Kirsten Pevidal 23560808

CS316 Class Names

* LexAnalyzer (Lexical Analyzer)
* Parser
* Val
  + IntVal
  + FloatVal
  + BoolVal
  + NilVal
  + PairVal
  + IdVal
* FunDefList (NOT NEEDED)
  + FunDefList(Fun Def) // <FunDefList> -> <FunDef>
  + FunDefList(FunDef d, FunDefList l) // <FunDefList> -> <FunDef><FunDefList>
* FunDef (NOT NEEDED)
  + FunDef(Header h, Exp e) // <FunDef> -> <Header><Exp>
* Header (NOT NEEDED)
  + Header(FunctionName fn, ParameterList pl) // <Header> -> <FunName><ParameterList>
* FunName (NOT NEEDED)
  + FunName(String ID) // <FunName> -> <Id>
* ParameterList (NOT NEEDED)
  + ParameterList(String ID, ParameterList) // <ParameterList> -> <ID><ParameterList>
* Exp
  + Exp\_ID(String ID) // <Exp> -> <Id>
  + Exp\_Int(String Int) // <Exp> -> <Int>
  + Exp\_Float(String Float) // <Exp> -> <Float>|<FloatE>|<FloatF>
  + Exp\_Nil(String Nil) // <Exp> -> <Nil>
  + Exp\_If(Exp e1, Exp e2, Exp e3) // <Exp> -> <If> <Exp> <Then> <Exp><Else><Exp>
  + FunExp(FunOp fo, ExpList) // <Exp> -> ‘(‘ <FunExp> ‘)’
* FunExp
  + FunExp(FunOp fo, ExpList xl) // <FunExp> -> <FunOp><ExpList>
* FunOp
  + FunOp\_Id(String Id) // <FunOP> -> <Id>
  + FunOp\_Pair(String Pair) // <FunOp> -> <Pair>
  + FunOp\_First(String First) // <FunOp> -> <First>
  + FunOp\_Second(String Second) // <FunOp> -> <Second>
  + ArithOp(String val) // <FunOp> -> <ArithOp>
  + BoolOp(String val) // <FunOp> -> <BoolOp>
  + CompOp(String val) // <FunOp> -> <CompOp>
* ArithOp
  + Arith\_Plus(String Plus) // <ArithOp> -> <Plus>
  + Arith\_Sub(String Sub) // <ArithOp> -> <Sub>
  + Arith\_Mul(String Mul) // <ArithOp> -> <Mul>
  + Arith\_Div(String Div) // <ArithOp> -> <Div>
* BoolOp
  + Bool\_And(String And) // <BoolOp> -> <And>
  + Bool\_Or(String Or) // <BoolOp> -> <Or>
  + Bool\_Not(String Not) // <BoolOp> -> <Not>
* CompOp
  + Comp\_Eq(String Eq) // <CompOp> -> <Eq>
  + Comp\_Gt(String Gt) // <CompOp> -> <Gt>
  + Comp\_Ge(String Ge) // <CompOp> -> <Ge>
  + Comp\_Lt(String Lt) // <CompOp> -> <Lt>
  + Comp\_Le(String Le) // <CompOp> -> <Le>
* ExpList
  + ExpList(Exp e) // <ExpList> -> <Exp>
  + ExpList(Exp e, ExpList xl) // <ExpList> -> <Exp><ExpList>