CSE 341 Fall 2024 – Programming Languages Assignment 1 Report

Muhammet Talha Memişoğlu-210104004009

- **INTRODUCTION**

In this assignment, I am expected to implement a program that converts C code to Lisp code while following the given rules.

- **METHODOLOGY**

The code is designed to convert a C-style source file ("input.c") into a Lisp-style syntax, which is saved in a new file ("output.lisp"). It achieves this by parsing lines of code from "input.c", identifying line types, applying conversions, and then writing the converted output to "output.lisp". This code is not designed for converting every C code, so it is not suitable for every C code. Here is a breakdown of each function group and its purpose:

Conversion Functions: These functions convert specific types of C syntax into corresponding Lisp expressions:

- **convert-print:** Converts printf statements into Lisp format expressions. It formats output based on whether it's printing an integer or a string.
- **convert-assign and convert-assigntop:** Handle assignment statements.
- **convert-if, convert-for, convert-main, convert-parant, convert-func, and convert-assign-func:** Convert various control structures and function declarations, including if statements, for loops, function definitions, and the main function.

Helper Functions: These functions are responsible for extracting specific parts of each line for easier manipulation by the conversion functions:

- **Variable Extractors**: Functions like extract-left, extract-right, left, right, func-name, left-var, and right-var isolate specific variable names or values, such as the left-hand side of an assignment or function parameters.
- **Operator and Keyword Extractors:** extract-operator and take-operator identify arithmetic or logical operators within expressions. Functions like extract-function-name and extract-text retrieve names of functions and strings in printf statements.

File I/O Functions: These manage reading from the input file and writing to the output file.

- **read-file:** Reads all lines from "input.c" and returns them as a list of strings.
- **write-file:** Writes a single converted line to "output.lisp", appending to the file.

Core Conversion Logic:

- **line-type and conversion-foo:** These two functions work together to identify the type of each line (e.g., if statement, for loop, function definition) and select the corresponding conversion function.
- **convert:** Accepts a conversion function and a line, applying the function to transform the line to Lisp syntax.
- **main:** Serves as the main execution flow. It reads “input.c”, applies conversions line-by-line, and writes the converted Lisp code to “output.lisp”.

- **CONCLUSION**

This program satisfies almost all the requirements of the assignment. The only function that violates the principles of functional programming is the **read-file** function and its usage in the main function. In addition, this program and its algorithms are not suitable for every C file, but it performs well for the given input file.