

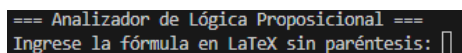
# Manual de uso

## 1. Uso del programa

Este programa toma como entrada una fórmula lógica escrita en LaTeX, sin paréntesis, y devuelve su versión bien formada, respetando la jerarquía de operadores definida en el archivo `tabla.csv`.

### 1.1. Entradas aceptadas

Una vez ejecutado el programa, aparecerá un recuadro de texto como el siguiente:



```
=== Analizador de Lógica Proposicional ===
Ingrese la fórmula en LaTeX sin paréntesis: □
```

Figura 1: Interfaz de entrada

Posteriormente, se debe ingresar la fórmula que se desee transformar en una fórmula bien formada, con las siguientes características:

- La fórmula debe estar escrita utilizando símbolos estándar de LaTeX.
- No debe incluir paréntesis manualmente. El programa los colocará automáticamente según la precedencia de operadores.
- El programa acepta los operandos `p`, `q`, `r`. (Se pueden añadir más operandos; el proceso se explica en la subsección 1.2.)
- El programa solo acepta los operadores básicos de la lógica proposicional, que son:
  - Negación: `\neg`
  - Conjunción: `\wedge`
  - Disyunción: `\vee`
  - Implicación: `\rightarrow`
  - Bicondicional: `\leftrightarrow`

## 1.2. Añadir operandos

Tener solo tres operandos puede no ser suficiente en fórmulas más complejas. Afortunadamente, es posible añadir más operandos:

Primero, el usuario debe dirigirse al archivo `tokenizer.py`, ubicado en la carpeta `logic`:

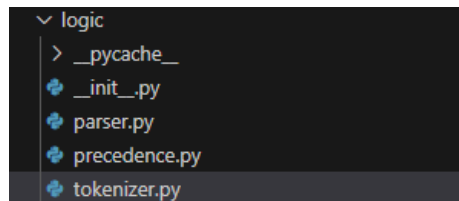


Figura 2: Ubicación del archivo `tokenizer.py`

Luego, debe localizar la sección de operadores y operandos válidos, cerca del inicio del archivo:

```
class LaTeXTokenizer:
    """tokenizador para fórmulas LaTeX de lógica proposicional"""

    def __init__(self):
        """Inicializa el tokenizador con las reglas de reemplazo"""
        self.latex_replacements = {
            '\\neg': 'neg',
            '\\vee': 'vee',
            '\\wedge': 'wedge',
            '\\rightarrow': 'rightarrow',
            '\\leftrightarrow': 'leftrightarrow',
        }

        # Operadores y operandos válidos
        self.binary_operators = {'vee', 'wedge', 'rightarrow', 'leftrightarrow'}
        self.unary_operators = {'neg'}
        self.operands = {'p', 'q', 'r'}
        self.all_operators = self.binary_operators.union(self.unary_operators)
```

Figura 3: Sección de operadores y operandos válidos

Una vez allí, solo es necesario añadir los nuevos operandos deseados, por ejemplo, `s` y `t`:

```
# Operadores y operandos válidos
self.binary_operators = {'vee', 'wedge', 'rightarrow', 'leftrightarrow'}
self.unary_operators = {'neg'}
self.operands = {'p', 'q', 'r', 's', 't'}
self.all_operators = self.binary_operators.union(self.unary_operators)
```

Figura 4: Ejemplo de nuevos operandos

Después de realizar estos cambios, guarde el archivo. No se requieren pasos adicionales.

## 1.3. Ejemplos de fórmulas aceptadas

A continuación, se presenta una lista de ejemplos válidos que se pueden ingresar al programa:

- $\neg p \wedge q$
- $p \rightarrow q \vee r$
- $p \leftrightarrow \neg q$
- $\neg p \leftrightarrow q \rightarrow r$

## 2. Jerarquía de operadores

Si se desea modificar la jerarquía de los operadores que utiliza el programa, se debe acceder al archivo `tabla.csv`, ubicado en la carpeta `data`:

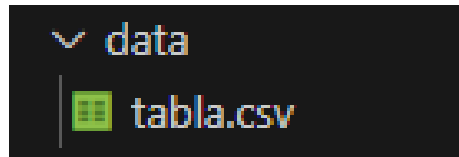


Figura 5: Ubicación del archivo `tabla.csv`

En dicho archivo se encuentra una tabla con la siguiente estructura:

```
,neg,wedge,vee,rightarrow,leftrightarrow
neg,>,>,>,>
wedge,<,>,>,>
vee,<,<,>,>
rightarrow,<,<,<,<
leftrightarrow,<,<,<,>
$,neg,wedge,vee,rightarrow,leftrightarrow
```

Figura 6: Ejemplo de tabla de precedencia

Para modificar la jerarquía, simplemente edite los signos "<" o ">" según lo deseado.

La tabla se interpreta como una matriz: el primer elemento de cada fila define su precedencia respecto a cada operador listado en las columnas. Por ejemplo, si se desea que `\wedge` tenga mayor jerarquía que `\neg`, el archivo se debe modificar como sigue:

```
,neg,wedge,vee,rightarrow,leftrightarrow
neg,>,<,>,>
wedge,>,>,>,>
```

Figura 7: Modificación de la jerarquía

Con este cambio, el programa reconocerá que `\wedge` tiene mayor jerarquía que `\neg`.

### 3. Posibles errores

- `pop from empty list`: probablemente hay un problema con el orden de operadores unarios (`\neg`) sin suficientes operandos.
- Error: '\$': probablemente falta una fila especial en `tabla.csv` que incluya el símbolo \$ como clave.