Lenguaje LOGIC

Lenguaje para la evaluación y simplificación de expresiones de lógica proposicional.

Jesus David Ayala Morales
Ana Sofía Matti Ríos
Daniel Eduardo Alvarez Terrazas

Objetivo del Proyecto: Crear un lenguaje que permita la simplificación y evaluación de expresiones de lógica proposicional, además de una interfaz interactiva para el usuario.

Resumen de Funcionalidades:

- Parseo de expresiones en texto a estructuras abstractas.
- Simplificación basada en reglas de lógica.
- Evaluación de expresiones usando un entorno personalizado.

Componentes Principales:

- *Parser:*

Convierte cadenas de texto como `"(p and q)"` en estructuras como `(l-and (l-atom 'p) (l-atom 'q))`.

- *Simplificador:*

Aplica reglas de lógica proposicional (e.g., idempotencia, absorción) para reducir expresiones a formas más simples.

- *Evaluador:*

Evalúa expresiones lógicas dado un entorno que define valores para las variables.

- *REPL Interactivo:*

Proporciona una interfaz gráfica para ingresar, simplificar, y evaluar expresiones.

- *Estructuras de Datos*:

Uso de tipos como `l-and`, `l-or`, `l-not`, `l-atom` para representar expresiones lógicas.

Entornos (`env`) extendidos para asociar variables con valores.

Lenguaje Utilizado: Racket

Archivos Principales:

- `parse.rkt`: Maneja el análisis de las expresiones.

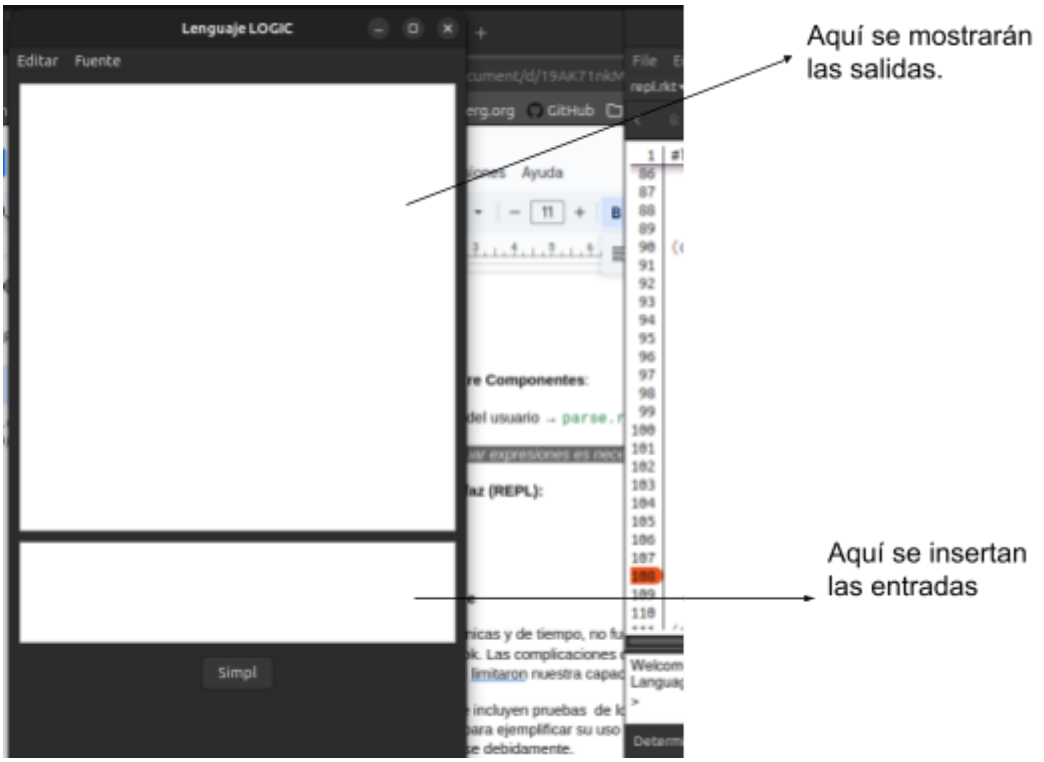
- `simpl.rkt`: Contiene el simplificador y las reglas aplicadas.
- `eval.rkt`: Implementa el evaluador lógico.
- `repl.rkt`: Proporciona la interfaz gráfica y conecta los componentes.

Interacción entre Componentes:

- Entrada del usuario → `parse.rkt` → `simpl.rkt` → Salida formateada

Nota: Para evaluar expresiones es necesario hacerlo desde la terminal de racket.

Uso de la interfaz (REPL):



Ejemplo: