Samples of various ASTs used in the textbook

First draft: January 9, 2018

These ASTs can be used in your compiler project. Since these ASTs are designed for a Java-like language, you may modify them for our mini-Pascal language.

These ASTs are taken from Chapters 8 and 9 of our slides. More detailed explanation can be found there.

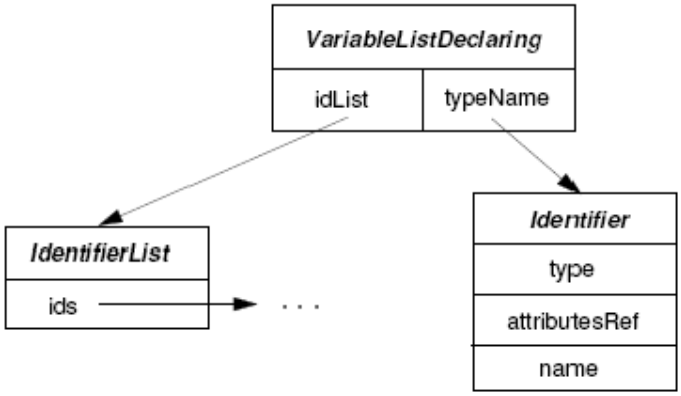


Figure 8.12: Abstract Syntax Tree for Variable Declarations

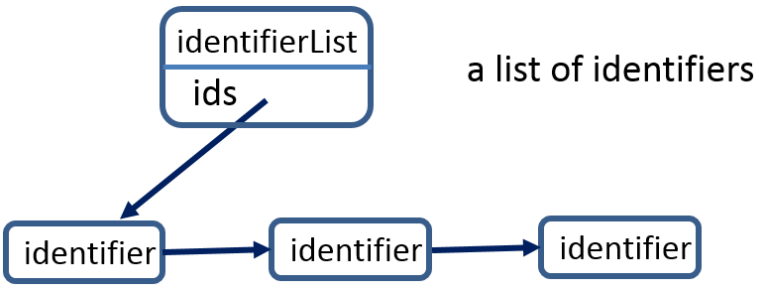


Figure 1: AST for a list of identifiers.

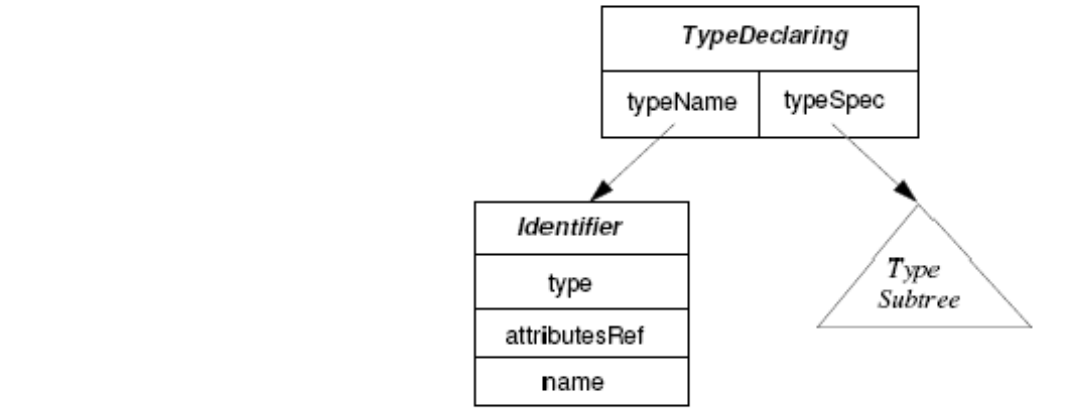


Figure 8.15: Abstract Syntax Tree for Type Declarations

Figure 2: Abstract syntax tree for type declaration

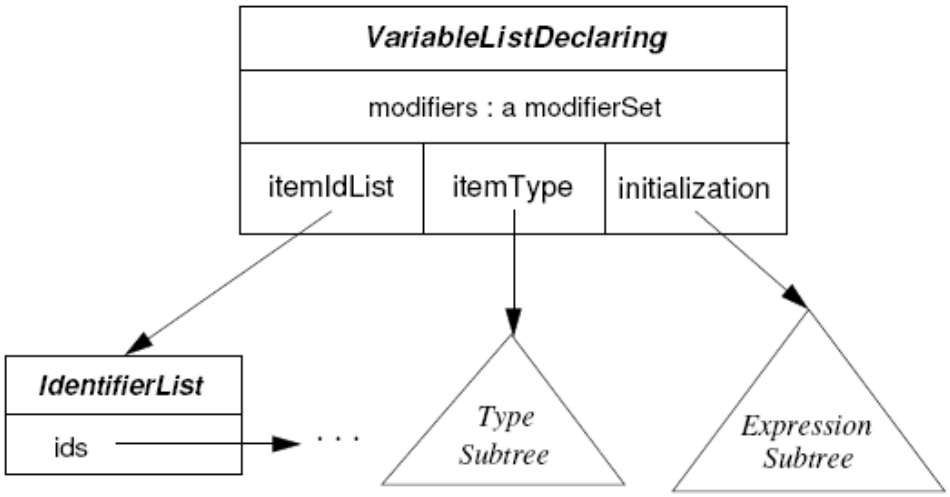


Figure 8.17: AST for Generalized Variable Declarations

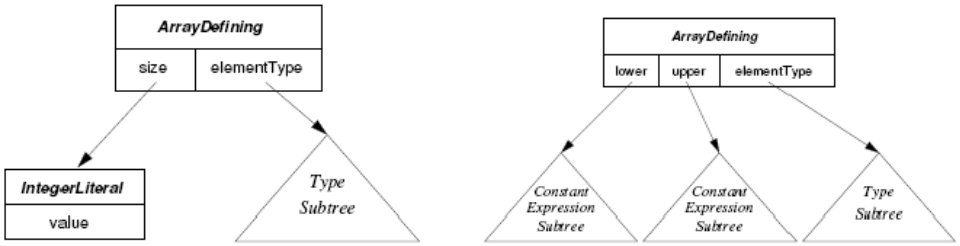


Figure 8.19: Abstract Syntax Trees for Array Definitions

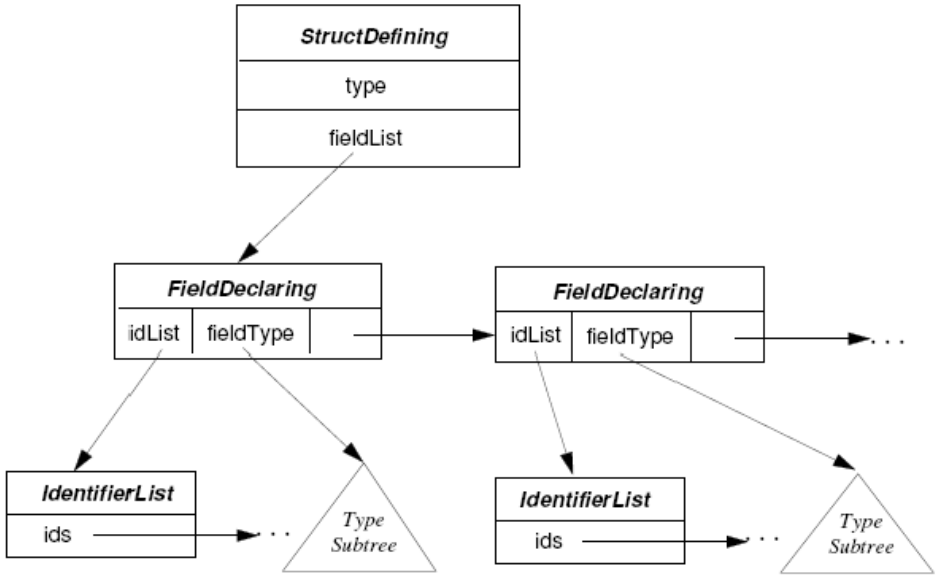


Figure 8.21: Abstract Syntax Tree for a Struct Definition

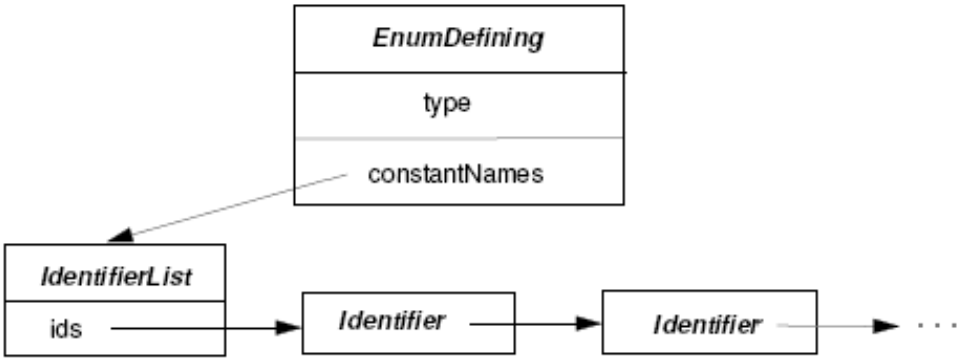


Figure 8.24: Abstract Syntax Tree for an Enumeration Type

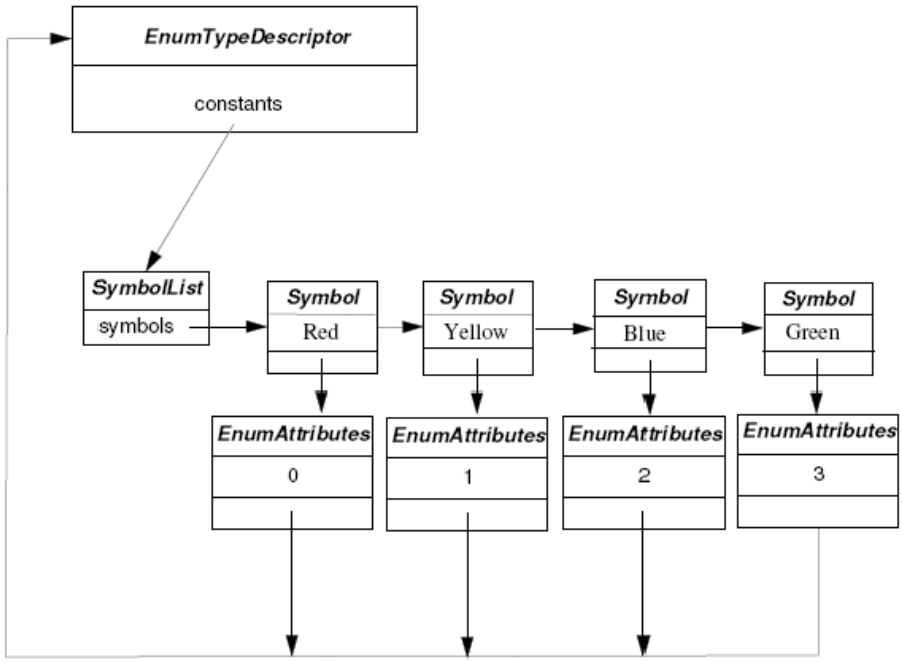


Figure 8.26: Representation of an Enumeration Type

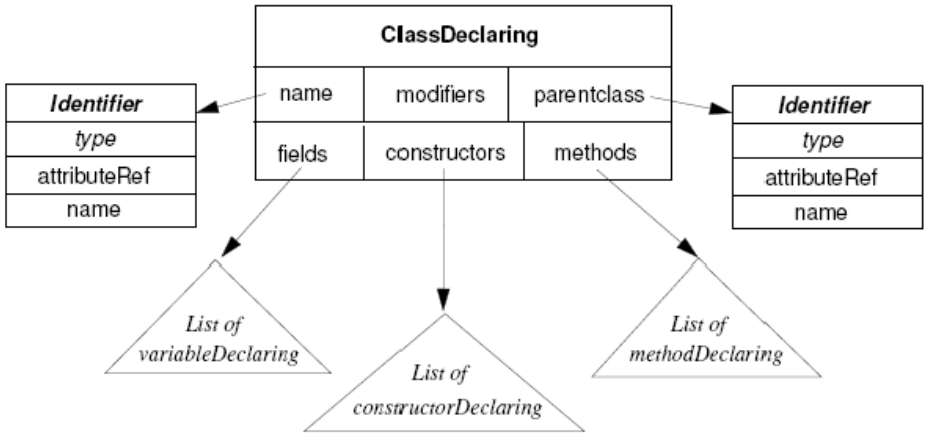


Figure 8.27: Abstract Syntax Tree for a Class Declaration

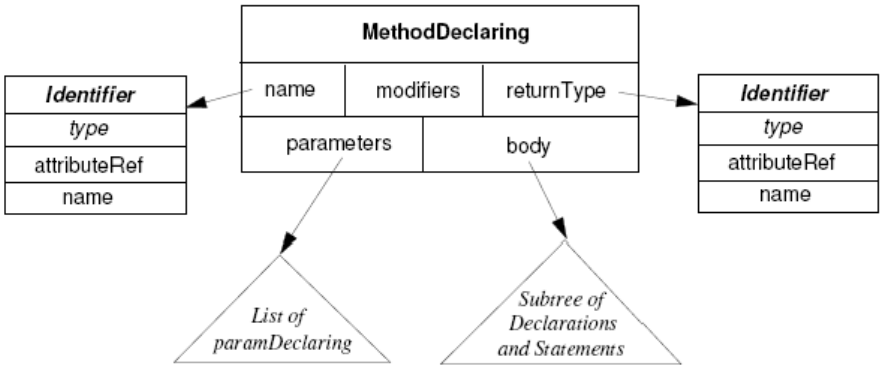


Figure 8.30: Abstract Syntax Tree for a Method Declaration

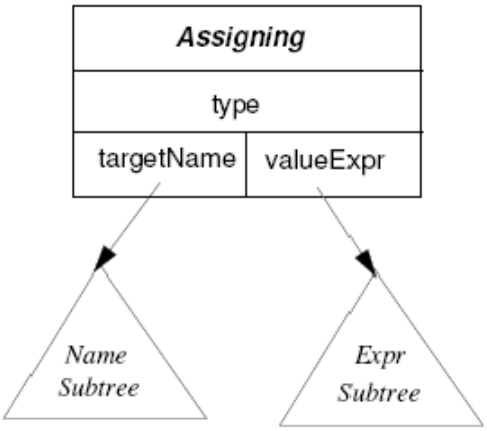


Figure 8.32: Abstract Syntax Tree for an Assignment

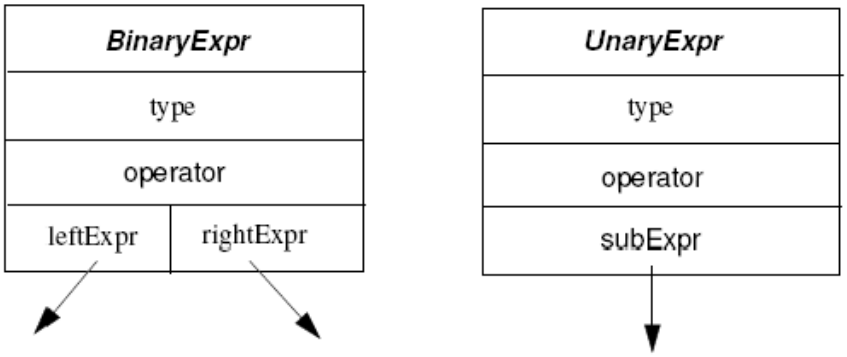


Figure 8.35: Abstract Syntax Tree Representations for Unary and Binary Expressions

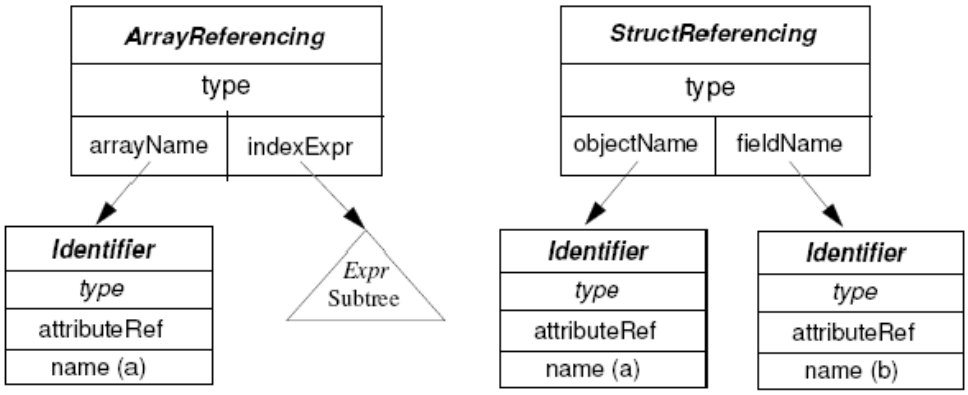


Figure 8.36: Abstract Syntax Trees for Array and Struct References

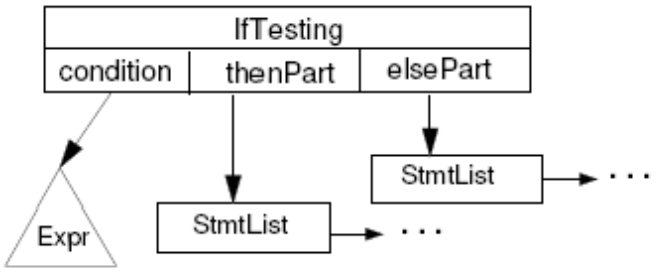


Figure 9.2: Abstract Syntax Tree for an If Statement

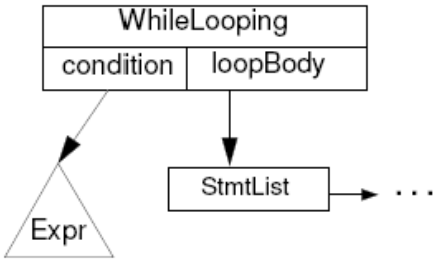


Figure 9.5: Abstract Syntax Tree for a While Statement

```
procedure      (WhileLooping wn)
    wn.terminatesNormally ← true                                    (21)
    wn.loopBody.isReachable ← true
    constExprVisitor ← new ConstExprVisitor()
    call wn.condition.          (constExprVisitor)
    conditionValue ← wn.condition.exprValue
    if conditionValue = true
    then
        wn.terminatesNormally ← false                               (22)
    else
        if conditionValue = false
        then
            wn.loopBody.isReachable ← false                         (23)
    call wn.loopBody.           (this)                              (24)
end
```

Figure 9.6: Reachability Analysis for a While Statement

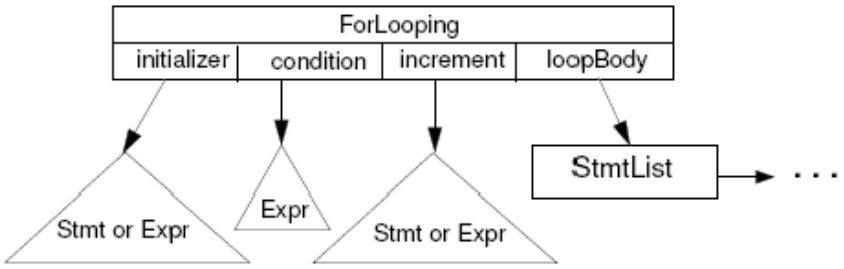


Figure 9.9: Abstract Syntax Tree for a For Loop

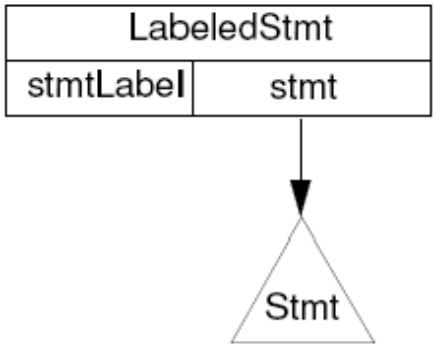


Figure 9.10: Abstract Syntax Tree for a Labeled Statement

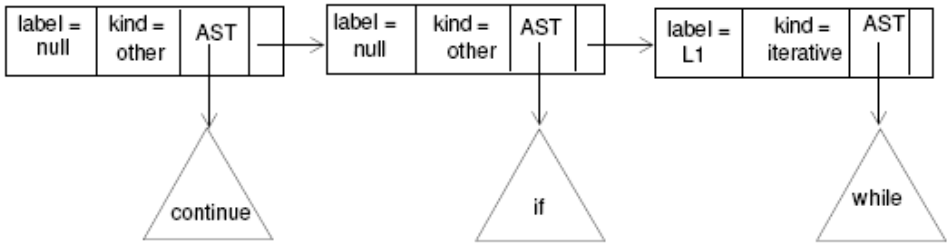


Figure 9.13: Example of a label list in a Continue Statement

```
procedure      (Returning rn)
    call       C          (rn)
    currentMethod ←    C          M        (rn)
    if rn.returnVal ≠ null
    then
        if currentMethod = null
        then
            call       ("A value may not be returned from a constructor")
        else
            if not              (currentMethod.returnType, rn.returnValue.type)
            then call          ("Illegal return type")
    else
        if currentMethod ≠ null and currentMethod.returnType ≠ void
        then  call         ("A value must be returned")
end
```

Figure 9.18: Semantic Analysis for a Return

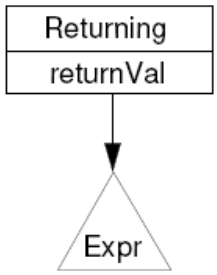


Figure 9.19: Abstract Syntax Tree for a Return Statement

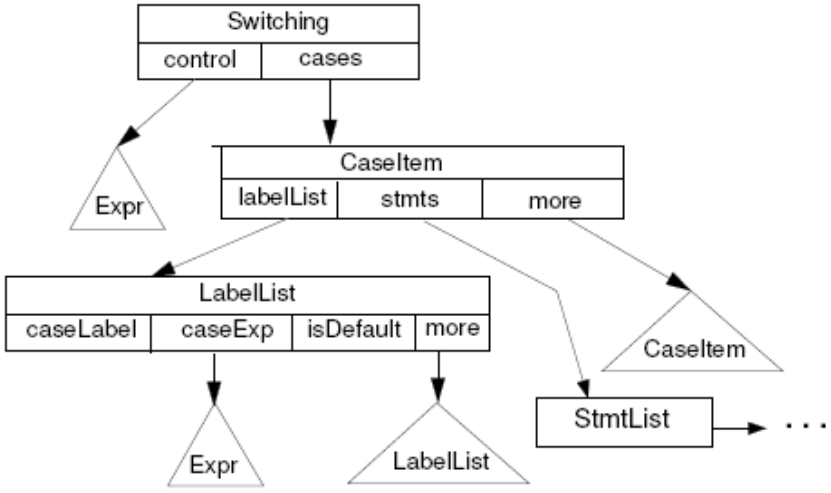
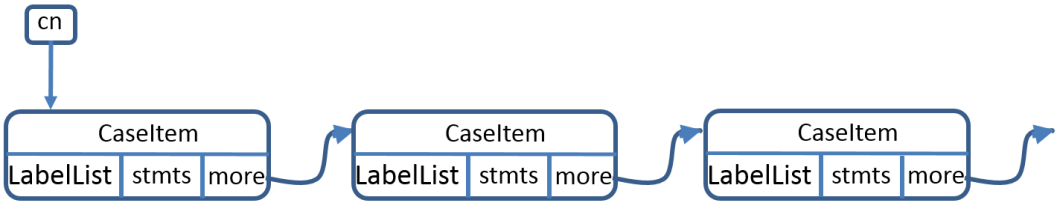


Figure 9.20: Abstract Syntax Tree for a Switch Statement



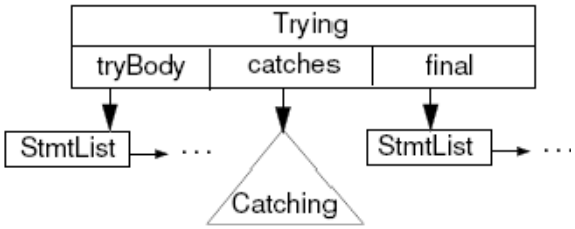


Figure 9.24: Abstract Syntax Tree for a Try Statement

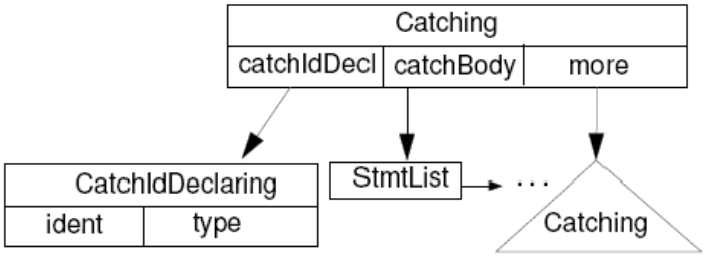


Figure 9.25: Abstract Syntax Tree for a Catch Block

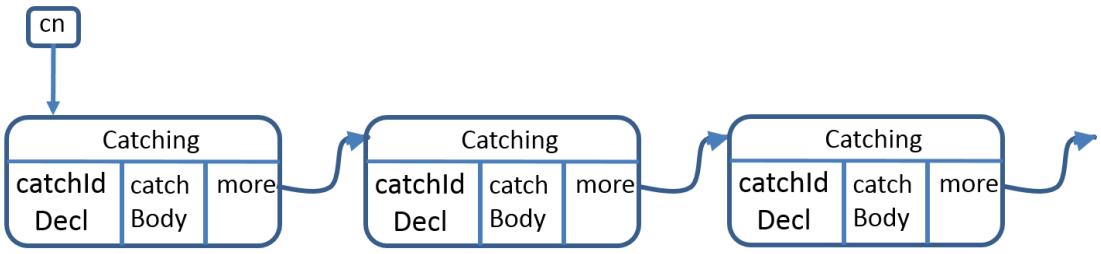


Figure 3: A list of `Catching` nodes suitable for tail recursion in the `Visit(Catching)` method.

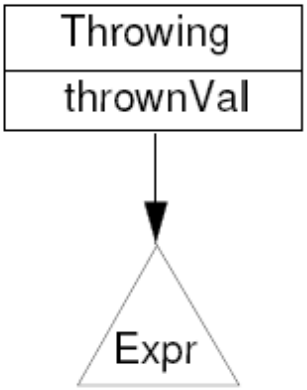


Figure 9.30: Abstract Syntax Tree for a Throw Statement

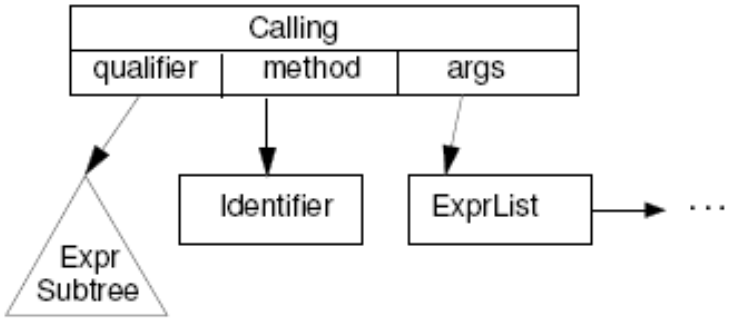


Figure 9.31: Abstract Syntax Tree for a Method Call