C SCI 316 (Kong): TinyJ Assignment 2

Your assignment is to complete a compiler which does all of the following whenever its input is a syntactically valid TinyJ source file:

1. It checks that declarations and uses of identifiers in the source file are consistent with Java's scope rules.
2. As long as no errors are detected in the source file, it translates the source file into a sequence of instructions for a stack-based virtual machine whose instruction set is given below. At the same time, it writes to the output file an "enhanced parse tree" of the source file; this shows the static address or the stackframe offset that the compiler has allocated to each int or array reference variable, the start address of the code generated for each method, and the time at which each instruction is generated.
3. If no errors are detected in the source file then a list of the instructions generated is also written to the output file.

This assignment is to be submitted *no later than* **Friday, December 11**, but if you finish the assignment earlier you will leave yourself more time to work on the next (and last) TinyJ assignment, which you will receive before our class on Monday, December 7. [**Note**: If euclid unexpectedly goes down after 6 pm on the due date, the deadline will ***not*** be extended. Try to submit no later than noon on that day, and sooner if you can.]

The 35 virtual machine instructions that may be generated by the compiler are shown below. Their meanings are explained on pp. 3 – 4 of: **http://euclid.cs.qc.edu/316/VM-instruction-set-and-hints-for-asn-2.pdf**

Operation	**Operand 1**	**Operand 2**
STOP		
PUSHNUM	<integer>	
PUSHSTATADDR	<address of static variable>	
PUSHLOCADDR	<local variable or parameter's stackframe offset>	
LOADFROMADDR		
SAVETOADDR		
WRITELNOP		
WRITEINT		
WRITESTRING	<address of first char>	<address of last char>
READINT		
CHANGESIGN		
NOT		
ADD		
SUB		
MUL		
DIV		
MOD		
AND		
OR		
EQ		
LT		
GT		
NE		
GE		
LE		
JUMP	<address of target instruction>	
JUMPONFALSE	<address of target instruction>	
CALLSTATMETHOD	<address of method's first instruction>	
INITSTKFRM	<no. of locations needed for method's local vars>	
RETURN	<no. of parameters the method has>	
HEAPALLOC		
ADDTOPTR		
PASSPARAM		
DISCARDVALUE		
NOP		

How to Install the Already-Written Parts of the Program (Assuming You Have Already Followed the Installation Instructions Provided with TinyJ Assignment 1)

1. ***Login to euclid*** **and enter these 3 commands (notice the uppercase `M` in `virtualMachine`):**

```
jar xvf TJasn.jar
javac -cp .  TJasn/TJ.java
javac -cp .  TJasn/virtualMachine/*.java
```

2. ***Logout from euclid.*** **Then *login to venus* and repeat step 1 on venus.**

Also do the next 3 steps **if** you did assignment 1 on your PC and plan to do this assignment on your PC too:

3. **In a cmd.exe (command prompt) window on the PC, make `c:\316java` your working directory by entering the following: `cd /d c:\316java`**
4. **Either use an scp or sftp client to download the file `TJasn.jar` from your home directory on *euclid* or *venus* to the c:\316java folder on your PC, or e-mail that file to yourself and save it in your c:\316java folder. [If you cannot remember how to do this, see installation step 9 on p. 3 of the Assignment 1 document, but substitute the filename `TJasn.jar` for `TJ1asn.jar` when you follow those instructions.]**
5. **On your PC, repeat step 1 in the cmd.exe window (with c:\316java as your working directory).**

Then, ***regardless of whether or not you did steps* 3, 4 *and* 5**, do the following:

6. Starting on **Wednesday, December 2**, ***either*** bring printouts of the following 18 files to class ***or*** bring a laptop / tablet computer to class and be ready to *quickly* bring any of these files up on your screen: **`ParserAndTranslator.java.txt` (from the TJasn directory on venus, euclid or your PC). The 17 other `.java.txt` files in the TJasn directory and its symbolTable and virtualMachine subdirectories; there are 3 `.java.txt` files in TJasn, 9 in symbolTable, and 6 in virtualMachine.**

How to Do This Assignment

The only file you need to change is `TJasn/ParserAndTranslator.java`. In this file, each /* ???????? */ comment must be replaced with appropriate code. *For most of these comments* (*specifically, the ones on lines 511 − 723*)*, a good way to begin is to* ***first copy over*** *code you wrote for the corresponding part of Parser.java in the previous assignment.* [**Note:** For the comment on line 511, you should only have to copy 3 statements! Do ***not*** copy over the lines of TJ1asn/Parser.java that correspond to lines 482–3, 500, 502–4, 507, and 513 in TJasn/ParserAndTranslator.java, and also do ***not*** copy the lines corresponding to lines 515, 519, and 537–8 in that file.] ***Then*** *make changes, so appropriate virtual machine instructions are generated.**

To recompile `TJasn/ParserAndTranslator.java` after you have edited it, enter: **`javac -cp . TJasn/ParserAndTranslator.java`** [This works on euclid or venus if your working directory is your home directory, and in a cmd.exe window on your PC (after step 5) if your working directory is c:\316java.]

*`ParserAndTranslator.java` might be a **Windows** text file: The line-feed character at the end of each line might be preceded by a Ctrl-M (carriage return). If so, and you plan to edit the file using emacs on venus or using vi, vim, or emacs on euclid, then you should first eliminate these Ctrl-M characters. This can be done by entering the following commands:

On **euclid**: `/users/kong300/convert`
On **venus**: `/home/faculty/ykong/convert`

To have a good chance of finishing by Friday 12/11, I recommend you ***keep to the following schedule***:

- For the /* ???????? */ comments on lines 511 − 723, copy over code you wrote for the corresponding part of TinyJ assignment 1 ***no later than*** Wednesday 12/2.
- Fill in the /* ???????? */ gaps that were on lines 638 – 682 ***before*** Monday 12/7.
- You will then have a few days in which to fill in the other /* ???????? */ gaps—i.e., the gaps that were on lines 492, 495, 511, 549, 593, 610, 627, and 723. Some **hints** are given on the last 3 pages of: **`http://euclid.cs.qc.edu/316/VM-instruction-set-and-hints-for-asn-2.pdf`**

How to Test Your Solution

You can execute ***your completed program*** as follows:

`java -cp . TJasn.TJ` ***`TinyJ-source-file-name output-file-name`***

For example, **`java -cp . TJasn.TJ CS316ex12.java 12.out`**

[Your working directory should be your home directory on venus or euclid, and `c:\316java` on a PC.]

When it asks you
```
Want debugging stop or post-execution dump? (y/n)
```
you should enter **`y`**

When it then asks
```
Enter MINIMUM no. of instructions to execute before debugging stop.
(Enter -1 to get a post-excution dump but no debugging stop.):
```
you should enter **`0`**

When it then asks
```
Stop after executing what instruction? (e.g., PUSHNUM)
(Enter * to stop after executing just 0 instructions.):
```
you should enter **`*`**

You can execute ***my solution*** to Assignments 2 and 3 as follows:

`java -cp TJsolclasses:. TJasn.TJ CS316ex`*`k`*`.java `*`k`*`.sol [venus or euclid]`

`java -cp TJsolclasses;. TJasn.TJ CS316ex`*`k`*`.java `*`k`*`.sol [PC, working dir=c:\316java]`

Here **`CS316ex`*`k`*`.java`** is the TinyJ source file and ***`k`*`.sol`** is the output file.

Test your solution by comparing the output file it produces to the output file produced by my solution, at least for the 16 `CS316ex`*k*`.java` files — use **`diff -c`** on venus and euclid, or **`fc /n`** on your PC. *When the input file is a correct TinyJ program, the output files produced by your and my solutions* ***should contain the same* "Instructions Generated:" *lists* (*near the end, just above the* "Data memory dump")**.

How to Submit Your Solution*

This assignment counts 2% towards your grade. To submit:

1. **Add a comment at the beginning of `ParserAndTranslator.java` that states your name and the names of the students you worked with (if any). As before, you may work with up to two other students.**
2. **Leave your final version of `ParserAndTranslator.java` in your `TJasn` directory on euclid, so it replaces the original version of that file, before midnight* on the due date. When two or three students work together, *each* of the students must leave the completed file in his/her directory. (If you are working on venus or your own PC, you can transfer the file `ParserAndTranslator.java` to euclid by following the instructions on p. 5 of the TinyJ Assignment 1 document; but you should substitute `TJasn/ParserAndTranslator.java` for `TJ1asn/Parser.java` and substitute `TJasn` for `TJ1asn` when following the instructions.)**

Be sure to test your submission on euclid. If your modified version of `ParserAndTranslator.java` cannot even be compiled without error on *euclid*, then you will receive no credit for your submission. **Do NOT open your submitted file `ParserAndTranslator.java` in an editor on euclid after the due date, unless you are resubmitting a corrected version of your solution as a *late* submission.**

*If euclid unexpectedly goes down after 6 p.m. on the due date, the deadline will ***not*** be extended. I suggest you try to submit no later than ***noon*** on that day, and sooner if you can.