Explicación detallada de roupycore.py

1. Importaciones

import sys
import re

import sys:
Carga una herramienta que permite a nuestro programa interactuar con el sistema operativo (por ejemplo, leer el nombre del archivo que el usuario quiere abrir).

import re:
Carga otra herramienta que sirve para trabajar con patrones de texto (por ejemplo, buscar palabras específicas dentro de un texto).

2. El "Lexer" (convertir el texto en tokens)

def lexer(code):
Crea una función llamada lexer que recibe como entrada un pedacito de texto llamado code.

El trabajo de esta función es:

- Leer el texto,
- Reconocer qué significa cada pedazo,
- Y convertirlo en tokens (como pequeñas etiquetas que dicen "esto es una palabra", "esto es un igual", "esto es un texto", etc.).

Dentro de lexer:

```
token_patterns = [
        ('STRING', r'"[^"]*"'),
        ('ID', r'[A-Za-z_][A-Za-z0-9_]*'),
        ('ASSIGN', r'='),
        ('NEWLINE', r'\n'),
        ('SKIP', r'[ \t]+'),
        ('MISMATCH', r'.'),
]
```

Aquí estamos definiendo patrones para saber qué tipo de cosas queremos reconocer:

NOMBRE DEL TOKEN	QUE DETECTA	EJEMPLO
STRING	*TEXTO ENTRE COMILLAS*	*“HOLA MUNDO”*
ID	*NOMBRE DE ALGO*	*show, nombre, edad, etc...*
ASSIGN	*El simbolo*	*=*
NEWLINE	*UN SALTO DE LINEA*	*(Cuando apretas “Enter”)*
SKIP	*Espacios o tabulaciones*	*“ “ o “\t”*

MISMATCH	*Cualquier cosa que no reconocimos*	*Un error*

token_regex = '|'.join(f'(?P<{pair[0]}>{pair[1]})' for pair in token_patterns)
Esta línea junta todos los patrones en una sola regla gigante para poder buscar todos los tipos de tokens de una sola vez.

Ahora empieza a analizar el texto:

for mo in re.finditer(token_regex, code):
Va buscando coincidencias en el texto de entrada code.

Dentro del for:

kind = mo.lastgroup
value = mo.group()

kind es el tipo de token que encontró (STRING, ID, ASSIGN, etc.).

value es el texto que encontró (por ejemplo, "Hola mundo", o =).

Ahora se hacen decisiones:

if kind == 'SKIP':
    continue
Si es un espacio o tabulación, lo ignoramos (porque no nos importa).

elif kind == 'MISMATCH':
    raise RuntimeError(f"Caracter inesperado: {value!r}")

Si encontramos algo que no reconocemos, tiramos un error diciendo "carácter inesperado".

else:
    yield kind, value

Si es algo válido, lo devolvemos como un par: (tipo de token, valor).

Ejemplo final del lexer:
Si el texto es:

show = "Hola"
El lexer generaría:

('ID', 'show')
('ASSIGN', '=')
('STRING', '"Hola"')

3. Interpretar los tokens

def interpret_tokens(tokens):
Creamos una función que recibe una lista de tokens, y su trabajo es interpretar qué significan.

Dentro:

if tokens[0][0] == 'ID' and tokens[0][1] == 'show' and tokens[1][0] == 'ASSIGN':
Preguntamos:
¿La primera palabra es show y luego viene un =?

if tokens[2][0] == 'STRING':
Luego:
¿Después viene un texto entre comillas?

Si todo eso es cierto:

text = tokens[2][1][1:-1]
print(text)

Saca las comillas del texto,

Imprime el contenido en pantalla.

Ejemplo:
Si los tokens eran show = "Hola", el programa mostraría:

Hola

4. Leer el archivo .rp

def run_rouy_file(file_path):
Esta función abre un archivo que el usuario le pasa (por ejemplo, historia.rp).

```
with open(file_path, 'r') as file:
    lines = file.readlines()
```

Abre el archivo,

Lee todas las líneas del archivo en una lista.

Para cada línea:

```
for line in lines:
    tokens = list(lexer(line.strip()))
    interpret_tokens(tokens)
```

Elimina espacios sobrantes,

Convierte la línea en tokens usando lexer,

Interpreta los tokens usando interpret_tokens.

5. Otro interpretar (viejo)

def interpret_line(line):

Esta función era del código original:
Interpretaba el texto sin lexer, directamente viendo si empezaba con show =.

No se usa ahora, porque ya tenemos el lexer. (Aunque podrías borrarla más adelante si quieres).

6. Puntos de entrada

if __name__ == "__main__":
Esta línea dice:
"Si ejecutamos este archivo directamente, hacé esto:"

Dentro:

```
if len(sys.argv) < 2:
    print("Usage: python roupycore.py <file.rp>")
```

Si el usuario no escribió el nombre de un archivo .rp,
mostramos un mensaje explicándole cómo debe usar el programa.

```
else:
    filename = sys.argv[1]
    run_rouy_file(filename)
```

Si el usuario sí pasó un archivo,
lo guardamos como filename,
y ejecutamos ***run_rouy_file(filename).***

python roupycore.py historia.rp
Se abre el archivo historia.rp.

Se lee cada línea.

Cada línea se convierte en tokens.

Se interpretan los tokens.

Si la línea dice show = "Hola", el programa escribe Hola en pantalla.