# Informe TP°3

En esta entrega, se implementó una estructura, en nuestro caso la polaca inversa, que almacena los elementos necesarios para luego generar código assembler.

La polaca inversa la implementamos mediante una lista de strings. Dentro de la polaca vamos a encontrar:

* Variables
* Parámetros
* Nombres de funciones
* Operadores (unarios, binarios)
* Indicadores personalizados print, JUMP y BF
* Posiciones de salto hacia un índice dentro de la polaca
* Nombres de label

## Actividades

* Incorporamos al objeto SymbolTableItem, el tipo use, que hace referencia al uso que se le va a dar a dicho lexema dentro del código:
  + FUNCTION 🡪 nombre de una función simple
  + FUNCTION\_PARAM 🡪 nombre de una función con parámetros
  + VARIABLE 🡪 identificador
  + CONST 🡪 constante string
  + PARAMETER 🡪 parametro de una función
  + VARIABLE\_ASSEMBLER 🡪 variable creada para el código assembler
* Asociación de cada variable con el ámbito al que pertenece. Para cada función que creamos (incluyendo la main), se agrega un .{nombre del ámbito} al ámbito actual. Esto nos permite a la hora de encontrar una variable declarada, poder asignarle el ámbito accediendo a una variable global llamada currentScope.
* Chequeos semánticos:
  + No permitir la declaración de dos variables o funciones con el mismo nombre dentro de un mismo ámbito.
  + No permitir utilizar una variable o función que no haya sido declarada previamente.
  + No permitir invocar a una función con parámetros sin el respectivo parámetro, lo mismo en el caso inverso.
* Incorporación de la polaca inversa con una estructura de List <String>. Dependiendo de que gramática sea, se van a insertar dentro de la polaca los elementos necesarios. Para cada ámbito se crea una polaca diferente, es decir, para cada función hay una polaca diferente.
* Desarrollo de una interfaz visual, que permite ver las polacas.