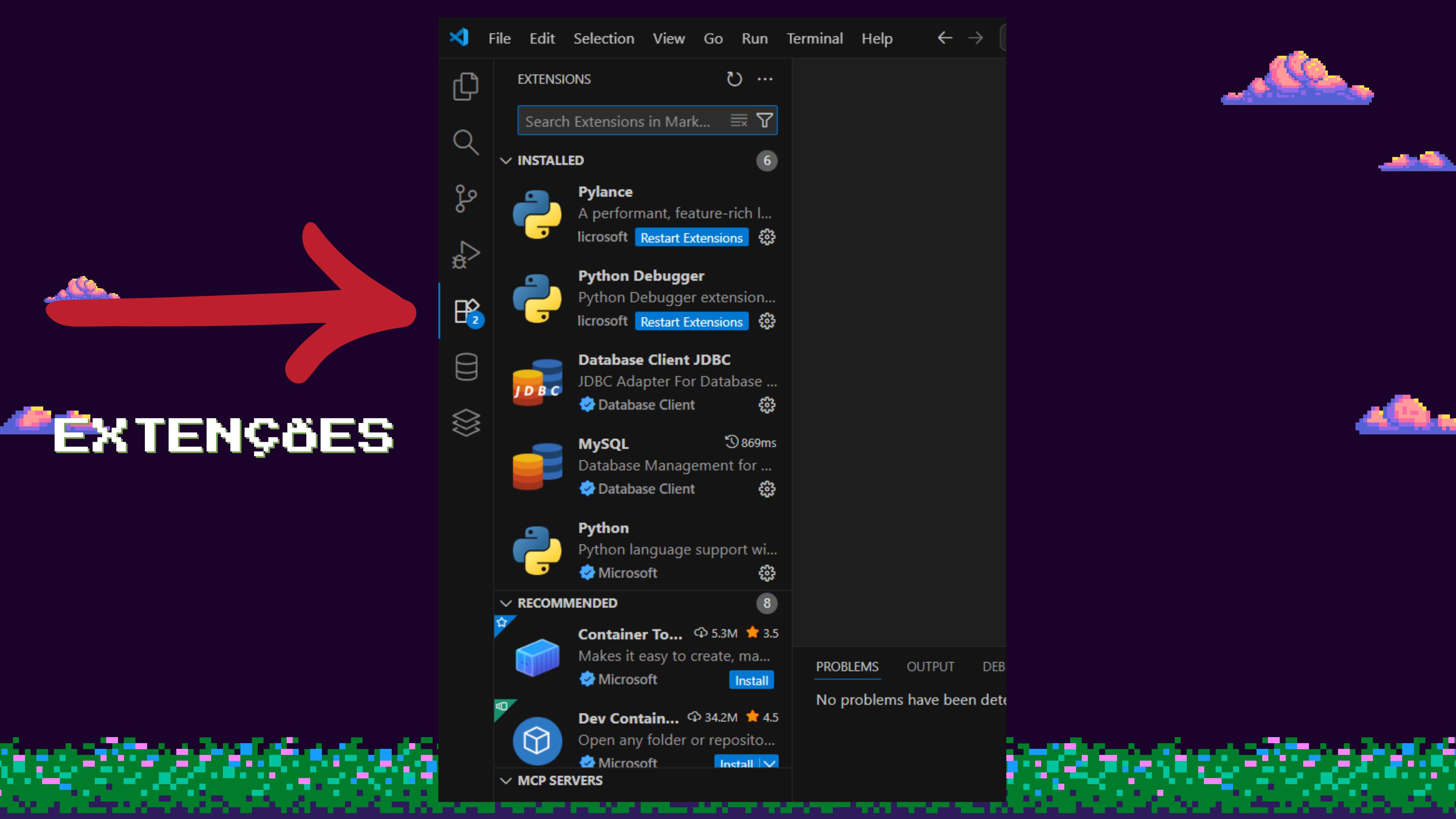
# NOSSAS FERRAMENTAS DE AVENTURA!



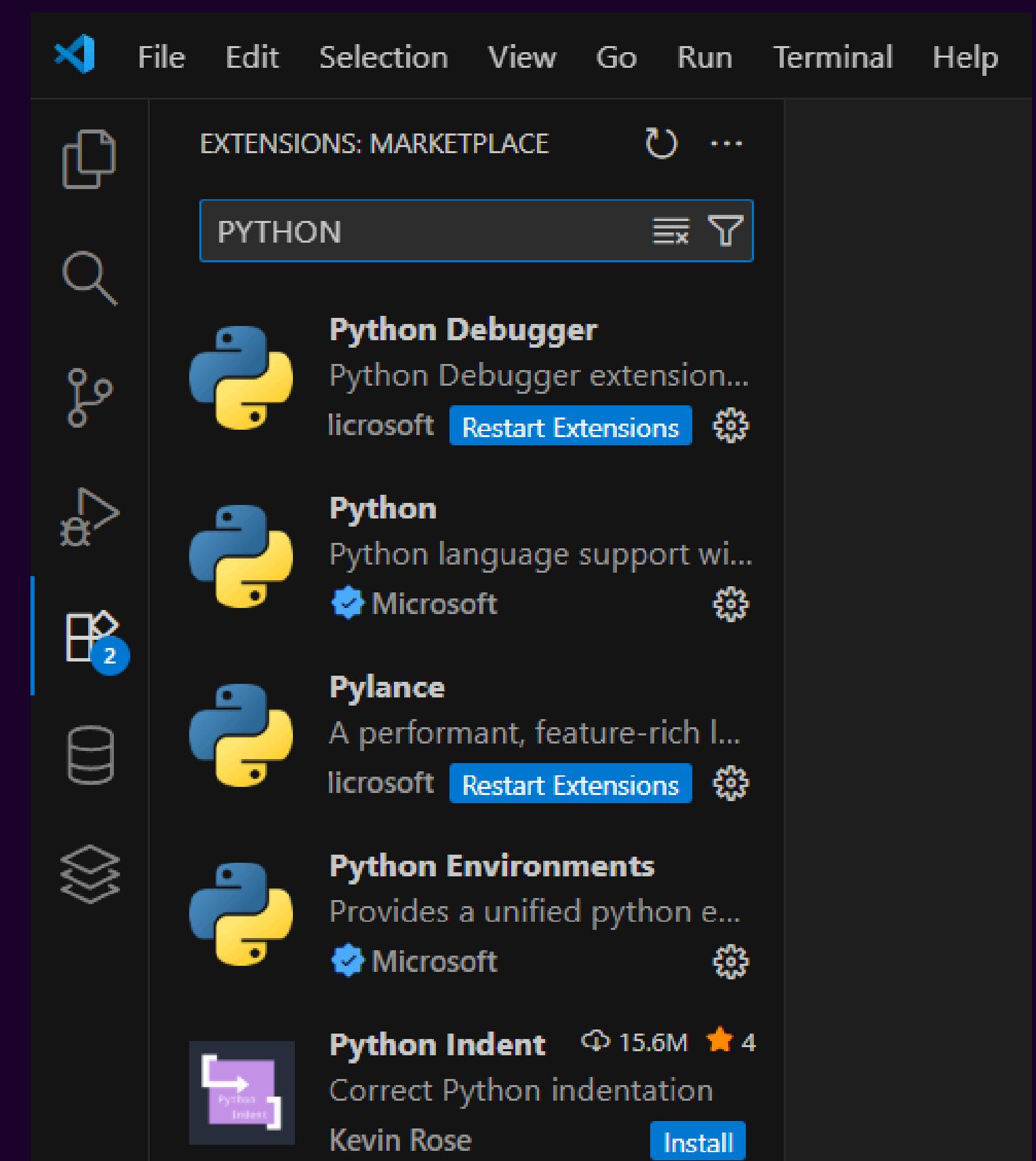
# NOSSAS FERRAMENTAS DE AVENTURA!

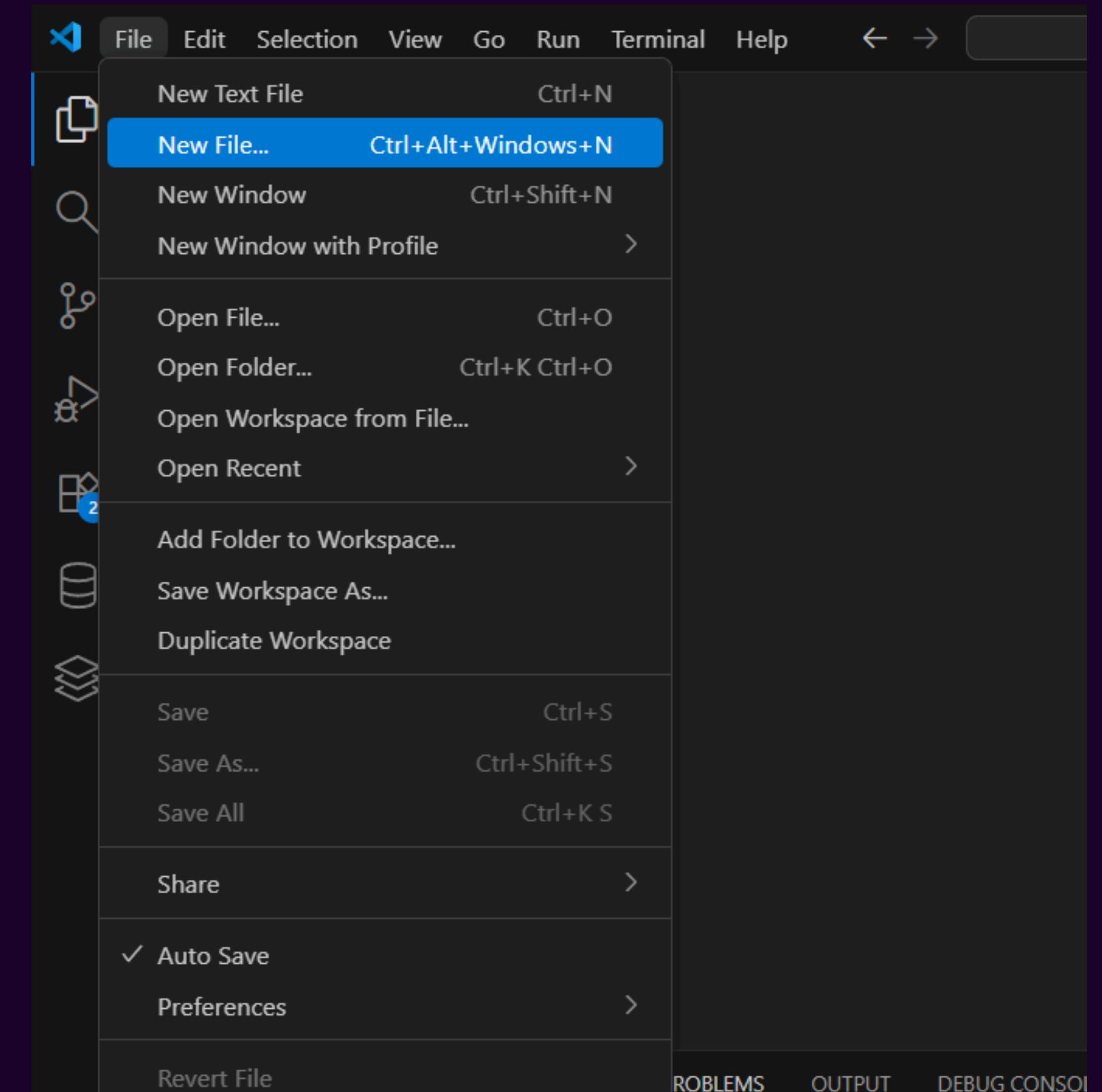


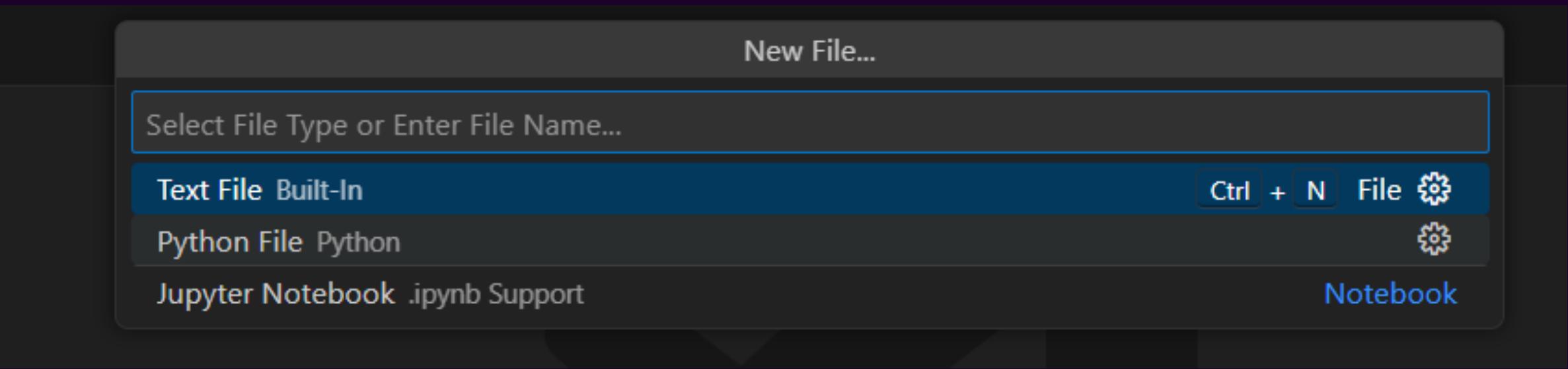
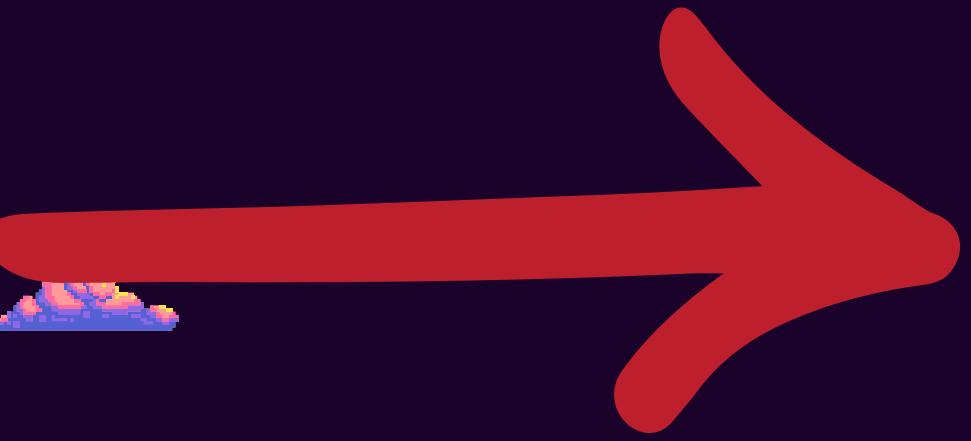
The screenshot shows the Visual Studio Code (VS Code) interface with a dark theme. The top bar includes the menu (File, Edit, Selection, View, Go, Run, Terminal, Help), a search bar (Q AdmJSK), and various status indicators. The left sidebar features the Explorer, Outline, and Timeline sections, with the Explorer showing a folder named 'ADMJSK'. The main workspace is dominated by a large, semi-transparent black 'X' logo. The bottom navigation bar includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS, along with a filter bar and terminal access keys. On the right side, there's a Chat interface with a message about 'Build with agent mode' and a 'Let's get started' button. A context menu is open in the center-right area, listing commands like 'Show All Commands' (Ctrl + Shift + P), 'Go to File' (Ctrl + P), 'Open Chat' (Ctrl + Alt + I), 'Open Settings' (Ctrl + ,), and 'Toggle Terminal' (Ctrl + T). The bottom status bar shows file paths and other system information.



# EXTENÇÕES

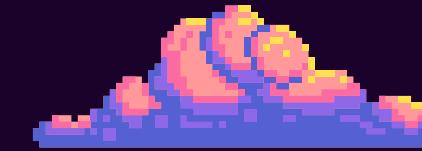
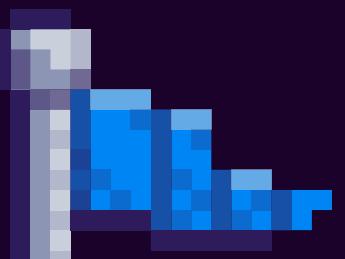
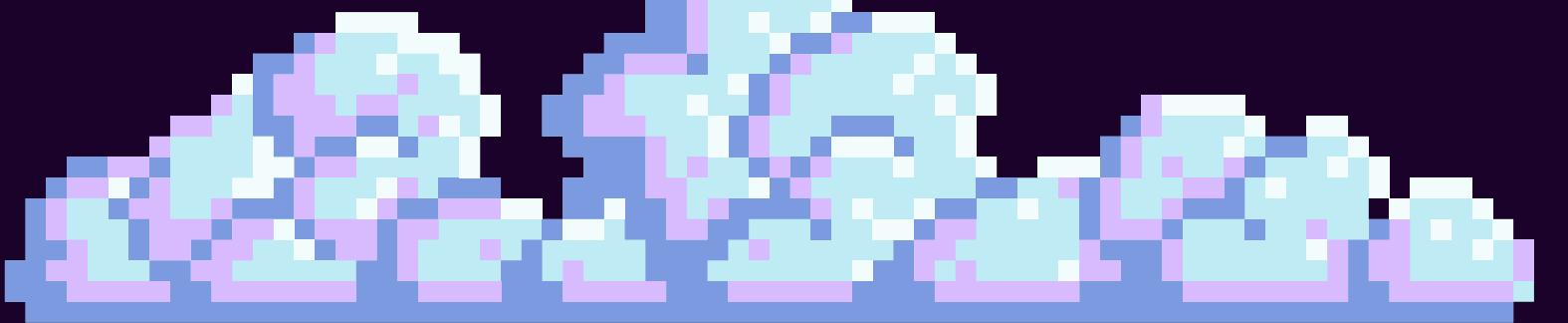
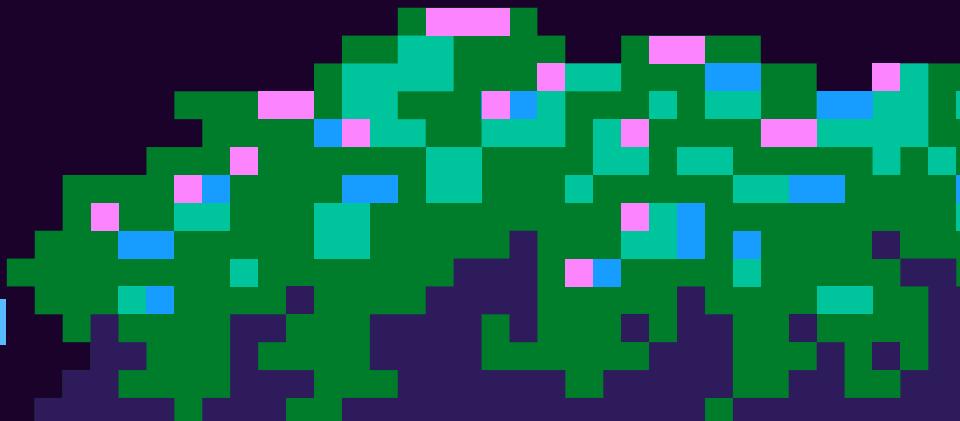
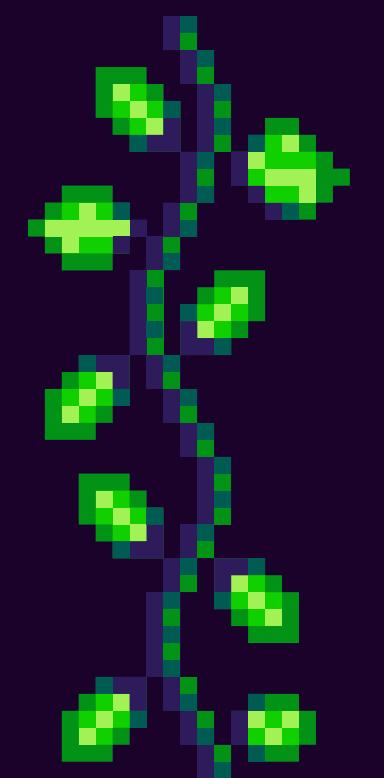






# ESCOLHER PYTHON?

# EXEMPLOS DE COMANDOS!



# OLÁ MUNDO

- Como eu sei o que está acontecendo no programa?
- E se eu quiser que o programa mostre algo?

EXEMPLOS:

```
print("Olá, mundo!")  
print(10 + 5)  
print("A soma de 10 + 5 é:", 10 + 5)
```

## Usamos Print!

# O QUE EU FAÇO SE EU QUERO GUARDAR ALGUMA INFORMAÇÃO?

Variaveis são como caixas em que guardamos informações para depois. Assim podemos reutilizar e até mesmo alterar variáveis

**EXEMPLOS:**

```
nome = "Lúcia"  
idade = 16  
vida = 100  
  
print(f"Nome: {nome}")  
print(f"idade : {idade}")  
print(f"Vida : {vida}")
```



**INPUT():**    O JOGADOR  
FALA COM    O JOGO

O input() deixa o usuário responder perguntas do programa.  
Assim o jogador participa da história!

**EXEMPLOS:**

```
nome = input("Qual é o seu nome? ")  
print("Bem-vindo,", nome)
```

# CONDICÕES

O computador toma decisões com:

**IF (se);**

**ELIF (senão se);**

**ELSE (senão).**

**EXEMPLOS:**

```
1 classe = input("1-Guerreiro 2-Mago 3-Arqueiro: ")
2
3 if classe == "1":
4     print("Você escolheu o Guerreiro!")
5 elif classe == "2":
6     print("Você escolheu o Mago!")
7 elif classe == "3":
8     print("Você escolheu o Arqueiro!")
9 else:
10    print("Opção inválida! Você será um Aldeão.")
```



# LISTAS E DICIONÁRIOS: GUARDANDO INIMIGOS

Listas guardam vários elementos; dicionários guardam informações nomeadas.

No exemplo abaixo, cada inimigo é uma ficha dentro de um baralho!

## EXEMPLOS:

```
inimigos = [  
    {"nome": "Goblin", "vida": 50, "ataque": 10},  
    {"nome": "Lobo", "vida": 40, "ataque": 8},  
]
```

# LOOP

Um loop é quando o computador repete uma ação várias vezes, até que algo mude.

Ele serve pra evitar repetir código manualmente, utilizando comando

## WHILE (ENQUANTO)

### EXEMPLOS:

```
i = 1
while i <= 5:
    print("Número:", i)
    i = i + 1
```

```
vida = 1
vida_inimigo = 2
while vida > 0 and vida_inimigo > 0:
    print(f"Vida do jogador: {vida}.\nVida do inimigo: {vida_inimigo}.")
    print("-----")
    vida -= 1
    vida_inimigo -= 1
```



```
import random

print("== BEM-VINDO AO MINI RPG EM PYTHON ==\n")

# --- CRIAÇÃO DO PERSONAGEM ---
nome = input("Qual é o nome do seu herói? ")

print("\nEscolha sua classe:")
print("1 - Guerreiro (forte e resistente)")
print("2 - Mago (usa magia poderosa)")
print("3 - Arqueiro (rápido e preciso)")

classe_opcao = input("Digite o número da classe: ")
```

```
if classe_opcao == "1":  
    classe = "Guerreiro"  
    vida = 120  
    ataque = 15  
elif classe_opcao == "2":  
    classe = "Mago"  
    vida = 80  
    ataque = 25  
elif classe_opcao == "3":  
    classe = "Arqueiro"  
    vida = 100  
    ataque = 18  
else:  
    print("Classe inválida! Você será um Aldeão.")  
    classe = "Aldeão"  
    vida = 60  
    ataque = 10
```

```
print(f"\n{nome} o {classe} entra em uma floresta sombria...\n")

# --- INIMIGOS ---
inimigos = [
    {"nome": "Goblin", "vida": 50, "ataque": 10},
    {"nome": "Lobo", "vida": 40, "ataque": 8},
    {"nome": "Orc", "vida": 80, "ataque": 12},
]

# Escolhe um inimigo aleatório
inimigo = random.choice(inimigos)
print(f"Um {inimigo['nome']} apareceu!\n")
```

```
# --- SISTEMA DE BATALHA ---
while vida > 0 and inimigo["vida"] > 0:
    print(f"Sua vida: {vida} | Vida do {inimigo['nome']}: {inimigo['vida']}")
    print("1 - Atacar")
    print("2 - Defender")
    print("3 - Fugir")

    acao = input("O que você faz? ")
```

```
if acao == "1":  
    dano = random.randint(ataque - 5, ataque + 5)  
    inimigo["vida"] -= dano  
    print(f"\nVocê atacou o {inimigo['nome']} e causou {dano} de dano!")  
elif acao == "2":  
    print("\nVocê se defende e reduz o próximo ataque!")  
    defesa = True  
elif acao == "3":  
    print("\nVocê fugiu da batalha! Covarde, mas vivo.")  
    break  
else:  
    print("\nAção inválida!")  
    continue
```

```
# Turno do inimigo
if inimigo["vida"] > 0:
    dano_inimigo = random.randint(inimigo["ataque"] - 3, inimigo["ataque"] + 3)
    if 'defesa' in locals() and defesa:
        dano_inimigo //= 2
        print("Você reduziu o dano pela metade!")
        defesa = False
    vida -= dano_inimigo
    print(f"O {inimigo['nome']} atacou e causou {dano_inimigo} de dano!\n")
```

```
# --- RESULTADO ---
if vida <= 0:
    print(f"\n💀 {nome} foi derrotado pelo {inimigo['nome']}... Fim de jogo.")
elif inimigo["vida"] <= 0:
    print(f"\n🏆 {nome} derrotou o {inimigo['nome']}! Vitória!")
```

**VAMOS  
AS  
MEJORES**

# CLASSES

Uma classe é como o molde de um personagem ou elemento do código. Ela define o que ele é (atributos) e o que ele pode fazer (métodos).

## EXEMPLOS:

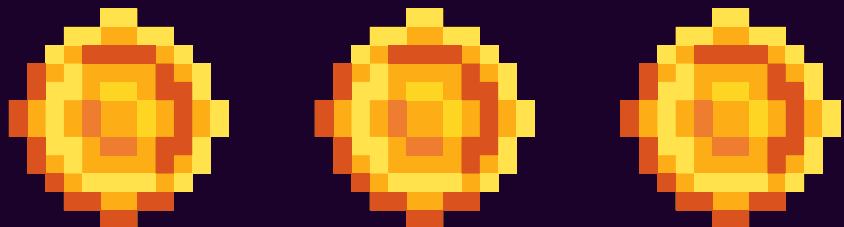
```
class Guerreiro:  
    def __init__(self, nome):  
        self.nome = nome  
        self.vida = 120  
        self.ataque = 15
```

# MÉTODOS

São as ações que a classes pode realizar — os comandos que ele sabe executar.

EXEMPLOS:

```
def atacar(self, inimigo):
    print(f"{self.nome} ataca {inimigo.nome}!")
    inimigo.vida -= self.ataque
```



```
import random

class Personagem:
    def __init__(self, nome, vida, ataque):
        self.nome = nome
        self.vida = vida
        self.ataque = ataque
        self.defendendo = False

    def atacar(self, alvo):
        dano = random.randint(self.ataque - 5, self.ataque + 5)
        print(f"{self.nome} ataca {alvo.nome}!")
        alvo.levar_dano(dano)

    def defender(self):
        self.defendendo = True
        print(f"{self.nome} se defende e reduzirá o próximo dano pela metade!")

    def levar_dano(self, dano):
        if self.defendendo:
            dano //= 2
            print(f"{self.nome} reduziu o dano pela metade!")
            self.defendendo = False
        self.vida -= dano
        print(f"{self.nome} sofreu {dano} de dano!")
```

```
class Guerreiro(Personagem):
    def __init__(self, nome):
        super().__init__(nome, vida=120, ataque=15)

class Mago(Personagem):
    def __init__(self, nome):
        super().__init__(nome, vida=80, ataque=25)

class Arqueiro(Personagem):
    def __init__(self, nome):
        super().__init__(nome, vida=100, ataque=18)

class Aldeao(Personagem):
    def __init__(self, nome):
        super().__init__(nome, vida=60, ataque=10)

class Inimigo(Personagem):
    # CORREÇÃO: __init__
    def __init__(self, nome, vida, ataque):
        super().__init__(nome, vida, ataque)
```

```
class Combate:  
    def __init__(self, heroi, inimigo):  
        self.heroi = heroi  
        self.inimigo = inimigo
```

```
def iniciar(self):  
  
    while self.heroi.vida > 0 and self.inimigo.vida > 0:  
        print(f"Vida de {self.heroi.nome}: {self.heroi.vida} | Vida de {self.inimigo.nome}: {self.inimigo.vida}")  
        print("1 - Atacar")  
        print("2 - Defender")  
        print("3 - Fugir")  
  
        acao = input("O que você faz? ")  
  
        if acao == "1":  
            self.heroi.atacar(self.inimigo)  
        elif acao == "2":  
            self.heroi.defender()  
        elif acao == "3":  
            print(f"{self.heroi.nome} fugiu da batalha!")  
            return  
        else:  
            print("Ação inválida!")  
            continue  
        if self.inimigo.vida > 0:  
            print(f"\nTurno de {self.inimigo.nome}:")  
            dano_inimigo = random.randint(self.inimigo.ataque - 3, self.inimigo.ataque + 3)  
            self.heroi.levar_dano(dano_inimigo)  
            print("----")  
  
        if self.heroi.vida <= 0:  
            print(f"\n💀 {self.heroi.nome} foi derrotado por {self.inimigo.nome}!")  
        elif self.inimigo.vida <= 0:  
            print(f"\n🏆 {self.heroi.nome} derrotou {self.inimigo.nome}! Vitória!")
```

```
print("== BEM-VINDO AO MINI RPG EM PYTHON ==\n")

nome = input("Qual é o nome do seu herói? ")

print("\nEscolha sua classe:")
print("1 - Guerreiro (forte e resistente)")
print("2 - Mago (usa magia poderosa)")
print("3 - Arqueiro (rápido e preciso)")

classe_opcao = input("Digite o número da classe: ")

if classe_opcao == "1":
    heroi = Guerreiro(nome)
elif classe_opcao == "2":
    heroi = Mago(nome)
elif classe_opcao == "3":
    heroi = Arqueiro(nome)
else:
    print("Classe inválida! Você será um Aldeão.")
    heroi = Aldeao(nome)
```

```
inimigos = [  
    Inimigo("Goblin", 50, 10),  
    Inimigo("Lobo", 40, 8),  
    Inimigo("Orc", 80, 12),  
]
```

```
inimigo_escolhido = random.choice(inimigos)  
Combate(heroi, inimigo_escolhido).iniciar()
```

OBRIGADO  
PELA  
ATENÇÃO!

