Example templates for code generation

Kim S. Larsen

February 20, 2008

Labels

- Make unique labels by appending counters.
- Generally make labels first, since code is generated recursively.

If statements

if ⟨expression⟩ **then** ⟨statement1⟩ **else** ⟨statement2⟩

```
                code for ⟨expression⟩
                cmp "⟨expression⟩-result", "true"
                jne elsepart
                code for ⟨statement1⟩
                jump endif
    elsepart:
                code for ⟨statement2⟩
    endif:
```

While statements

while ⟨expression⟩ **do** ⟨statement⟩

whilestart:
code for ⟨expression⟩
cmp "⟨expression⟩-result", "true"
jne whileend
code for ⟨statement⟩
jump whilestart
whileend:

Space allocation statements

allocate ⟨id-expression⟩ **of length** ⟨expression⟩

code for ⟨expression⟩
(code for out-of-memory check)
move "heap-counter", "address of ⟨id-expression⟩"
add the value of ⟨expression⟩ to heap-counter

Index expressions

⟨id-expression⟩ **[** ⟨expression⟩ **]**

code for ⟨expression⟩
(code for range checks)
look up address of ⟨id-expression⟩
compute final address

Addition expressions

⟨expression1⟩ + ⟨expression2⟩

code for ⟨expression1⟩
place result in temporary
code for ⟨expression2⟩
place result in temporary
add temporaries
place result in temporary

Function definitions

Code must be generated according to the stack frame convention. Make sure function labels are all produced in advance.

code for local functions
startfunc:
code for variable declarations
code for start-of-function
code for function body
endfunc:
code for end-of-function

Return statements

return ⟨expression⟩

code for ⟨expression⟩
move "⟨expression⟩-result", **%eax**
jump to label of end-of-function code