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

Assignment VI

Objectives

1. To gain experience programming with recursively defined data types in ML.
2. To demonstrate an understanding of diML+ops static semantics by implementing a type checker.
3. To demonstrate an understanding of diML+ops dynamic semantics by implementing an interpreter.

Due Date: Sunday, March 20, 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 *as6.sml* with the command `use "as4.sml";`. Then implement the following values in *as6.sml*.

(1) `tc : expr -> typ option`

This function takes a diML+ops expression *e* and returns NONE iff *e* is an ill-typed program and SOME *t* iff *e* is a well-typed program having type *t*. Note: For e_1=e_2 to be well typed in diML+ops, e_1 and e_2 have to have the same type, which can't be a function type.

(2) `exception Stuck`

Section 5.2 of the *Elements of ML Programming* textbook has details on using exceptions in ML.

(3) `eval : expr -> expr`

This function takes a diML+ops expression *e* and evaluates *e* for as many steps as possible. If evaluation of *e* converges to a value *v*, then *eval(e)* returns *v*; if *e* diverges then so does *eval(e)*. Function *eval* must raise exception *Stuck* at any point that evaluation gets "stuck" without a value being produced (but note that because diML+ops is type safe, only ill-typed expressions can get stuck before becoming values). Evaluation in diML+ops is left-to-right and call-by-value, with short-circuit conjunction and disjunction operators.

Throughout this assignment, you may assume that all variable names in expressions being type checked and evaluated are unique, so you never have to alpha-convert expressions.

Hints: My *as6.sml* is 67 lines of code (not counting comments and whitespace) and took about 1.5 hours to implement and test.

SML has useful "or-patterns", which are described at http://smlnj.org/doc/features.html.

Sample Executions

```
- use "as6.sml";
...
- val e1 = ApplyExpr(IntExpr 1, IntExpr 2);
val e1 = ApplyExpr (IntExpr 1,IntExpr 2) : expr
- (* define factorial *)
- val e2 = FunExpr("f",Int,Int,"x",IfExpr(OpExpr(VarExpr "x",Less,IntExpr 2),IntExpr
1,OpExpr(VarExpr "x",Times,ApplyExpr(VarExpr("f"),OpExpr(VarExpr "x",Minus,IntExpr
1)))));
val e2 = FunExpr ("f",Int,Int,"x",IfExpr (OpExpr #,IntExpr #,OpExpr #)) : expr
- tc e1;
val it = NONE : typ option
- eval e1;

uncaught exception Stuck
  raised at: as6.sml:60.46-60.51
- tc e2;
val it = SOME (Arrow (Int,Int)) : typ option
- (eval e2)=e2;
val it = true : bool
- eval (ApplyExpr(e2,IntExpr 5));
val it = IntExpr 120 : expr
- eval (OpExpr(TrueExpr,Plus,ApplyExpr(FunExpr("f",Bool,Int,"x",ApplyExpr(VarExpr "f",
VarExpr "x")),TrueExpr)));

uncaught exception Stuck
  raised at: as6.sml:76.22-76.27
```

Grading

For full credit, your implementation must:

- be commented and formatted appropriately (as on previous assignments).
- use ML features like pattern matching—including using or-patterns—when appropriate.
- not define any extra top-level values.
- compile and run on the C4 machines with no errors or warnings (except for *Stuck* exceptions raised at appropriate times).
- not use any library functions.
- not use any ML features that cause *side effects* to occur (e.g., I/O or references/pointers).
- not be significantly more complicated than is necessary.

Please note that we will test submissions on inputs not shown in the sample executions above.

Submission Notes

- Type the following pledge as an initial comment in your *as6.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 *as6.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%.