
TD7 - Friday, November 13

1 Intermediary Code Generation (TP)

By now, you should have already (!!!) finished the single pass compiler (exercise in TD4) that assumed direct code generation from a small subset of Pascal (source language) to the target x86-like assembler code. In what follows, you are asked to modify your compiler such that the code generator produces an intermediate representation. The source language is the same small Pascal-like language, having only integer variables but nested functions are allowed this time.

The intermediary representation (seen in course) is :

```
MOVE reg, exp
MOVE MEM[exp], exp
JUMP label
CJUMP relop, exp, exp, label_true, label_false (evaluate expressions and compare them based on
relop, than jump to the corresponding label)
CALL f_id, exp list
label :
label_begin_taille_local : (pour les fonctions)
label_end_taille_local :
```

The local variables that "escape" will be dealt with as seen in the previous td. ("Escape analysis").