Homework 8

April 8, 2020

1 Write a parser for a toy language

In this homework, you will use Parsec library to implement a parser for a toy language with the following ML-like syntax in EBNF grammar.

```
 <Decl> ::= { (<FunDecl> | <ValDecl>) }

 <FunDecl> ::= 'fun' <Ident> <Ident> '=' <Exp>

 <ValDecl> ::= 'val' <Ident> '=' <Exp>

 <Expr> ::= <Comp>
         | 'if' <Expr> 'then' <Expr> 'else' <Expr>
         | 'let' <Decl> { <Decl> } 'in' <Expr> 'end'
         | 'fn' <Ident> '=>' <Expr>

 <Comp> ::= <Plus> { ('>' | '=' | '<') <Plus> }

 <Plus> ::= <Mult> { ('+' | '-') <Mult> }

 <Mult> ::= <App>  { ('*' | '/') <App> }

 <App>  ::= <Fact> { <Fact> }

 <Fact> ::= '(' <Expr> ')'
         |  <Integer>
         |  <Identifier>
```

The parser should take a string and return a list of declarations using the following data types that define the abstract syntax.

```
-- function/variable declaration
data Decl = Fun String String Exp -- fun f x = e;
          | Val String Exp        -- val x = e;
```

```
-- expressions
data Exp = Lt Exp Exp     -- e1 < e2
         | Gt Exp Exp     -- e1 > e2
         | Eq Exp Exp     -- e1 = e2
         | Plus Exp Exp   -- e1 + e2
         | Minus Exp Exp  -- e1 - e2
         | Times Exp Exp  -- e1 * e2
         | Div Exp Exp    -- e1 div e2
         | Var String     -- x
         | If Exp Exp Exp -- if e0 then e1 else e2
         | Fn String Exp  -- fn x => e
         | Let [Decl] Exp -- let val x = e0; fun f = e1; in e2 end
         | App Exp Exp    -- e1 e2
         | Const Integer  -- n
```

The top-level parser has the type `Paser [Decl]`, which is a parser that returns a list of `Decl` values.

2 Requirement

Use the template file to complete the parser.

3 Testing

You can test the implementation using the following `main`

```
import AST
import Parser

main :: IO ()
main = do
         let fact = "fun fact x = if x < 1 then 1 else x * fact (x-1)\n"
         let   it = "val it = let val y = 10 in fact y end"

         let d = (fact ++ it)
         run prog d
```

The output looks like the following (line breaks added for clarity):

```
[
 fun fact x = if (x < 1) then 1 else (x * (fact (x - 1))),
 val it = (let [val y = 10] in (fact y) end)
]
```

4 Submission

Please write your solution in a file – `hwk8.hs` and submit it to the dropbox.