Transformers

Matt Might

University of Utah

matt.might.net

```
function id(x)
{
  return x ;  // comment
}
```

FUNCTION

IDENT(id)

LPAR

IDENT(x)

RPAR

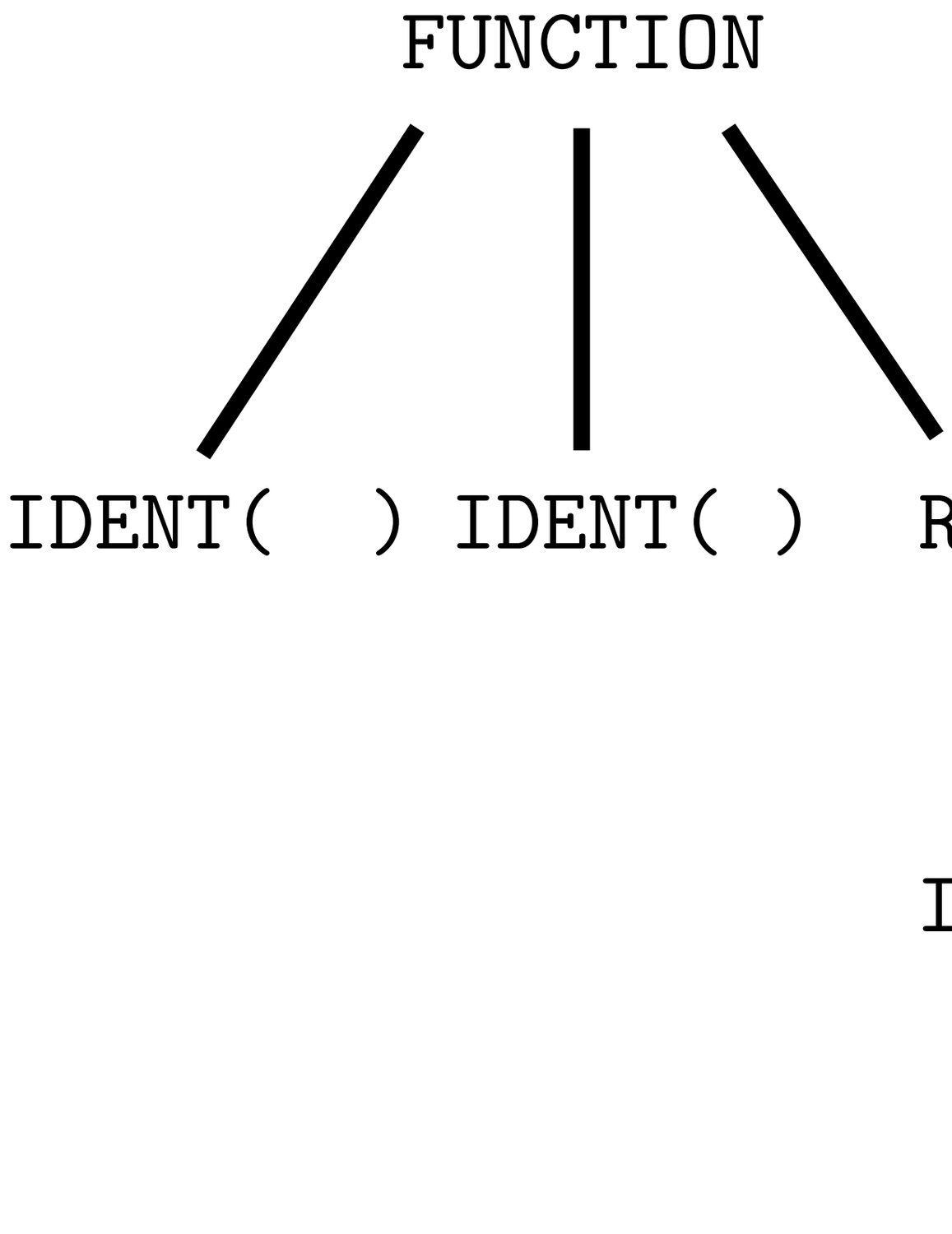
LBRACE

RETURN

IDENT(x)

SEMI

RBRACE



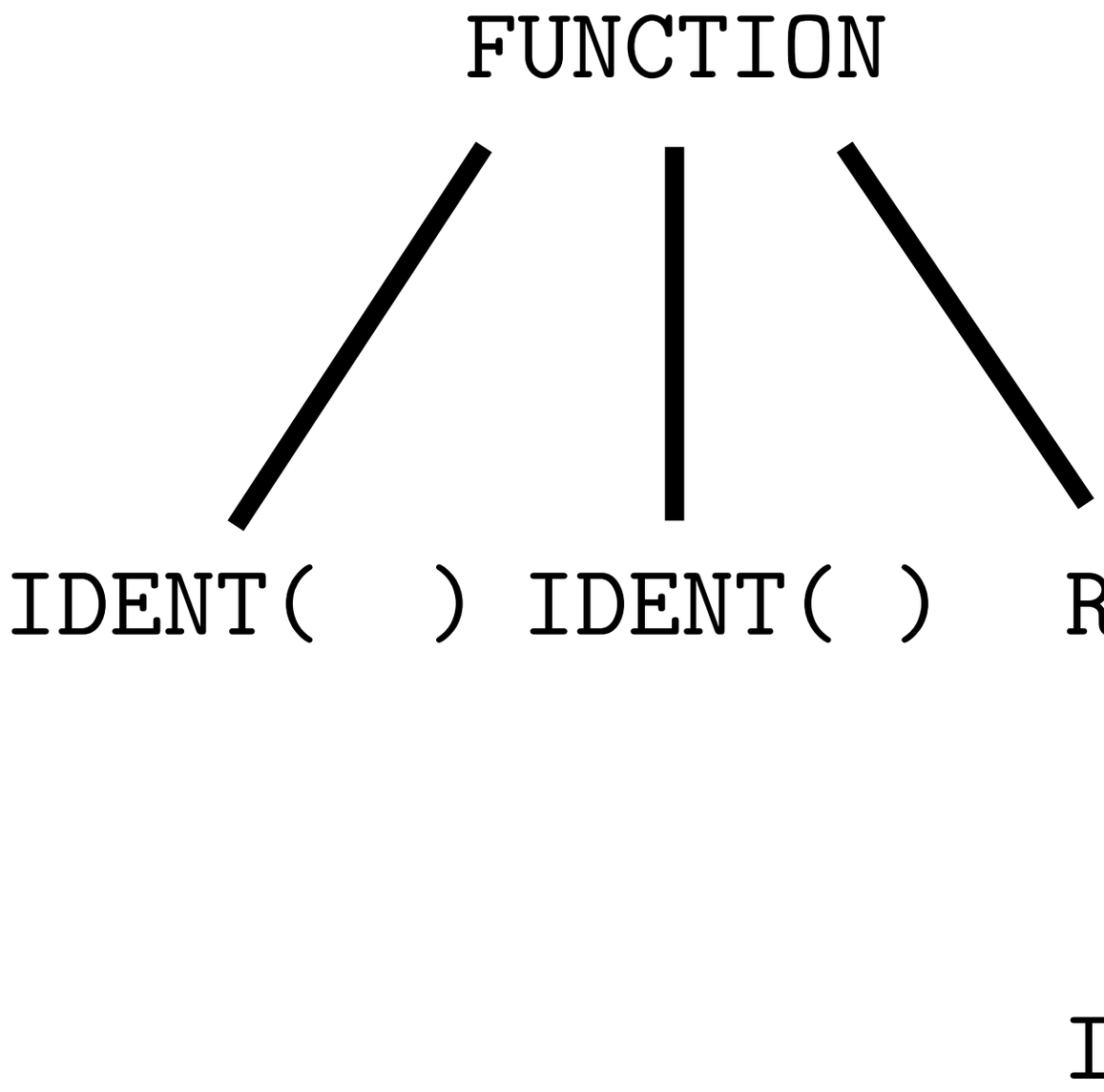
LPAR

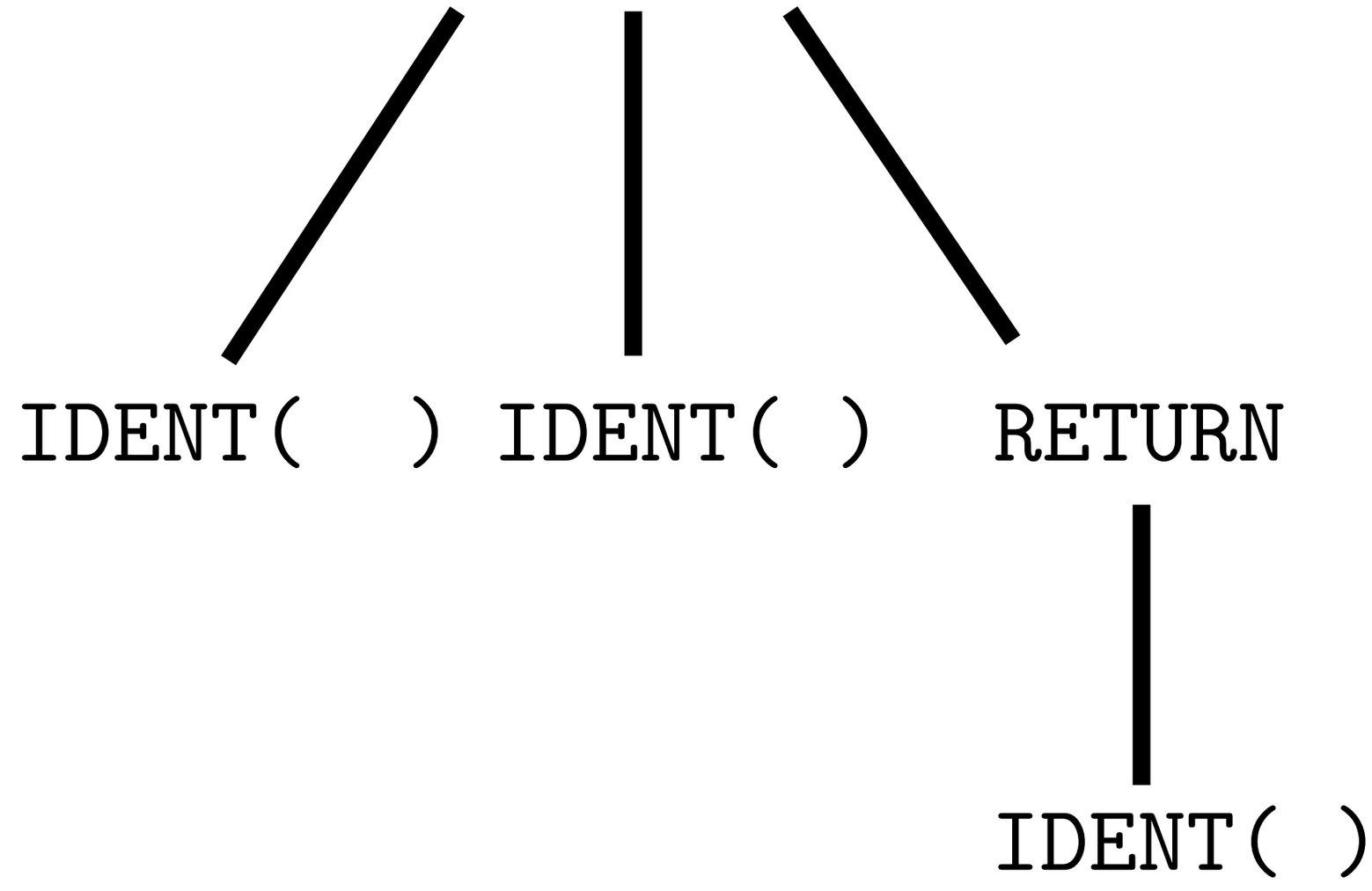
RPAR

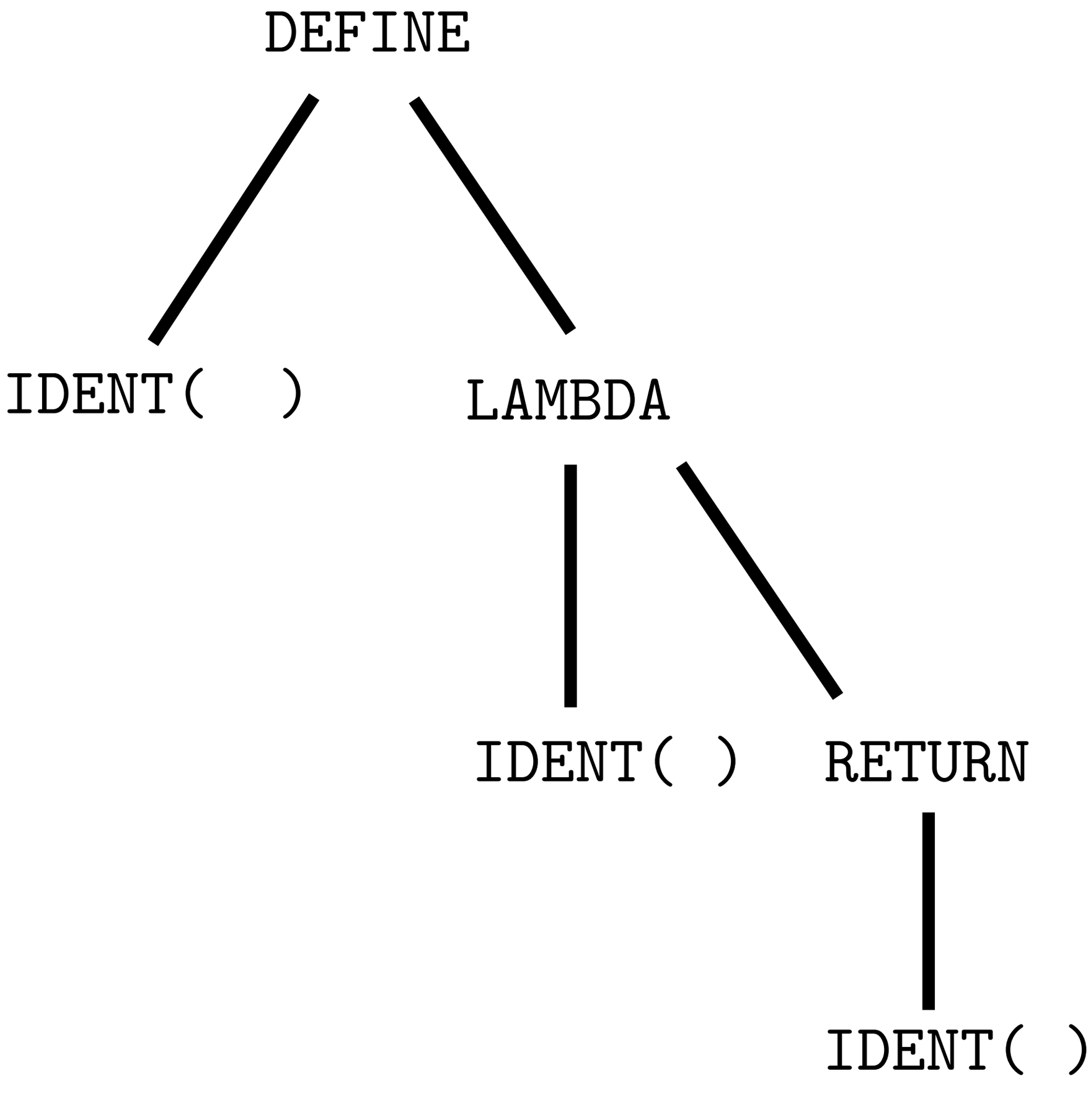
LBRACE

SEMI

RBRACE


FUNCTION
IDENT( ) IDENT( ) RETURN
IDENT( )


FUNCTION
IDENT( ) IDENT( ) RETURN
IDENT( )


DEFINE
IDENT( )
LAMBDA
IDENT( )
RETURN
IDENT( )

Types of transforms

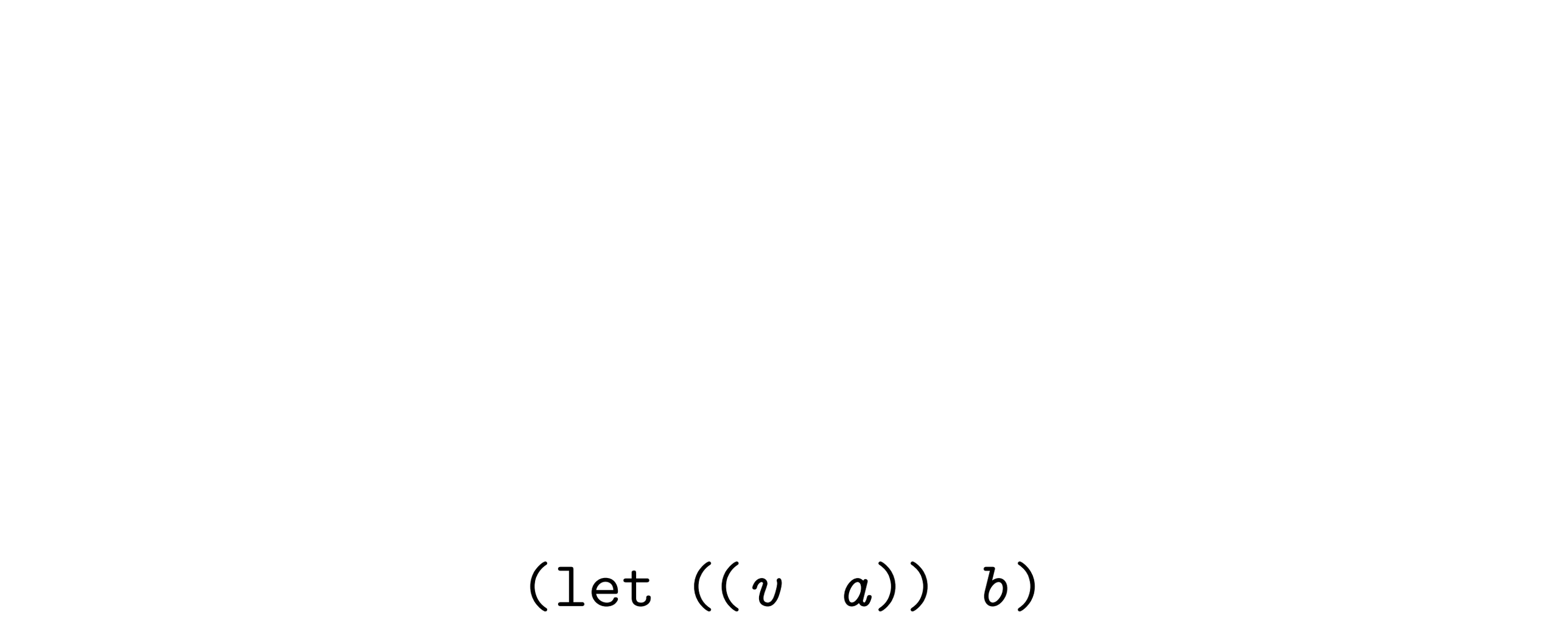
Canonicalization

```
(define (f x) x)
```

```
(define pi 3.14)
```

```
(define f (λ (x) x))
(define pi 3.14)
```

Desugaring


(let ((v a)) b)

```
((λ (v) b) a)
```

Translation

(set! v a)

```
mov $v a
```

```
'(1 2 3)
```

```
(define `λ (λ 'love 'I 'you))
((λ λ λ)
  `(λ (⊙ λ ∪) λ)
  `(λ (λ ⊙ ∪) λ)
  `(λ (∪ ⊙ λ) λ)))
```

Source: David Van Horn

Midterm: March 7

P1 *new* draft grades:

Out today

P2: New grading process

```
(define `λ (λ 'love 'I 'you))
((λ λ λ)
  `(λ (⊙ λ ∪) λ)
  `(λ (λ ⊙ ∪) λ)
  `(λ (∪ ⊙ λ) λ)))
```

Source: David Van Horn

A message from the Bolick Brothers

List transforms

Compiling quotes

Traversal styles

Questions?

High-level translation

Compile to high-level language
(Scheme, Java, JavaScript)

Why high-level?

- Easier than targeting low-level language
- Better performance than interpretation
- Access to large libraries (JVM, .NET)
- (Project 3 is a high-level translator)

Compiling to Java

- Value classes and a run-time environment
- Mapping constructs from Scheme to Java

Value.java

Language

```
<exp> ::= <const>
        |  <var>
        |  <prim>
        |  (lambda (<var> ...) <exp>)
        |  (if <exp> <exp> <exp>)
        |  (set! <var> <exp>)
        |  (let ((<var> <exp>) ...) <exp>)
        |  (letrec ((<var> <lam>)) <exp>)
        |  (begin <exp> ...)
        |  (<exp> <exp> ...)
```

<int>

<prim>

(λ (<var> ...) <exp>)

(set! <var> <exp>)

<var>

```
(if <exp1>
    <exp2>
    <exp3>)
```

(<exp$_0$> <exp$_1$> ...)