Programming Languages (COP 4020/6021) [Fall 2016]
Assignment VIII

Objectives

1. To demonstrate an understanding of evaluation contexts.
2. To implement an interpreter for diML+ops based on evaluation contexts.

Due Date: Sunday, April 24, 2016, at 11:59pm.

Machine Details: Complete this assignment by yourself on the following CSEE network computers: c4lab01, c4lab02, ..., c4lab20. These machines are physically located in the Center 4 lab (ENB 220). Do not use any server machines like grad, babbage, sunblast, etc. You can connect to the C4 machines from home using SSH. You are responsible for ensuring that your programs compile and execute properly on these machines.

Assignment Description

First, correct any problems with your implementation of the `sub` function from Assignment IV. Then, in a directory containing a copy of *as4.sml*, begin a new file called *as8.sml* with the command `use "as4.sml";`. Then implement the following in *as8.sml*.

(1) Exception and datatype declarations for diML+ops evaluation contexts:

```
exception Stuck;
datatype ctxt = Hole | IfCtxt of ctxt*expr*expr | OpCtxt1 of ctxt*binop*expr
  | OpCtxt2 of int*binop*ctxt | OpCtxt3 of bool*binop*ctxt
  | ApplyCtxt1 of ctxt*expr | ApplyCtxt2 of string*typ*typ*string*expr*ctxt
```

You can just copy-paste this code into your *as8.sml* file.

(2) `isVal : expr -> bool`
This function takes a diML+ops expression *e* and returns true iff *e* is a value.

(3) `fill : ctxt -> expr -> expr`
This function takes a diML+ops evaluation context *E* and expression *e* and returns *E[e]*.

(4) `decompose : expr -> ctxt * expr`
This function takes an expression *e* and returns an *(E,e')* such that *e=E[e']* and *e'* can take a beta step. If no such *(E,e')* exists, then this function raises the *Stuck* exception.

(5) `beta : expr -> expr`
This function returns the result of β-stepping its argument; if no β-step is possible, *Stuck* is raised.

(6) `smallStep : expr -> expr`
This function returns the result of stepping its argument; if no step is possible, *Stuck* is raised. This small-step operation must be defined in terms of evaluation contexts (as discussed in class).

(7) `bigStep : expr -> expr`
This function returns the value resulting from fully evaluating its argument expression *e*. If *e* gets stuck, this function raises *Stuck*, and if *e* diverges, so does this function. This big-step operation must be defined in terms of small-step operations.

Notes: Assume that all variable names have been chosen to avoid capture; hence, `sub` from Assignment IV can be used to perform substitutions. Also, every one of the 6 functions (numbered (2)-(7)) should be legitimately invoked somewhere in `smallStep` or `bigStep`. My *as8.sml* is 52 lines of code and took about 1.5 hours to implement and test.

Sample Executions

```
- use "as8.sml";
...
- Control.Print.printDepth:=20;
...
-  val  mult  =  FunExpr("f",Int,Arrow(Int,Int),"m",FunExpr("g",Int,Int,"n",OpExpr(VarExpr
"m",Times,VarExpr "n")));
...
- val pe = (OpExpr(IntExpr 4,Equal,ApplyExpr(ApplyExpr(mult,IntExpr 2),IntExpr 2)));
...
- decompose pe;
val it =
  (OpCtxt2 (4,Equal,ApplyCtxt1 (Hole,IntExpr 2)),
   ApplyExpr
     (FunExpr
        ("f",Int,Arrow (Int,Int),"m",
         FunExpr ("g",Int,Int,"n",OpExpr (VarExpr "m",Times,VarExpr "n"))),
      IntExpr 2)) : ctxt * expr
- pe = (fill (#1 it) (#2 it));
val it = true : bool
- bigStep pe;
val it = TrueExpr : expr
- smallStep pe;
val it =
  OpExpr
    (IntExpr 4,Equal,
     ApplyExpr
       (FunExpr ("g",Int,Int,"n",OpExpr (IntExpr 2,Times,VarExpr "n")),
        IntExpr 2)) : expr
- smallStep it;
val it = OpExpr (IntExpr 4,Equal,OpExpr (IntExpr 2,Times,IntExpr 2)) : expr
- smallStep it;
val it = OpExpr (IntExpr 4,Equal,IntExpr 4) : expr
- smallStep it;
val it = TrueExpr : expr
- smallStep it;

uncaught exception Stuck
  raised at: as8.sml:38.25-38.30
```

Grading

For full credit, your implementation must obey the same bullet points stated for Assignment VI grading. That is, your code must be commented and formatted appropriately, use ML features like pattern matching—including using or-patterns—when appropriate, etc. As always, we'll test submissions on inputs not shown in the sample executions above.

Submission Notes

- Type the following pledge as an initial comment in your *as8.sml* file: "I pledge my Honor that I have not cheated, and will not cheat, on this assignment." Type your name after the pledge. Not including this pledge will lower your grade 50%.
- Upload and submit your *as8.sml* file in Canvas.
- You may submit your assignment in Canvas as many times as you like; we will grade your latest submission.
- For every day that your assignment is late (up to 3 days), your grade reduces 10%.