CS4212 – Compiler Design
School of Computing
National University of Singapore

September 6, 2012

Problem Set 2

Semester 1, 2012/13

Due: ***September 23, 23:59*** **Marks: 10**

Submission: In IVLE, in the cs4212 workbin, you will find a folder called `“Homework submissions”.` In that folder, there are currently ***5 subfolders:*** **`PS2P01,…,PS2P04`**. The last two digits of the folder name indicate the solution that is to be submitted into that folder: the solution to *Question 1* into **`PS2P01`**, and so on (that is, you need to submit 5 separate solutions to 5 questions). A solution should consist of a ***single text file*** that can be loaded into the SWI-Prolog environment and executed. You should provide as much supplementary information about your solution as you can, ***in the form of Prolog comments***.

You are allowed to work in teams of 2 on each problem set. Each team should make a single submission for each problem. Please indicate clearly who the authors of the submission are, in a comment at the beginning of the file. Though not expressly required, it is recommended that you change teammates for subsequent problem sets, so you get to make more friends and interact with more people.

Problem 1 [2 mark, submit to PS2P01]

Implement the `+=` , `-=` , `*=` , `/=` , `++`, `--` operators of C in the language of `comp_expr_naive_4.pro` . Thus, the following toy program should become legal in your version of the compiler:

```
a = 240 ; b = 144 ; i = 0 ;
while ( i < b ) do {
    if ( b > a )
        then { b += a }
        else { a -= b } ;
    i ++ ;
}
```

Hint: you may implement these operators as syntactic sugar. For instance, the expression `x += E` can be translated on the fly into `x = x + E`, which can be recursively passed to the compiler.

Problem 2 **[2 mark, submit to PS2P02]**

Extend the language of the toy compiler `comp_expr_naive_4.pro` with the conditional operator `?:` . The semantics of the operator must be similar to the equivalent operator in C. That is, if `E1`, `E2`, and `E3` are legal expressions, then so is `E1 ? E2 : E3`. Moreover, if `E1` evaluates to a non-zero value, then the value of the entire expression is the value of `E2`. Otherwise, it is the value of `E3`.

Problem 3 **[3 mark, submit to PS2P03]**

Implement a `goto` statement for the toy language of Lecture 4. The format should be similar to the one of the language C. For defining labels, use the '`::`' (double colon) instead of '`:`' (simple colon). The attempt to use the simple colon will cause an operator precedence conflict in Prolog. Thus

```
label1 ::
label2 :: x = 3 ;
```

is a properly labelled statement (we allow multiple labels on the same statement). To make this work, you will need the operator declarations:

```
:- op(950,fx,goto).
:- op(970,xfy,::).
:- op(969,xf,::).
```

added to your compiler code.

Problem 4 **[2 mark, submit to PS2P04]**

Implement in the language of the toy compiler given in the Prolog file `comp_expr_naive_4.pro`, the simultaneous assignment operator available in Python. For instance, the statement

```
(x,y,z) = (y,z,x)
```

will "rotate" the values of `x`, `y, and z` by simultaneously assigning the tuple components of the right hand side of the assignment to the corresponding tuple components of the left hand side. The implementation can be achieved by "inventing" temporary storage/variables for each of the tuple components. This statement can be implemented either as syntactic sugar, or directly as a language feature.