UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ciencias

Lenguajes de Programación

MiniLisp

Proyecto 1

Presenta:
Lugo Díaz Ordaz Gretel Alexandra
Ramírez Juárez María Fernanda
Rojo Peña Manuel Ianluck

Profesor:
Manuel Soto Romero

Ayudantes:
Diego Méndez Medina
Erick Daniel Arroyo Martínez

Grupo: 7121, 2026-1

Fecha de entrega:
11 de octubre, 2025

Índice

Capítulo 1

Introducción

Como hemos visto en el curso Leguajes de Programación, al menos hasta la fecha de entrega del presente proyecto, en el desarrollo de un lenguaje de programación resulta fundamental comprender cómo se definen formalmente sus componentes y cómo estos se traducen a estructuras que una computadora puede interpretar y ejecutar.

1.1. Motivación

En el desarrollo de lenguajes de programación, Una de las motivaciones principales de este proyecto es acercarse al diseño de un lenguaje minimalista —en este caso, MINILISP— que permita practicar la construcción de gramáticas formales, analizadores léxicos y sintácticos, así como el modelado de árboles de sintaxis abstracta en un entorno académico.

1.2. Objetivos

El objetivo del proyecto es implementar un subconjunto del lenguaje Lisp con un conjunto reducido pero representativo de operaciones: expresiones aritméticas y booleanas, estructuras de control (if, cond), mecanismos de definición local (let, letrec, let*), funciones anónimas (lambda), listas y pares. Para ello, se diseña una gramática en notación BNF/EBNF, se define un conjunto de tokens para el análisis léxico y se construyen las estructuras de datos necesarias en Haskell para representar el árbol de sintaxis abstracta (ASA). De esta manera, se busca no solo capturar la semántica básica del lenguaje, sino también poner en práctica técnicas de diseño de compiladores a pequeña escala.

1.3. Delimitación del Proyecto

La delimitación del proyecto consiste en que MINILISP no pretende ser una implementación completa de Lisp, sino una versión simplificada con fines didácticos. Se restringe el conjunto de operaciones soportadas, se omite el manejo de macros y de entrada/salida, y se centra únicamente en el análisis sintáctico y la representación interna de programas. Con

esto, se logra un balance entre la complejidad teórica y la viabilidad de implementación en el tiempo disponible.

Definimos la Gramática para MiniLisp en **EBNF**:

Gramática MINILISP

```
<Expr>  ::=  <Var>
          |  <Num>
          |  <Bool>
          |  (+ <Expr> <Expr> {<Expr>})
          |  (- <Expr> <Expr> {<Expr>})
          |  (* <Expr> <Expr> {<Expr>})
          |  (/ <Expr> <Expr> {<Expr>})
          |  (add1 <Expr>)
          |  (sub1 <Expr>)
          |  (sqrt <Expr>)
          |  (expt <Expr>)
          |  (not <Expr>)
          |  (= <Expr> <Expr> {<Expr>})
          |  (<  <Expr> <Expr> {<Expr>})
          |  (>  <Expr> <Expr> {<Expr>})
          |  (<= <Expr> <Expr> {<Expr>})
          |  (>= <Expr> <Expr> {<Expr>})
          |  (!= <Expr> <Expr> {<Expr>})
          |  (<Expr>, <Expr>)
          |  (fst <Expr>)
          |  (snd <Expr>)
          |  (let ((<Var> <Expr>) {<Var> <Expr>}) <Expr>)
          |  (letrec <Var> <Expr> <Expr>)
          |  (let* ((<Var> <Expr>) {<Var> <Expr>}) <Expr>)
          |  (if0 <Expr> <Expr> <Expr>)
          |  (if <Expr> <Expr> <Expr>)
          |  (lambda (<Var> {<Var>}) <Expr>)
          |  (<Expr> <Expr>)
          |  [<Expr> {, <Expr>}]
          |  (head <Expr>)
          |  (tail <Expr>)
          |  (cond [<Expr> <Expr>] {[<Expr> <Expr>]} [else <Expr>])

 <Var>  ::=  Identificador de variable
 <Num>  ::=  Constante entera
<Bool>  ::=  #t | #f
```

Hacemos un abuso de notación para aclarar que el uso de '**[**' '**]**' no es para indicar opcionalidad de la notación de EBNF sino que son los símbolos que usamos para representar

listas, también por eso están en negritas.

Capítulo 2

Formalización

2.1. Sintaxis Concreta/Léxica

2.2. Sintaxis Libre de Contexto

2.3. Eliminación de azúcar sintáctica

2.4. Semántica operacional

2.4.1. Notaciones, derivaciones y reglas de inferencia correspondientes

Capítulo 3

Justificaciones

Capítulo 4

Investigación

4.1. Contextualización conceptual (ej. syntactic sugar, SOS, EBNF),

4.2. Referencias históricas y aportes relevantes de la disciplina.

Capítulo 5

Resultados

Capítulo 6

Conclusiones

Bibliografía

Bibliografía

[1] Autor. *Título del libro*. Editorial, Año.

[2] Autor. .ªrtículo". Revista, Año.