
CS 421 --- Interpreter Activity 2

Manager	Keeps team on track	
Recorder	Records decisions / QC	
Reporter	Reports to class	
Reflector	Assesses team performance	

(Hand in one copy per team. Do NOT mangle or staple the corners; it will jam the scanner!)

Here is part of the code for the i4.hs interpreter.

```
0 data Val = IntVal Integer
1   deriving (Show,Eq)
2
3 data Exp = IntExp Integer
4         | IntOpExp String Exp Exp
5         | VarExp String
6         | LetExp String Val Exp
7   deriving (Show,Eq)
8
9 type Env = [(String,Val)]
10
11 intOps = [ ("+",(+))
12          , ("-",(-))
13          , ("*",(*))
14          , ("/",div)]
15
16 liftIntOp f (IntVal i1) (IntVal i2) = IntVal (f i1 i2)
17 liftIntOp f _ _ = IntVal 0
18
19 eval :: Exp -> Env -> Val
20 eval (IntExp i) _ = IntVal i
21
22 eval (IntOpExp op e1 e2) env =
23   let v1 = eval e1 env
24       v2 = eval e2 env
25       Just f = lookup op intOps
26   in liftIntOp f v1 v2
27
28 eval (VarExp v) env =
29   case lookup v env of
30     Just vv -> v
31     Nothing -> IntVal 0
32
33 eval (LetExp var e1 e2) env =
34   let v1 = eval e1 env
35   in eval e2 (var,v1):env
```

Problem 1) Code review this. Three lines have errors, and they are different ones than from last time! Find them and correct them.

Problem 2) Consider the following code:

```
0 Prelude> delta = 1
1 Prelude> inc x = x + delta
2 Prelude> inc 10
3 11
4 Prelude> messup x = let delta = 2 in delta + inc x
5 Prelude> messup 10
6 -- What happens?
```

Have each member of the team predict the output of `messup 10`. Come to a consensus about the output and why.

Adding functions to the Interpreter

Consider the following code that would add functions to the language. For the data types we are only showing the added clauses to save space.

```
0 data Val = FunVal String Exp
1
2 data Exp = FunExp String Exp
3         | FunApp Exp Exp
4
5 eval (FunExp var body) env = FunVal var body
6 eval (FunApp e1 e2) env =
7   let FunVal var body = eval e1 env
8       arg              = eval e2 env
9   in eval body ((var,arg):env)
10
11 example = eval (FunExp "x" (IntOpExp "+" (VarExp "x") (VarExp "delta"))) [("delta", IntVal 1)]
```

Problem 3) The constructor for `FunVal` takes an `Exp` for the body, not a `Val`. Why do we not evaluate the body of the function when we create the `FunVal`?

Problem 4) What does the syntax `((var, arg):env)` indicate?

Problem 5) Here is a reasonably equivalent program to the Haskell one above.

```
0 Prelude> eval (LetExp "delta" (IntExp 1)
1             (LetExp "inc" (FunVal "x" (IntOpExp "+" (VarExp "x") (VarExp "delta"))))
2             (LetExp "messup" (FunVal "x" (LetExp "delta" (IntExp 2)
3                 (IntOpExp "+" (VarExp "delta") (FunApp (VarExp "inc") (VarExp "x"))))))
4             (FunApp (VarExp "messup") (IntExp 10))))
5             [ ]
```

Does it give the same result as the Haskell program? Hint: the answer is ``no.'' What goes wrong?

Closures

Problem 6) The instructor will talk briefly about *Closures*, which fix this problem. Modify the given code to implement closures.

Interpreter Activity 2--- Reflector's Report

Manager	Keeps team on track	
Recorder	Records decisions	
Reporter	Reports to Class	
Reflector	Assesses team performance	

1. What was a strength of your team's performance for this activity?

2. What could you do next time to increase your team's performance?

3. What insights did you have about the activity or your team's interaction today?

Interpreter Activity 2 --- Team's Assessment (SII)

Manager or Reflector: Consider the objectives of this activity and your team's experience with it, and then answer the following questions after consulting with your team.

1. What was a **strength** of this activity? List one aspect that helped it achieve its purpose.

2. What is one things we could do to **improve** this activity to make it more effective?

3. What **insights** did you have about the activity, either the content or at the meta level?