**Printing Prefix Infix Postfix**

Recall [tree.scala](http://lara.epfl.ch/w/cc09:tree.scala) for the while language

* it contained pretty-printer for infix expressions

Pretty printer for prefix, infix, and postfix expressions:

[sealed](http://scala-lang.org) [abstract](http://scala-lang.org) [class](http://scala-lang.org) Expr

[case](http://scala-lang.org) [class](http://scala-lang.org) [Var](http://scala-lang.org)(varID: String) [extends](http://scala-lang.org) Expr

[case](http://scala-lang.org) [class](http://scala-lang.org) IntLiteral(value: Int) [extends](http://scala-lang.org) Expr

[case](http://scala-lang.org) [class](http://scala-lang.org) Plus(lhs: Expr, rhs: Expr) [extends](http://scala-lang.org) Expr

[case](http://scala-lang.org) [class](http://scala-lang.org) Times(lhs: Expr, rhs: Expr) [extends](http://scala-lang.org) Expr

[sealed](http://scala-lang.org) [abstract](http://scala-lang.org) [class](http://scala-lang.org) Token

[case](http://scala-lang.org) [class](http://scala-lang.org) ID(str : String) [extends](http://scala-lang.org) Token

[case](http://scala-lang.org) [class](http://scala-lang.org) Const(c : Int) [extends](http://scala-lang.org) Token

[case](http://scala-lang.org) [class](http://scala-lang.org) Add [extends](http://scala-lang.org) Token

[case](http://scala-lang.org) [class](http://scala-lang.org) Mul [extends](http://scala-lang.org) Token

[object](http://scala-lang.org) Printer {

[def](http://scala-lang.org) prefix(e : Expr) : List[Token] = e [match](http://scala-lang.org) {

[case](http://scala-lang.org) [Var](http://scala-lang.org)(id) => List(ID(id))

[case](http://scala-lang.org) IntLiteral(c) => List(Const(c))

[case](http://scala-lang.org) Plus(lhs,rhs) => List(Add()) ::: prefix(lhs) ::: prefix(rhs)

[case](http://scala-lang.org) Times(lhs,rhs) => List(Mul()) ::: prefix(lhs) ::: prefix(rhs)

}

[def](http://scala-lang.org) infix(e : Expr) : List[Token] = e [match](http://scala-lang.org) { // should emit parantheses!

[case](http://scala-lang.org) [Var](http://scala-lang.org)(id) => List(ID(id))

[case](http://scala-lang.org) IntLiteral(c) => List(Const(c))

[case](http://scala-lang.org) Plus(lhs,rhs) => infix(lhs) ::: List(Add()) ::: infix(rhs)

[case](http://scala-lang.org) Times(lhs,rhs) => infix(lhs) ::: List(Mul()) ::: infix(rhs)

}

[def](http://scala-lang.org) postfix(e : Expr) : List[Token] = e [match](http://scala-lang.org) {

[case](http://scala-lang.org) [Var](http://scala-lang.org)(id) => List(ID(id))

[case](http://scala-lang.org) IntLiteral(c) => List(Const(c))

[case](http://scala-lang.org) Plus(lhs,rhs) => postfix(lhs) ::: postfix(rhs) ::: List(Add())

[case](http://scala-lang.org) Times(lhs,rhs) => postfix(lhs) ::: postfix(rhs) ::: List(Mul())

}

}

[object](http://scala-lang.org) Test {

[def](http://scala-lang.org) main(args : Array[String]) = {

[val](http://scala-lang.org) t = Plus([Var](http://scala-lang.org)("x"),Times([Var](http://scala-lang.org)("y"),[Var](http://scala-lang.org)("z")))

println("prefix: " + Printer.prefix(t))

println("infix: " + Printer.infix(t))

println("postfix: " + Printer.postfix(t))

}

}

Output:

$ scala Test

prefix: List(Add(), ID(x), Mul(), ID(y), ID(z))

infix: List(ID(x), Add(), ID(y), Mul(), ID(z))

postfix: List(ID(x), ID(y), ID(z), Mul(), Add())