CS4215 Programming Language Implementation

Lab task for Week 11
A Virtual Machine for cPL

1. Download the file `http://www.comp.nus.edu.sg/~cs4215/labtasks/week11.zip`, and extract it to the Eclipse workspace folder. The workspace folder should now contain a file `cs4215_week_11`.

2. In Eclipse, go to "File", "New", "Project", "Java Project", "Next", and choose "`cs4215_week_11`" as "Project name". Press "Finish".

3. Use the "Run Configurations" to run `cPLcompiler.cplc` with a file name (for example `test.cmpl`) as "Program argument". The file should contain the instruction to print an integer, say 123:

   ```
   print 123
   ```

   The compiler should reply:

   ```
   cvml code written to test.cvml
   ```

   Now, you can interpret the compiled program using the virtual machine by running `cPLvm.impl` with the base name of the file you just compiled (in the example `test`, resulting in `123`).

   Note that the given virtual machine in `VM.java` cannot handle exceptions, threads and wait/signal. Also note that in cPL, every thread, including the "parent" thread in which the entire program starts, ignores the result of evaluation of its body expression. Therefore, you need to write "`print 123`" to see the result in the program above.

   It is your task to complete the virtual machine by covering the entire instruction set given in `cVML`.

   This includes the following:

   - Built-in exceptions: Note that the compiler compiles the two builtin exceptions for division by zero and invalid record property access, respectively. It then saves the addresses of the two builtin exceptions in the `.cvml` file. The virtual machine loads these addresses and passes them to the `run` method of `cPLvm.VM` as arguments

     ```
     int divisionByZeroAddress,
     int invalidRecordAccessAddress
     ```

Using this knowledge, you should be able to improve the implementation of the machine instructions `DIV` and `DOT` such that the right actions are taken in all cases.

- Throwing and catching exceptions: Implement the machine instructions `TRY` and `THROW` as described in the notes. *Exceptions that are not caught in the thread in which they were thrown, should lead to silent termination of the thread without any error message.*
- Optimization of `ENDTRY`: Often, `RTN` statements immediately follow `ENDTRY` instructions. These `ENDTRY` instructions are omitted by the cPL compiler. As a result, `RTN` needs to be prepared to encounter unnecessary `catch` frames on the runtime stack.
- Threads: Implement the machine instructions `STARTTHREAD` and `ENDTHREAD` as described in the notes.
- Wait and signal: Implement the machine instructions `WAIT` and `SIGNAL` as implemented in the notes.

4. Submit the resulting file

    - `VM.java`

    from your folder `imPLvm` in the IVLE workbook "Week 11".

    Make sure that you do not change any other files when you test your programs.

    Suggestion: When you are done with the solution, save your four files in a secure place. Then download a fresh copy of the lab task, and place your three files into that copy. Then re-do your tests.