Project 5: Intermediate Code

CSC 4351, Spring 2016

Due: 16 April 2016

Implement the translation to intermediate code as described on p. 178 in the textbook. You only need to modify classes `Translate.Translate` and `Translate.IfThenElseExp`. Do not follow the description of 'a simpler `Translate`' in the book. Since classes `Ex`, `Nx`, and `Cx` are already provided, it's straightforward to generate good intermediate code.

Environment and Support Files

For working on this lab, change the environment variable `PROJ` in your `.profile` file to `chap7`. As usual, you can find support code in `${TIGER}/${PROJ}`.

There are some updates to the files in the `Frame`, `Mips`, and `Semant` packages. The `Tree` package contains all the tree node classes.

Unfortunately, there's a name clash for Windoze users between the classes `Tree.Exp` and `Tree.EXP`. I made a copy of the class files available in which `EXP` gets renamed to `UEXP`.

Compilation

Since we don't use any non-Java source files anymore, it's easiest to use '`javac -g *.java`' manually for compiling.

For translating an input file `test.tig` to intermediate code, execute

```
java Translate.Main test.tig
```

You can also still use `Parse.Main` and `Semant.Main`.

Submission

Since there are only two files to modify, you can either submit them individually, or you could submit the entire working directory structure:

```
cd prog5; rm */*.class; ~cs4351_bau/bin/p_copy 5
```

In the README file, provide any information that will help the grader to give you partial credit. Explain what's implemented and what's not. Explain important design decisions in your code.