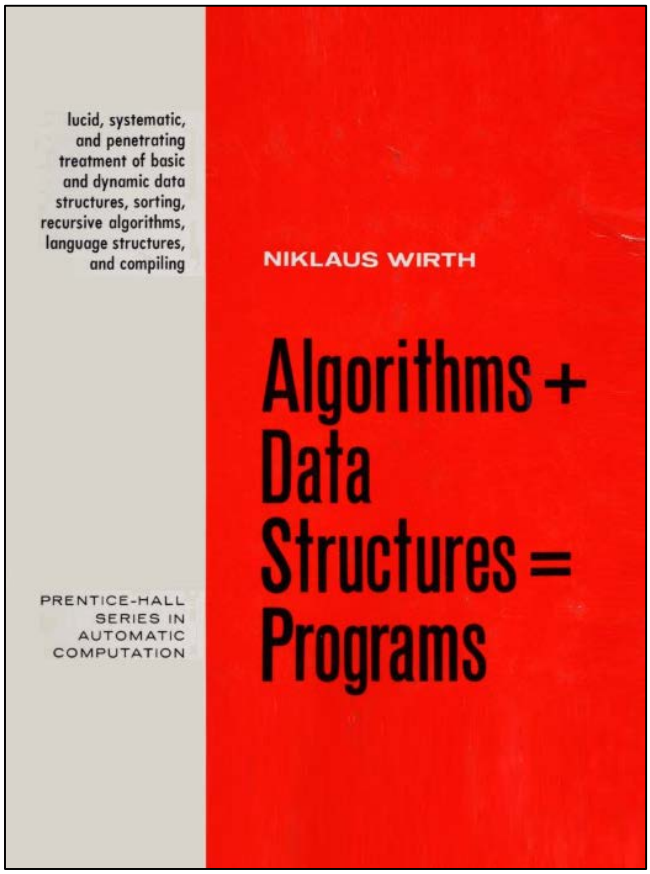
Trabajo Práctico
Intérprete de PL/0 en Clojure

El lenguaje PL/0 apareció en 1976 en el libro *Algorithms + Data Structures = Programs* de Niklaus Wirth.



Se trata de un lenguaje de tamaño reducido, ideal para estudiar o desarrollar intérpretes o compiladores que procesen los programas escritos con él.

La *asignación* es su proposición básica. Los conceptos fundamentales de la **programación estructurada** (secuencia, condición y repetición) están representados por las proposiciones *begin/end*, *if/then* y *while/do*.

PL/0 incorpora el concepto de subrutina mediante la declaración de procedimientos y las llamadas a los mismos. Esto ofrece la oportunidad de presentar el concepto de localidad de las constantes, las variables y los procedimientos.

Para reducir su complejidad, PL/0 sólo ofrece un tipo de datos: los enteros. Es posible declarar variables y constantes de este tipo.

PL/0 dispone de los operadores aritméticos y relacionales convencionales.

Para poder, en una computadora actual, ejecutar programas escritos en PL/0, en este trabajo práctico se pide desarrollar un **intérprete de PL/0** en el lenguaje **Clojure**, a fin de que pueda correr en una ***Java Virtual Machine*** contemporánea.

El lenguaje a interpretar deberá incluir, además del PL/0 original, las proposiciones *readln*, *write* y *writeln*. Estas dos últimas proposiciones deberán poder escribir cadenas literales, además de números.

A los efectos de desarrollar el intérprete solicitado, se deberá partir de los siguientes materiales proporcionados por la cátedra en el aula virtual de la materia en el campus FIUBA:

- Capítulo 5 de Wirth, N. (1976). *Algorithms + Data Structures = Programs*. Prentice-Hall, donde se presentan el lenguaje PL/0 y su sintaxis en forma de diagramas de sintaxis.
- Sintaxis del lenguaje PL/0 en BNF y en EBNF, como complemento a la anterior.
- Apunte de la cátedra: *Interpretación de programas de computadora*, donde se explican la estructura y el funcionamiento de los distintos tipos de intérpretes posibles, en particular la *compilación interpretativa*, que es la estrategia a utilizar en este trabajo práctico.
- Apunte de la cátedra: *Clojure*, donde se resume el lenguaje a utilizar para el desarrollo.
- Tutorial: *Pasos para crear un proyecto en Clojure usando Leiningen*, donde se indica cómo desarrollar un proyecto dividido en código fuente y pruebas, y su despliegue como archivo *jar*.
- 10 programas correctos escritos en PL/0, para correrlos con el intérprete.
- 10 programas erróneos escritos en PL/0, para depurarlos con el intérprete.
- Código fuente de un intérprete de PL/0 sin terminar, para completarlo. El mismo contiene una función que debe ser terminada y 19 funciones que deben desarrollarse por completo.

Con este trabajo práctico, se espera que las/los estudiantes adquieran conocimientos profundos sobre el proceso de interpretación de programas y el funcionamiento de los intérpretes de lenguajes de programación y que, a la vez, pongan en práctica los conceptos del paradigma de **Programación Funcional** vistos durante el cuatrimestre.

Para aprobar la cursada, se deberá entregar un **proyecto** compuesto por el **código fuente** del intérprete de PL/0 en Clojure y las **pruebas** de las funciones desarrolladas (como mínimo, se deberán incluir las pruebas correspondientes a los ejemplos que acompañan el código fuente del intérprete sin terminar proporcionado por la cátedra).

Al momento de rendir la evaluación final de la materia, se deberá modificar el intérprete presentado en este trabajo práctico, incorporándole alguna funcionalidad adicional.