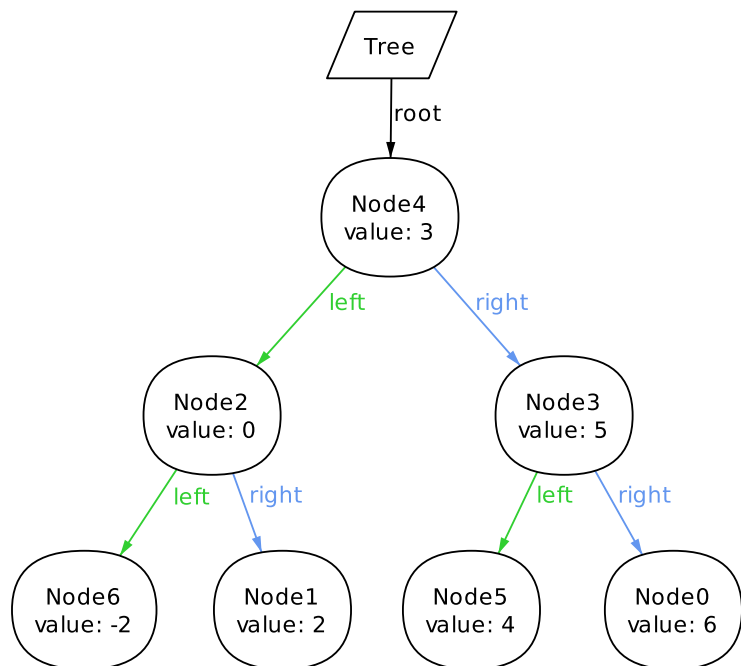*Relational Logic*

Alloy[90,91] is a first-order logic with relations and transitive closure. *Relation* is a fancy word for table; *tuple* is a fancy word for row. An intuition for transitive closure is 'where can we get to from here?'

```
sig Tree { root : Node }
sig Node {
    left , right : lone Node ,
    value : one Int ,
}
```

Figure 9: An Alloy model of a binary tree. An intuition, good to a first-order approximation, is to read this like class declarations in an object-oriented language. Going beyond this intuition, root, left, right, and value are really binary relations; see the example below.

Figure 10: An instance of a binary tree. If we write the relations from this figure out in tabular form they look like this:

Tree		root	
Tree0		Tree0	Node4

Node		value	
Node0		Node0	6
Node1		Node1	2
Node2		Node2	0
Node3		Node3	5
Node4		Node4	3
Node5		Node5	4
Node6		Node6	-2

left		right	
Node4	Node2	Node4	Node3
Node2	Node6	Node2	Node1
Node3	Node5	Node3	Node0

Int is the set of all integers, and iden is the identity relation (a binary relation that maps every atom to itself).

	iden	
	Tree0	Tree0
	Node0	Node0
	Node1	Node1
Int	Node2	Node2
-8	Node3	Node3
-7	Node4	Node4
:	Node5	Node5
:	Node6	Node6
6	-8	-8
7	-7	-7
	:	:
	6	6
	7	7

union

left + right	
Node4	Node2
Node2	Node6
Node3	Node5
Node4	Node3
Node2	Node1
Node3	Node0

transitive closure

^left	
Node4	Node2
Node2	Node6
Node3	Node5
Node4	Node6

transitive closure

^right	
Node4	Node3
Node2	Node1
Node3	Node0
Node4	Node0

join

Tree0.root
Node4

root.^(left + right)	
Tree0	Node0
Tree0	Node1
Tree0	Node2
Tree0	Node3
Tree0	Node5
Tree0	Node6

Comprehension questions:

1. Why isn't Node4 included in this relation?

2. How could the expression be changed so that Node4 would be included?

3. What does the expression Tree0.root.^(left + right) evaluate to?

reflexive transitive closure: *r ≡ ^r + iden