Semester roadmap:

Ch 1: Sets
Ch 2: Sequences
—Test 1, Sept 25 —
Ch 3: Symbolic logic
Ch 4: Proofs
— Test 2, Oct 18 —
Ch 5: Relations
Ch 6: Functions
— Test 3, Nov 22 —
Ch 7: Self reference

Chapter 7 roadmap:

- Recursively-defined sets (**Today**)
- Recursive proofs I: Structural induction (next week Monday)
- Recursive proofs II: Mathematical induction (next week Wednesday)
- Non-recursive programs—loops (next week Friday)
- Recursive proofs III: Loop invariants (week-after Monday)
- A language processor (week-after Wednesday)

Axiom 7

There exists a whole number 0.

Axiom 8

Every whole number n has a successor, $\texttt{succ}\ n$*.*

Axiom 9

No whole number has 0 as its successor.

Axiom 10

If $a, b \in \mathbb{W}$, then $a = b$ iff $\texttt{succ}\ a = \texttt{succ}\ b$.

A *whole number* is either zero or one more than another whole number.

Compare to:
A *pizza* is either crust or a topping together with its sub-pizza.

5 is a whole number because

5 is a whole number because it is the successor of 4, which is a whole number because

5 is a whole number because it is the successor of

4, which is a whole number because it is the successor of

3, which is a whole number because

5 is a whole number because it is the successor of

4, which is a whole number because it is the successor of

3, which is a whole number because it is the successor of

2, which is a whole number because

5 is a whole number because it is the successor of

4, which is a whole number because it is the successor of

3, which is a whole number because it is the successor of

2, which is a whole number because it is the successor of

1, which is a whole number because

5 is a whole number because it is the successor of

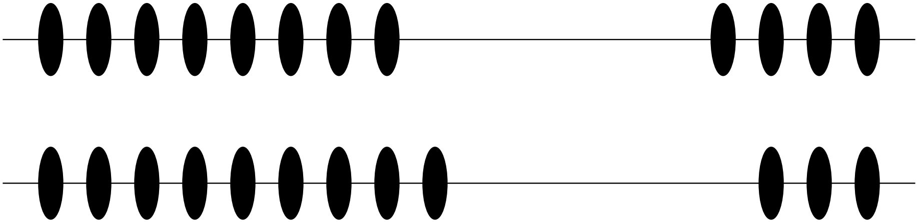
4, which is a whole number because it is the successor of

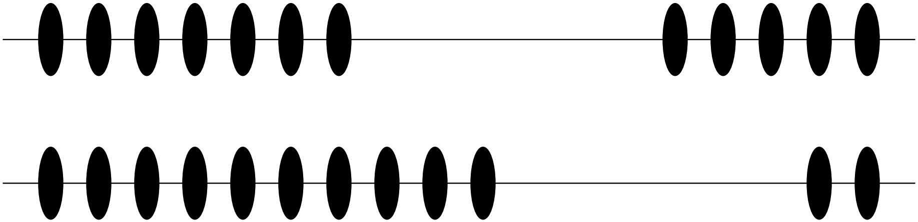
3, which is a whole number because it is the successor of

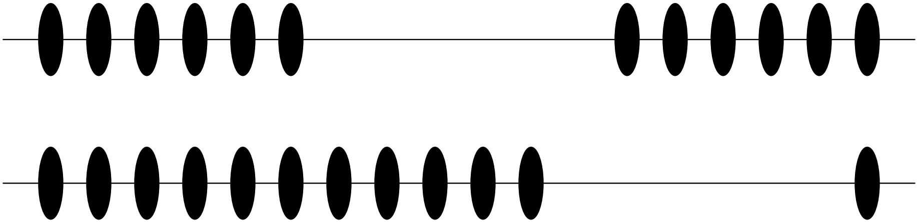
2, which is a whole number because it is the successor of

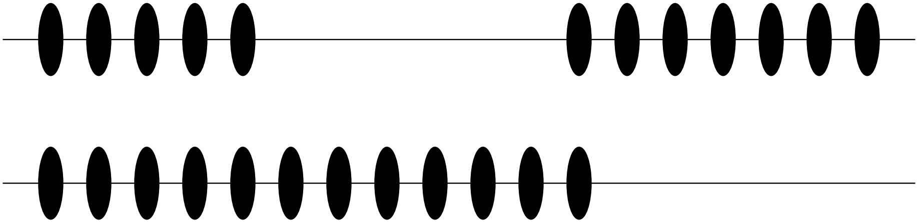
1, which is a whole number because it is the successor of

0, which is a whole number by Axiom 7.





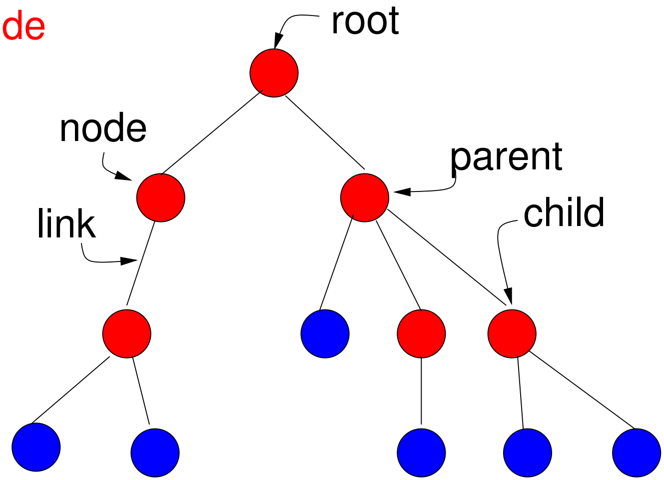




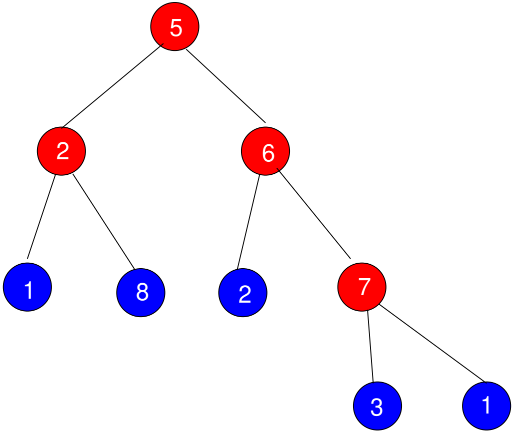
Tree

internal node

leaf



Full Binary Tree



Expression trees:

$$
\begin{aligned}
\textit{Expression} &\rightarrow \textit{Variable} \mid \textit{Constant} \\
&\quad \mid (\ \textit{Expression}\ \textit{Operator}\ \textit{Expression}\ ) \\[1em]
\textit{Operator} &\rightarrow + \mid - \mid * \mid /
\end{aligned}
$$