Formal Languages and Compilers
Laboratory 6

1 Exercice (Translator from mini C to Pseudo Assembler)

Beginning with the scanner and the parser for the **mini C** language realised in the past labs, obtain a translator that:

- taking in input a file written in **mini C** syntax

- produces a file written in **Pseudo Assembler** syntax

The syntax of the **Pseudo Assembler** language is reported in the document with the title "Pseudo Assembler interpreter", that can be downloaded from the site of the course. Also the interpreter that, taking in input a program written in "Pseudo Assembler" executes it, can be downloaded from the site of the course.

For the translation of constructs like `if` and `while`, inherited attributes must be used in order to manage in a correct way the label names.

1.1 Input

```
/* Esempio algoritmo di ordinamento        i = 0;
Bubble sort */                             while (i < pos - 1){
                                             j = i + 1;
                                             if (x[i] > x[j]){
double x[5];                                   swap = x[j];
int i, j;                                      x[j] = x[i];
double swap;                                   x[i] = swap;
int pos;                                     }
                                             i = i + 1;
                                           }
/* Inizializzazione vettore */             pos = pos-1;
x[0] = -2.0;
x[1] = -3.0;                              }
x[2] = 3.0;
x[3] = 5.0;
x[4] = 2.5/3;                            /* Stampa risultati */
                                         i = 0;
                                         while(i<5){
/* Bubble sort */                          print x[i];
pos = 5;                                   i = i + 1;
while(pos > 0.0){                         }
```

1.2 Output

```
DOUBLE x[5]            ASS x[2]
INT i                 EVAL 5.0
INT j                 ASS x[3]
DOUBLE swap           EVAL 2.5
INT pos               ASS x[4]
EVAL -2.0             EVAL 5
ASS x[0]              ASS pos
EVAL -3.0             L1: EVAL pos 0 >  /* while (line 18) */
ASS x[1]              GOTOF L2
EVAL 3.0             EVAL 0
```

```
ASS i                                        ASS i
L3: EVAL i pos 1 -  <  /* while (line 20) */ GOTO L3
GOTOF L4                                     L4: EVAL pos 1 -
EVAL i 1 +                                   ASS pos
ASS j                                        GOTO L1
EVAL x[i] x[j] >  /* if (line 22) */         L2: EVAL 0
GOTOF L5                                      ASS i
EVAL x[j]                                     L6: EVAL i 5 <  /* while (line 35) */
ASS swap                                     GOTOF L7
EVAL x[i]                                    PRINT x[i]
ASS x[j]                                     EVAL i 1 +
EVAL swap                                    ASS i
ASS x[i]                                     GOTO L6
L5: EVAL i 1 +                               L7: END
```