

Phase 5 : Code Generation

UNIVERSITY OF DELAWARE · COMPUTER & INFORMATION SCIENCES DEPARTMENT



Program.java

public void cgen(PrintWriter s) {
 CgenClassTable cct = new
 CgenClassTable(getClasses(), s);
}



codeGeneration.CgenClassTable.java

The following steps are performed in the constructor

try {

}

- installBasicClasses();
- installClasses(cls);
- buildInheritanceTree();
- code();
- exitScope();
- } catch (NoScopePresentException e) {
 - Utilities.fatalError(e);

UNIVERSITY OF DELAWARE · COMPUTER & INFORMATION SCIENCES DEPARTMENT



Generating code for classes

- installClasses(cls) would generate the code for all the classes by iterating through all the classes.
- code() performs all the data related functions, and the in the last line calls codeMethods()
- codeMethods() function calls the treeNodes.Method.cgen(...) function, which needs to be implemented by you.

UNIVERSITY OF DELAWARE · COMPUTER & INFORMATION SCIENCES DEPARTMENT



treeNodes.Method.cgen(...)

```
    This might look familiar to you now:
    public void cgen(java.io.PrintWriter str, codeGeneration.CgenNode cls ) {
    cls.setMethodFormals(formals);
    expr.cgen(str, cls);
    }
```

Implement all

cgen(java.io.PrintWriter str, codeGeneration.CgenNode cls) methods in treeNodes.* classes



- Look at spim.pdf if you would like to look at the instruction set.
- Compare the code generated by the binary and use diff to find places where you might be going wrong.