## 5.3 Code Generation Rules

| | |
|---|---|
| c | `dest <- fresh_tmp()`<br>`code.add("li dest, c")`<br>`return dest` |
| x | `# get the temporary associated to x.`<br>`reg <- symbol_table[x]`<br>`return reg` |
| $e_1$+$e_2$ | `t1 <- GenCodeExpr(e_1)`<br>`t2 <- GenCodeExpr(e_2)`<br>`dest <- fresh_tmp()`<br>`code.add("add dest, t1, t2")`<br>`return dest` |
| $e_1$-$e_2$ | `t1 <- GenCodeExpr(e_1)`<br>`t2 <- GenCodeExpr(e_2)`<br>`dest <- fresh_tmp()`<br>`code.add("sub dest, t1, t2")`<br>`return dest` |
| true | `dest <-fresh_tmp()`<br>`code.add("li dest, 1")`<br>`return dest` |
| $e_1 < e_2$ | `dest <- fresh_tmp()`<br>`t1 <- GenCodeExpr(e1)`<br>`t2 <- GenCodeExpr(e2)`<br>`endrel <- new_label()`<br>`code.add("li dest, 0")`<br>`# if t1>=t2 jump to endrel`<br>`code.add("bge endrel, t1, t2")`<br>`code.add("li dest, 1")`<br>`code.addLabel(endrel)`<br>`return dest` |

Figure 5.1: 3@ Code generation for numerical or Boolean expressions

| | |
|---|---|
| x = e | dest <- GenCodeExpr(e)<br>loc <- symbol_table[x]<br>code.add("mv *loc*, *dest*") |
| S1; S2 | # Just concatenate codes<br>GenCodeSmt(S1)<br>GenCodeSmt(S2) |
| if $b$ then $S1$ else $S2$ | lelse <- new_label()<br>lendif <- new_label()<br>t1 <- GenCodeExpr(b)<br>#if the condition is false, jump to else<br>code.add("beq *lelse*, *t1*, 0")<br>GenCodeSmt(S1) # then<br>code.add("j *lendif*")<br>code.addLabel(lelse)<br>GenCodeSmt(S2) # else<br>code.addLabel(lendif) |
| while( $b$ ){ $S$ } | ltest <- new_label()<br>lendwhile <- new_label()<br>code.addLabel(ltest)<br>t1 <- GenCodeExpr(b)<br>code.add("beq *lendwhile*, *t1*, 0")<br>GenCodeSmt(S) # execute S<br>code.add("j *ltest*") # and jump to the test<br>code.addLabel(lendwhile) # else it is done. |

Figure 5.2: 3@ Code generation for Statements

## 5.4 Allocations

EXERCISE #5 ▸ **Prepare the lab: allocations**
After code generation, we obtain the following code:

```
li temp_0, 42
li temp_1, 1
add temp_2, temp_1, temp_0
```

- Compute the naive allocation and rewrite the code accordingly.
- Compute the all-in-mem allocation and rewrite the code accordingly.