УДК 004.43

Колібабчук Д. І., здобувач вищої освіти,
Донецький національний університет імені Василя Стуса

S-ВИРАЗИ, ГОМОІКОННІСТЬ ТА REPL: АНАЛІЗ АРХІТЕКТУРИ LISP

Анотація. Тези присвячено аналізу філософії Lisp. Розглянуто, як єдиний синтаксис S-виразів уможливлює гомоіконність («код як дані»). Показано, що це дає змогу створювати потужні макроси та забезпечує ефективну інтерактивну розробку в середовищі REPL.

Ключові слова: Lisp, гомоіконність, інтерпретатор, REPL, S-вирази, макроси.

Вступ. У програмуванні існують два основні підходи до виконання коду: інтерпретація та компіляція. Інтерпретатор виконує вихідний код програми рядок за рядком у реальному часі. Це забезпечує гнучкість, платформонезалежність і швидкий цикл розробки, але зазвичай призводить до нижчої продуктивності. Навпаки, компілятор заздалегідь перетворює весь код на оптимізований машинний код, що забезпечує високу швидкість виконання, але робить процес розробки повільнішим і прив'язує програму до конкретної платформи. Сучасні мови часто використовують гібридні підходи, як-от Just-In-Time (JIT) компіляція, що поєднують переваги обох моделей.

Lisp, одна з найстаріших мов програмування, є ідеальним прикладом мови, побудованої навколо інтерпретаційної філософії. Його унікальна сила випливає із синергії трьох елементів: радикально простого синтаксису, філософії гомоіконності («код як дані») та інтерактивного середовища розробки.

Основний текст. Головною особливістю Lisp є його синтаксична однорідність. І код, і дані представлені у вигляді символьних виразів (S-виразів). S-вираз – це або атом (базовий елемент, як-от число 42 або символ my-function), або список (послідовність елементів у круглих дужках) [1; 2].

Програма на Lisp – це просто S-вираз. Для виконання коду використовується єдина структура – префіксна нотація: (оператор операнд1 операнд2...). Перший елемент списку – це функція, а решта – її аргументи. Наприклад, математичний вираз $(1 + 2 * 3) - 4$ у Lisp записується як $(-(+1(* 2\ 3))4)$. Обчислення відбувається рекурсивно, «зсередини назовні». Ця синтаксична простота є ключем до найпотужнішої властивості мови [1; 2].

Гомоіконність – це властивість мови, за якої основне представлення програми є структурою даних, що належить до примітивного типу самої мови. Інакше кажучи, немає синтаксичної різниці між кодом і даними [3].

У Lisp код, записаний у вигляді S-виразів, після зчитування стає звичайною структурою даних – списком. Це дає змогу програмам на Lisp маніпулювати іншими програмами (або самими собою) за допомогою стандартних функцій для роботи зі списками.

Найважливішим наслідком гомоіконності є макроси. Це спеціальні функції, які отримують на вхід структури коду (S-вирази) і перетворюють їх на нові структури коду до моменту їх виконання. За допомогою макросів програміст може розширювати синтаксис мови, створювати нові керуючі конструкції та будувати предметно-орієнтовані мови (DSL) безпосередньо всередині Lisp.

Філософія Lisp знаходить своє практичне втілення в інтерактивному середовищі розробки – циклі «читання-обчислення-друк» (Read-Eval-Print Loop, або REPL). Це нескінченний цикл, у якому система:

1. Читає (Read) введений користувачем S-вираз і перетворює його на структуру даних (список).

2. Обчислює (Eval) цю структуру даних відповідно до правил мови.

3. Друкує (Print) результат обчислення користувачу.

REPL у Lisp – це не просто командна оболонка, а потужний інструмент для REPL-Driven Development (RDD) – інкрементальної розробки «знизу вгору» через постійний діалог із живою програмою. Коли виникає помилка, система не завершує роботу, а запускає новий REPL у контексті помилки, даючи змогу програмісту інспектувати стан програми, виправляти код «на льоту» і продовжувати виконання [4].

Висновки. Сила Lisp полягає в тісній інтеграції його ключових елементів. Простий синтаксис S-виразів є необхідною умовою для гомоіконності, яка уможливлює потужну систему макросів для розширення мови. Ця архітектура знаходить своє ідеальне вираження в інтерактивному середовищі REPL. Разом ці елементи створюють надзвичайно гнучке та потужне середовище, яке слугує не просто для виконання інструкцій, а як інструмент для мислення.

Список використаних джерел

1. Luger G. F. 11 S-expressions, The Syntax of Lisp. The University of New Mexico. URL: https://www.cs.unm.edu/~luger/ai-final2/LISP/CH%2011_S-expressions,%20The%20Syntax%20of%20Lisp.pdf (дата звернення: 19.10.2025).

2. Програмування мовою ЛІСП: лабораторна робота № 1 з курсу «Системи штучного інтелекту». Кафедра програмного забезпечення. Національний університет «Львівська політехніка». URL: https://pz.lpnu.ua/files/lisp/lisplab1.pdf (дата звернення: 19.10.2025).

3. Homoiconicity. *Wikipedia.* Last edited on 18.08.2025. URL: https://en.wikipedia.org/wiki/Homoiconicity (дата звернення: 19.10.2025).

4. Levins M. On REPL-Driven Development. *GitHub.* 18.12.2020. URL: https://mikelevins.github.io/posts/2020-12-18-repl-driven/ (дата звернення: 19.10.2025).