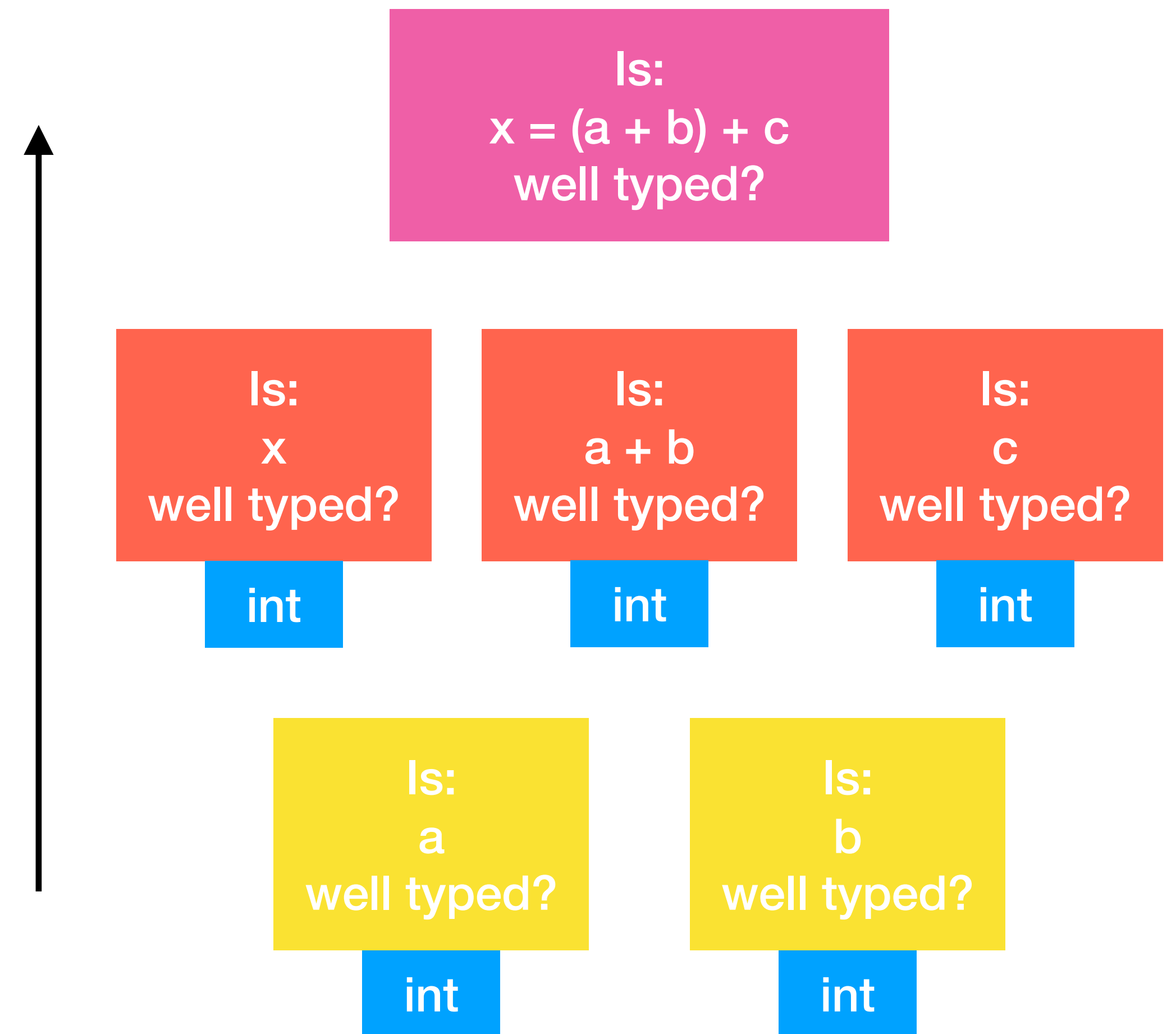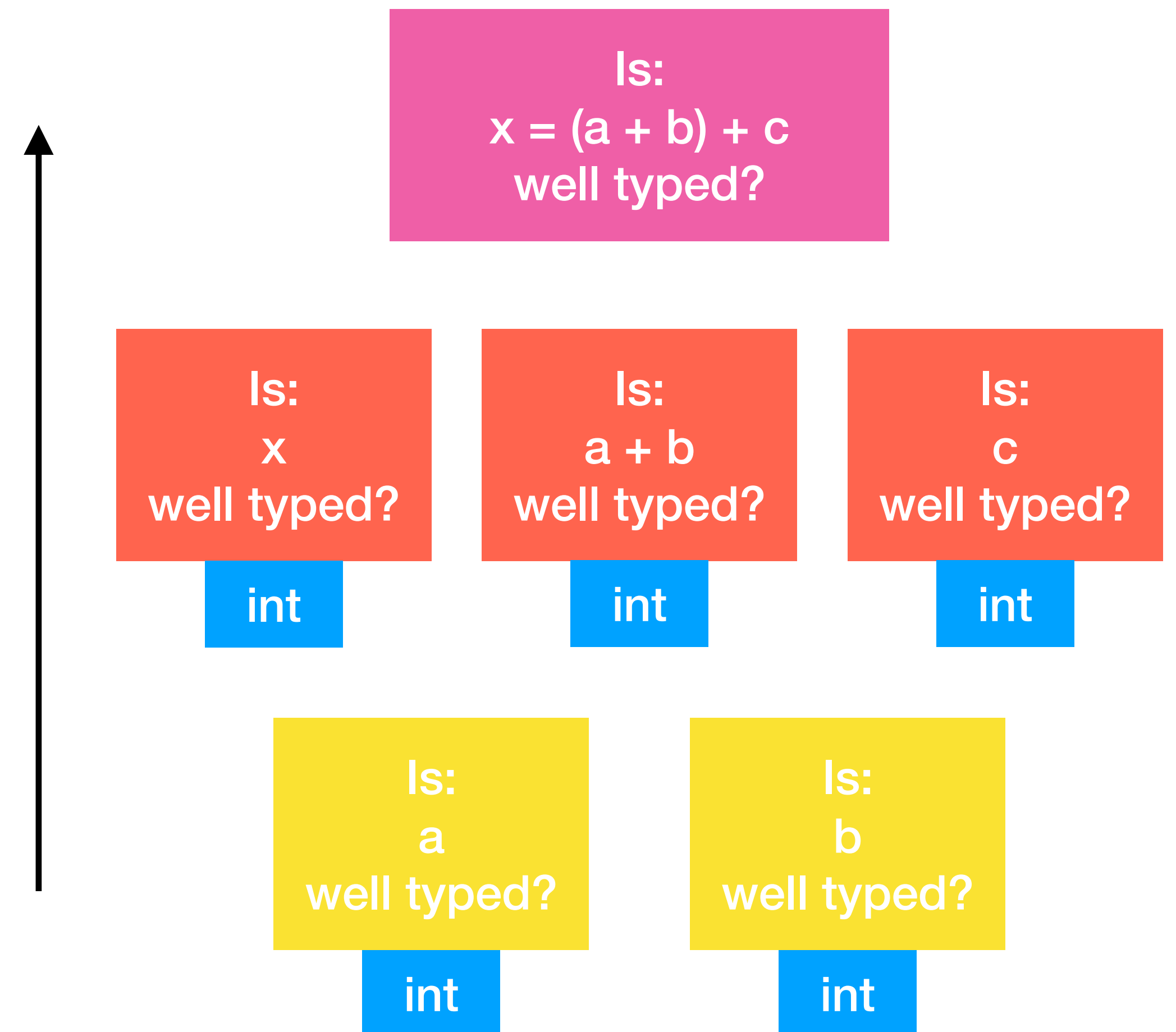# Implementing Type Checking

walk the AST

- For each syntactic structure that can have a type, add a type field to the AST
- Perform a post-order walk of the AST to assign types to each node in the AST

- Base cases:
  - Variables: get types from symbol table
  - Literals: get types from node type

Is:
x = (a + b) + c
well typed?

Is:
x
well typed?

int

Is:
a + b
well typed?

int

Is:
c
well typed?

int

Is:
a
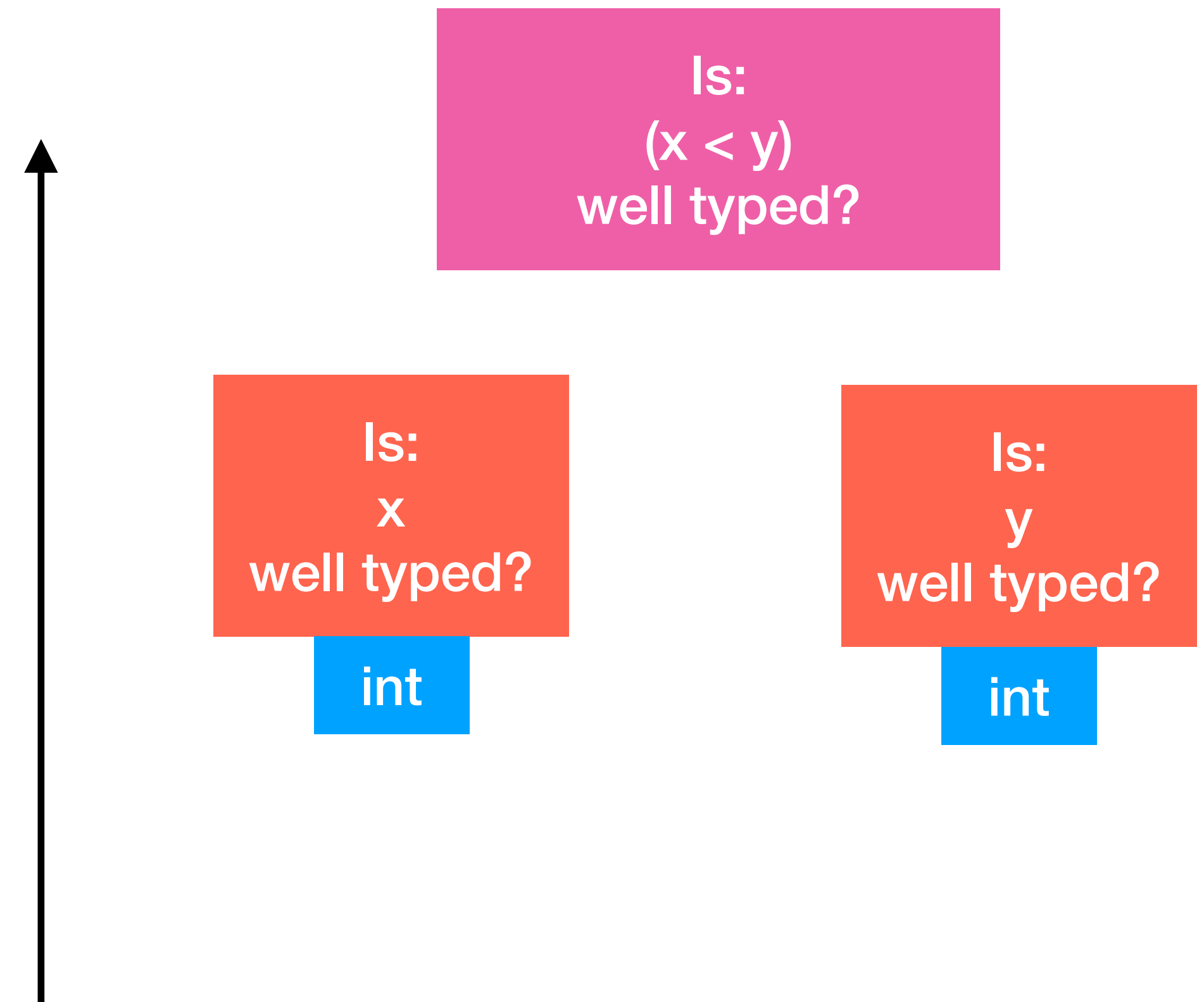well typed?

int

Is:
b
well typed?

int

walk the AST

- For each syntactic structure that can have a type, add a type field to the AST
- Perform a post-order walk of the AST to assign types to each node in the AST

- Inductive cases:
  - Expressions: compare types of component sub-expressions
  - Assignment: compare LHS and RHS

Is:
x = (a + b) + c
well typed?

Is:
x
well typed?

int

Is:
a + b
well typed?

int

Is:
c
well typed?

int

Is:
a
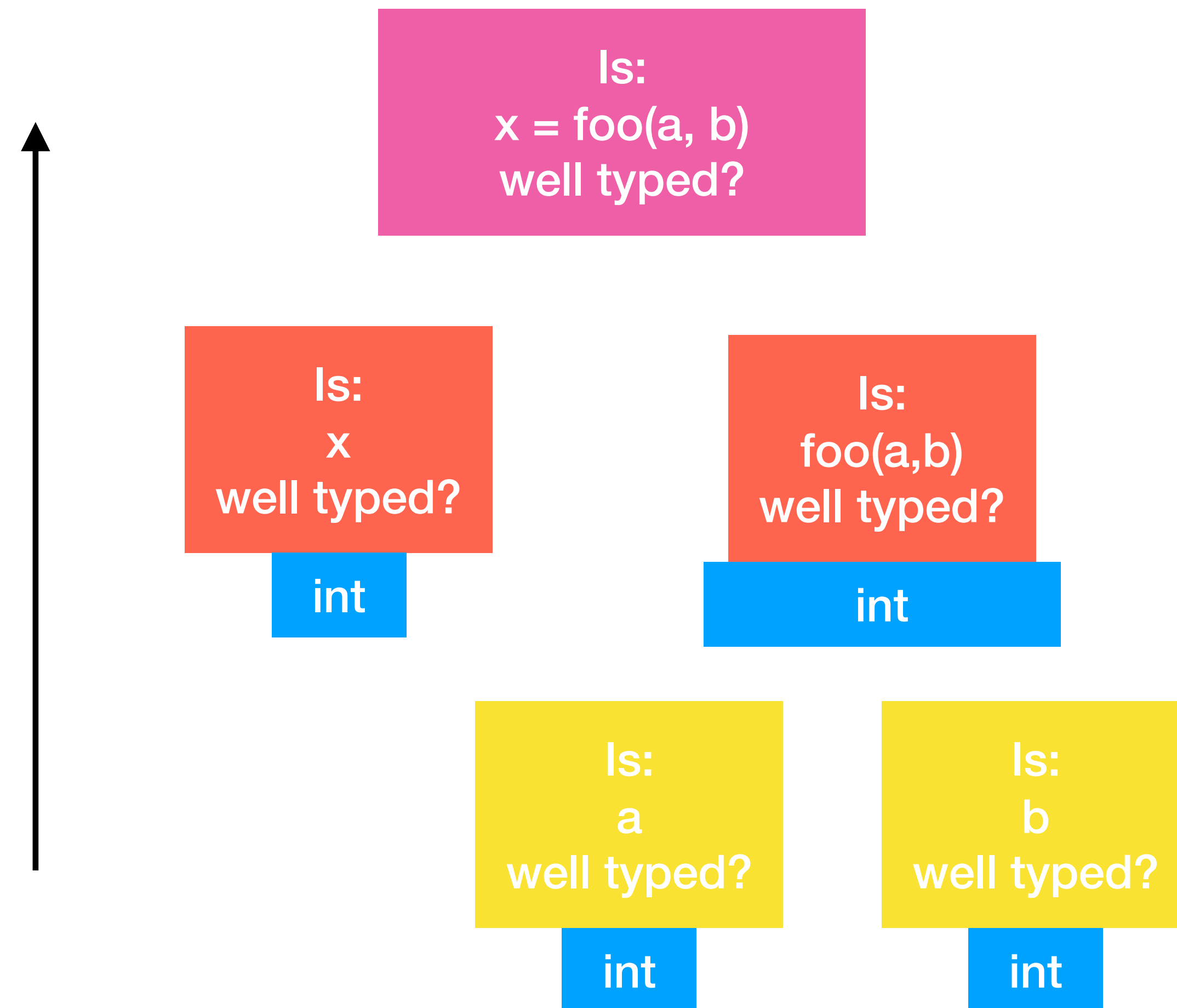well typed?

int

Is:
b
well typed?

int

walk the AST

- For each syntactic structure that can have a type, add a type field to the AST
- Perform a post-order walk of the AST to assign types to each node in the AST

- Inductive cases:
  - Expressions: compare types of component sub-expressions
  - Assignment: compare LHS and RHS
  - Conditionals: compare LHS and RHS

Is:
(x < y)
well typed?

Is:
x
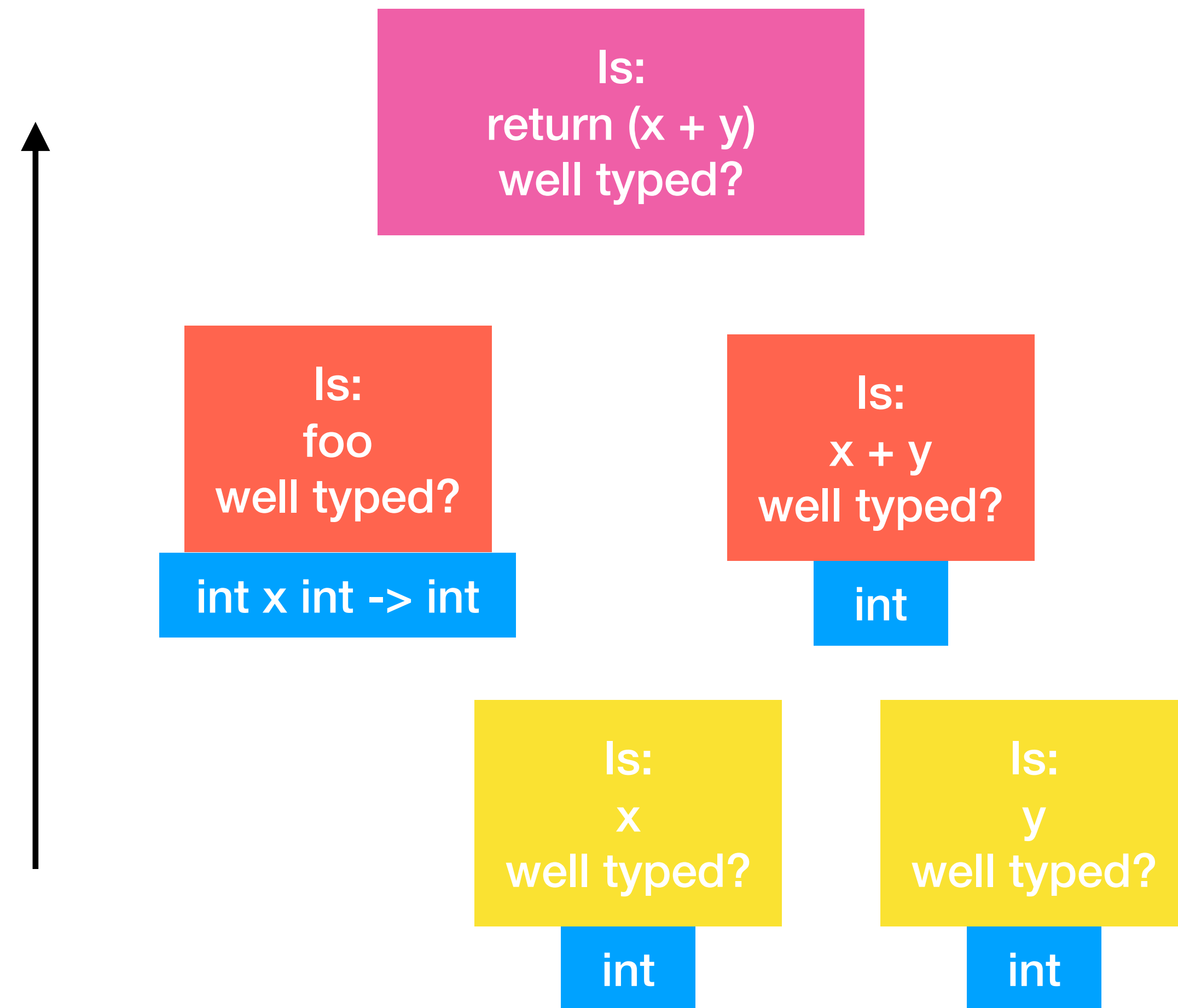well typed?

int

Is:
y
well typed?

int

walk the AST

- For each syntactic structure that can have a type, add a type field to the AST
- Perform a post-order walk of the AST to assign types to each node in the AST

- Inductive cases:
  - Function calls: compare types of sub-expressions to argument types in symbol table
  - Note: type assigned to function call should be the *return* type of the function!

Is:
x = foo(a, b)
well typed?

Is:
x
well typed?
int

Is:
foo(a,b)
well typed?
int

Is:
a
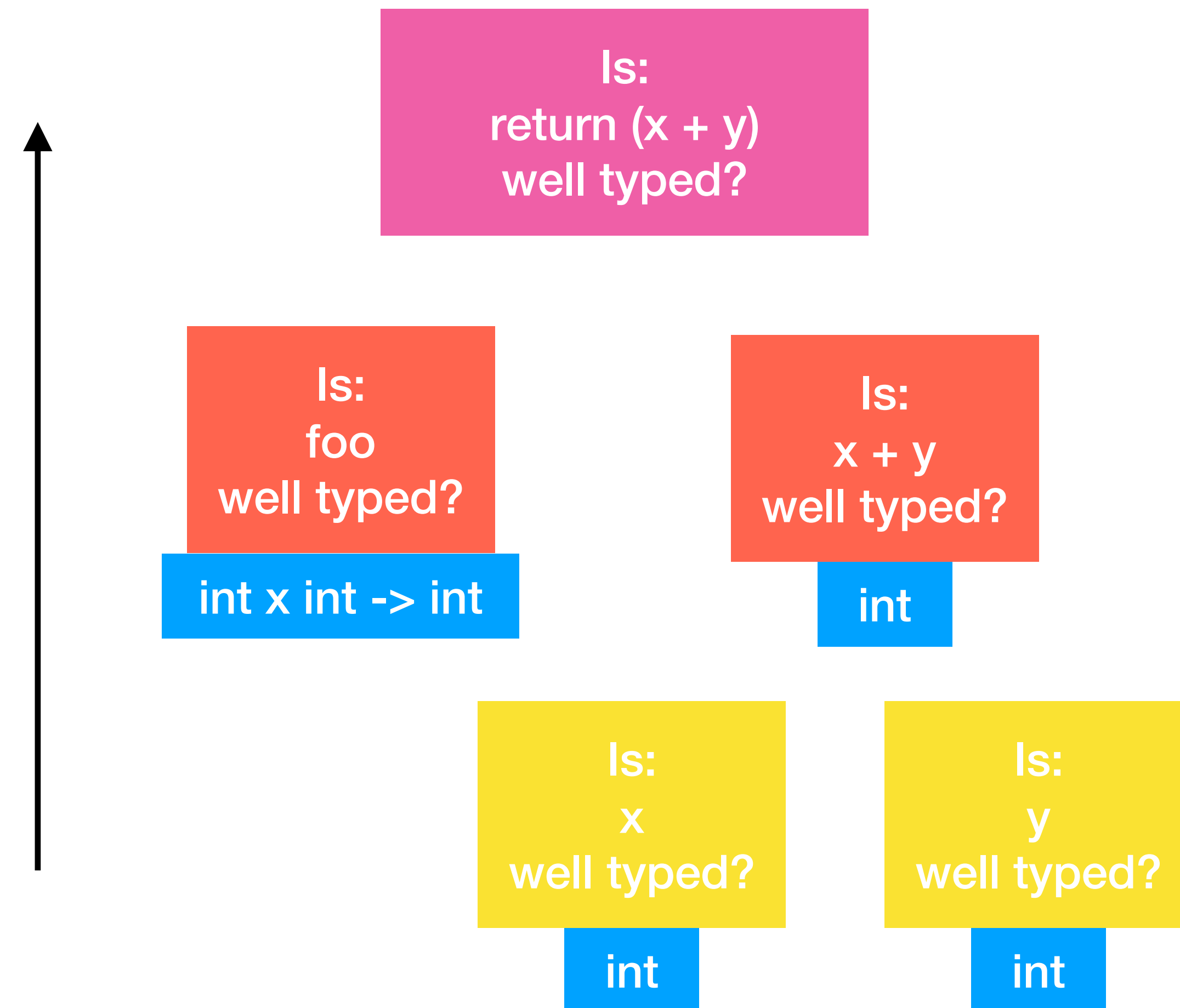well typed?
int

Is:
b
well typed?
int

walk the AST

- For each syntactic structure that can have a type, add a type field to the AST
- Perform a post-order walk of the AST to assign types to each node in the AST

- Inductive cases:
  - Function calls: compare types of sub-expressions to argument types in symbol table
  - Return statements: compare return expression type to return type of function in symbol table

Is:
return (x + y)
well typed?

Is:
foo
well typed?

int x int -> int

Is:
x + y
well typed?

int

Is:
x
well typed?

int

Is:
y
well typed?

int

walk the AST

- For each syntactic structure that can have a type, add a type field to the AST
- Perform a post-order walk of the AST to assign types to each node in the AST

- If any node cannot be typed, return an error!

Is:
return (x + y)
well typed?

Is:
foo
well typed?

int x int -> int

Is:
x + y
well typed?

int

Is:
x
well typed?

int

Is:
y
well typed?

int

next: optimization