What is wrong with my induction proof?

In a drunken haze I decided that the solution to the recurrence $T(1)=1$, $T(n)= 1 + T(n-1)$ is

$1 + 2 + 3 + \ldots + n$.

Theorem: The solution to the recurrence is $n(n+1)/2$.

Proof. [Basis] $T(1)=1$ and $1 *(1+1)/2 = 1$ as required.

[Induction step] Assume that $1 + 2 + \ldots + n-1 + n = n(n+1)/2$.

We want to prove that $1 + 2 + \ldots + n-1 + n + (n+1) = (n+1)(n+2)/2 = (n^2 +3n +2)/2$.

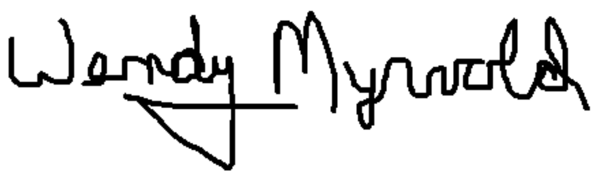
By induction, $1 + 2 + \ldots + n= n(n+1)/2$.

So $1 + 2 + .. + n + (n+1)= n(n+1)/2 + (n+1)$.

Simplifying: $(n^2 + n + 2n + 2)/2= (n^2 +3n +2)/2$ as required.

Students are expected to attend all the classes. In order to write the final exam, students should not miss more than 6 classes with the exception of extreme circumstances with appropriate documentation (for example, a note from a health care provider).

Your signature on the attendance sheet must be legible:



OK

NOT OK

If you want credit for attendance I should be able to read your name from what you have written.

DO NOT SIGN the sheet for a friend.

Announcements

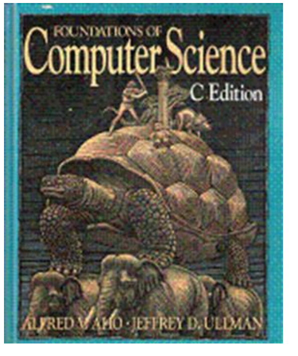
Assignment #1:

Part 1A Programming Questions: Upload to connex by before Saturday Sept. 23 at 11:55pm. You are allowed to resubmit an unlimited number of times up until the deadline.

Part 1B Written Questions: Now available. Hand in on paper at the beginning of class on Thursday September 28.

Powerpoint slides will be posted: click on the "Selected class notes" link on the course web page.

This free book is a better reference for induction and Big Oh notation than our text:



http://infolab.stanford.edu/~ullman/focs.html

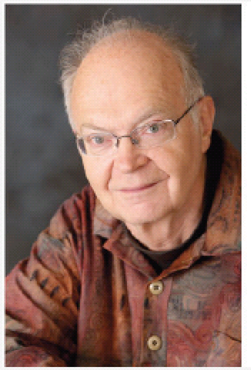
Slide by: Kevin Wayne
Princeton University

Why study algorithms?

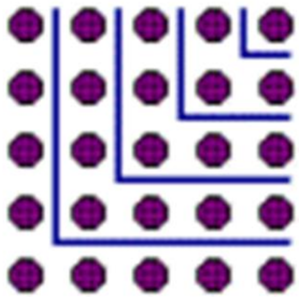
For intellectual stimulation.

> " *For me, great algorithms are the poetry of computation. Just like verse, they can be terse, allusive, dense, and even mysterious. But once unlocked, they cast a brilliant new light on some aspect of computing.* " — *F. Sullivan*

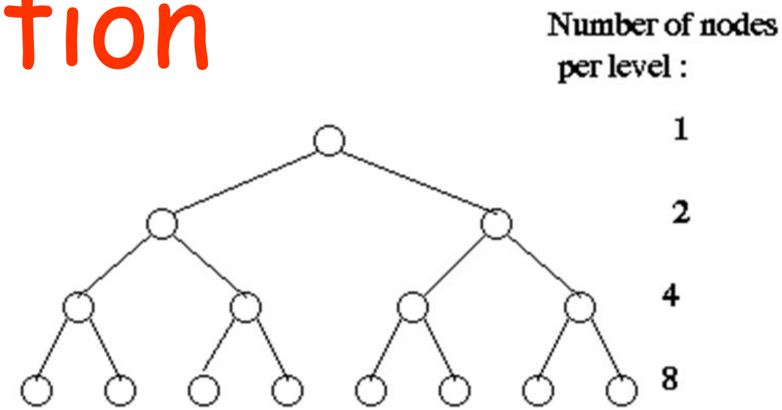
> " *An algorithm must be seen to be believed.* " — *D. E. Knuth*



Induction



$$n^2 = 1 + 3 + 5 + \ldots + (2n - 1)$$



Induction is very similar to recursion and one goal of this class is to become skilled at writing recursive programs.

It is a useful tool for proving that programs are correct.

Time complexities of algorithms will be computed by solving recurrence relations. Induction can then be used to prove that the answers you find are correct.

First class:

Prove by induction that the number of binary strings of length k is 2^k.

For example:

The binary strings of length 3 are:

000, 001, 010, 011,
100, 101, 110, 111

and there is 8 of them, and $8 = 2^3$.

Natural Numbers

$\mathbb{N} = \{ 0, 1, 2, 3, 4, \ldots \}$

Inductive Definition:

[Basis] 0 is in the set $\mathbb{N}$

[Inductive step]:

If k is in $\mathbb{N}$

then k+1 is in $\mathbb{N}$

Why do we write proofs?

Why do we write proofs?

To convince others that a particular mathematical statement is true. The arguments need to be convincing and easy to read.

Prove by induction that the number of binary strings of length k is 2^k.

My base case: k=0.
There is only one binary string of length 0, the empty string (which in CSC 320 we will denote by ε).

The formula states that the number of strings of length 0 is $2^k = 2^0 = 1$ and hence, the formula gives the correct value when k=0.

How do we know if we should include more cases in the base case or not?

For elegance: do not include unnecessary cases.

Assume that the number of binary strings of length k is 2^k. This mathematical statement is our induction hypothesis.

It is a good idea to write down what we are trying to prove next so that you will know you have arrived at the goal when you get there.

We want to prove that the number of binary strings of length k+1 is 2^{k+1}.

DIGRESSION:

How do we go from length k to length k+1?

It helps to start in this case with an inductive definition of a binary string.

Concatenation of strings x and y denoted $x \cdot y$ or simply xy means write down x followed by y.

Inductive definition of a binary string:

[Base Case] The empty string ε is a binary string.

[Induction step]

If w is a binary string, then the two strings

w 0

and

w 1

are also binary strings.

Continuing on with the proof:

Each binary string of length k+1 has a prefix of length k that is an arbitrary binary string of length k and this prefix is followed by one more symbol which is 0 or 1.

Recall our induction hypothesis: The number of binary strings of length k is 2^k.

By induction, the number of possible prefixes of length k is 2^k.

The number of choices for the last symbol is 2. Therefore, the total number of binary strings of length k+1 is equal to $2^k * 2 = 2^{k+1}$ as required.

Common problem in solutions submitted:

Using only algebra but no words or connections to strings. You will not get any marks for proofs like this in CSC 225.

Many students apply the induction hypothesis without explaining to the reader what you were doing. You will lose marks on the assignment if you do not explain where you are applying the induction hypothesis.

A proof is intended for someone to read. It will be easier for someone to understand and believe in your proof if you explain what you are doing algebraically at every step.

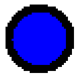
Your proof will be more elegant if you don't change the variable names (for this problem, stick to k instead of switching to n).

Induction:

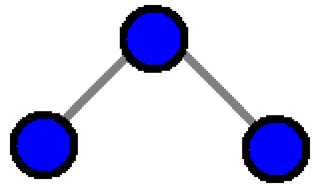
I want you to:

1. Understand why it works as a proof technique.
2. Write proofs that explain clearly what you are doing at every step (except for very simple algebra). Be sure to mention where it is that you apply the induction hypothesis. Everything you write should be mathematically valid.
3. Be able to use it on novel applications (requires understanding).
4. If you try to prove a hypothesis that is not correct, I want you to indicate where and why the induction proof fails. You will get zero marks for "proofs" for incorrect statements.
5. Elegance is good (e.g. don't put more in the base case than you really need).

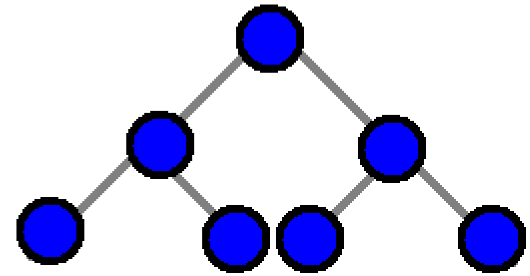
Complete Binary Trees:



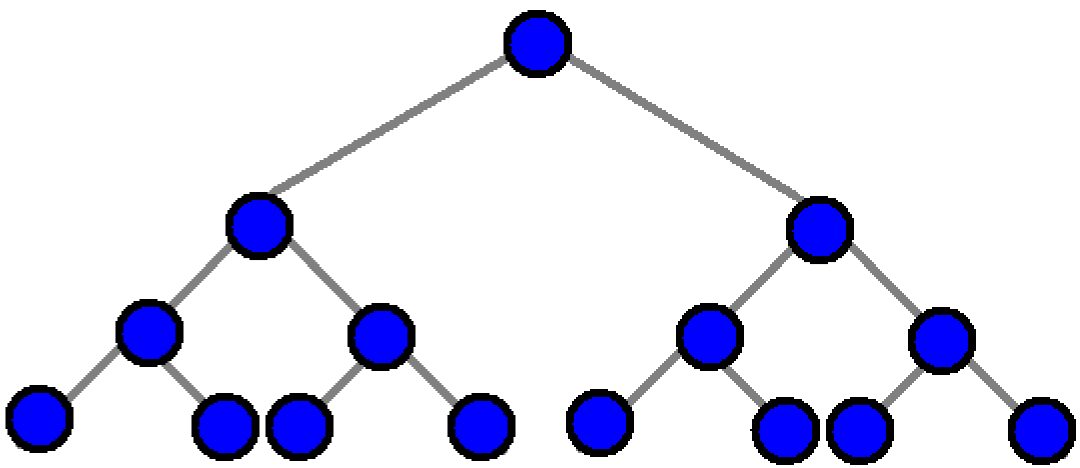
Height 0



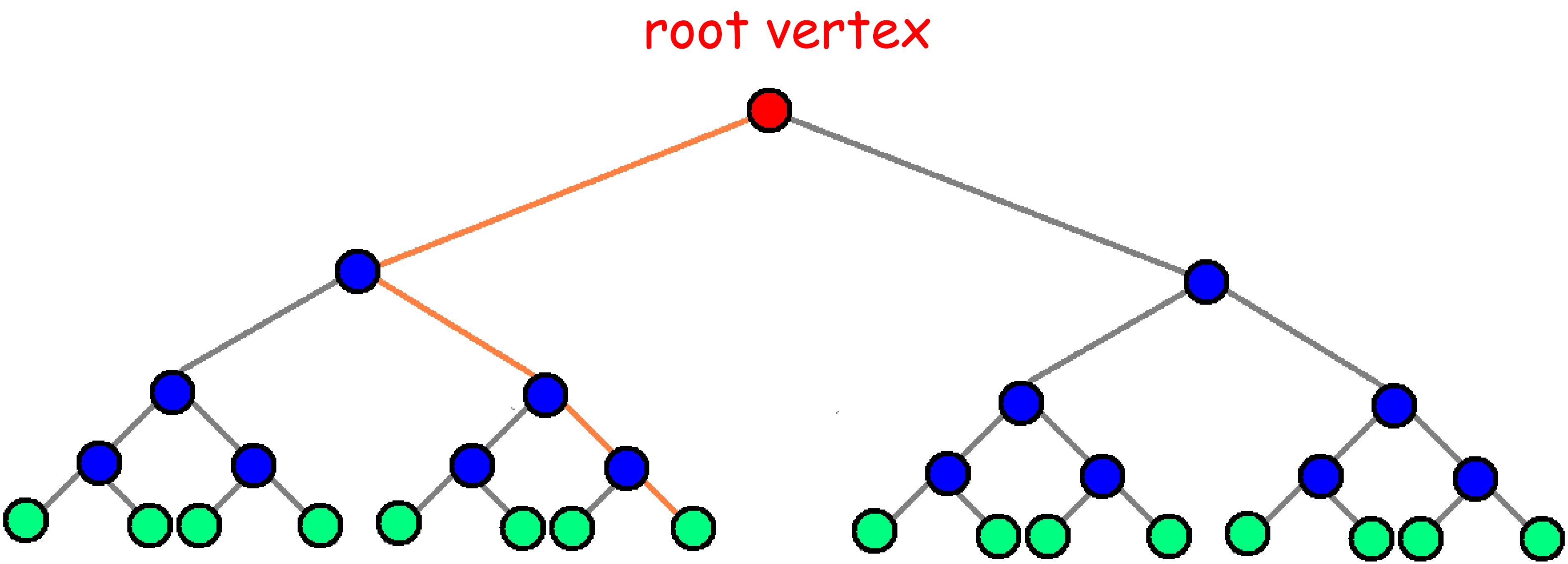
Height 1



Height 2

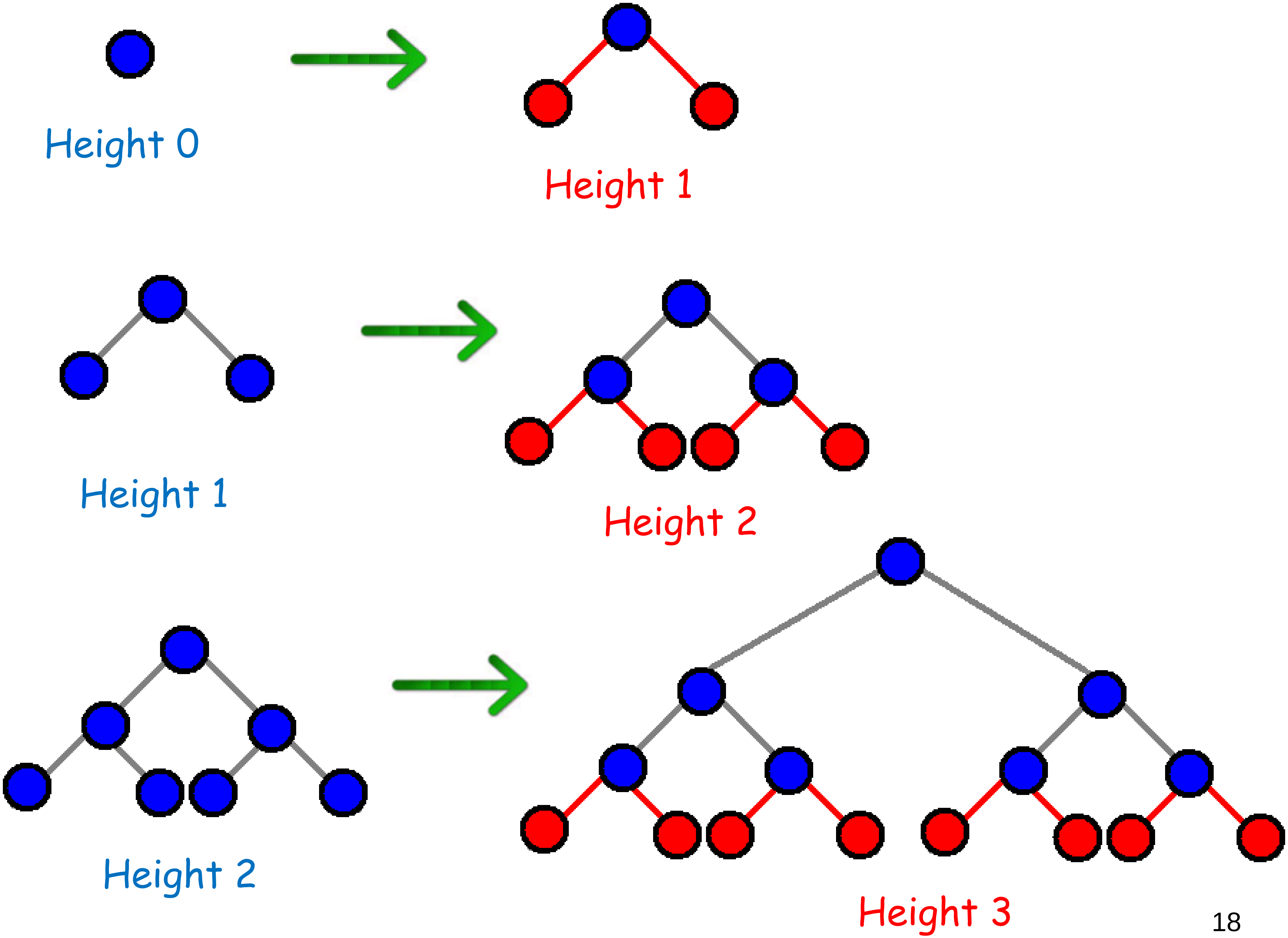


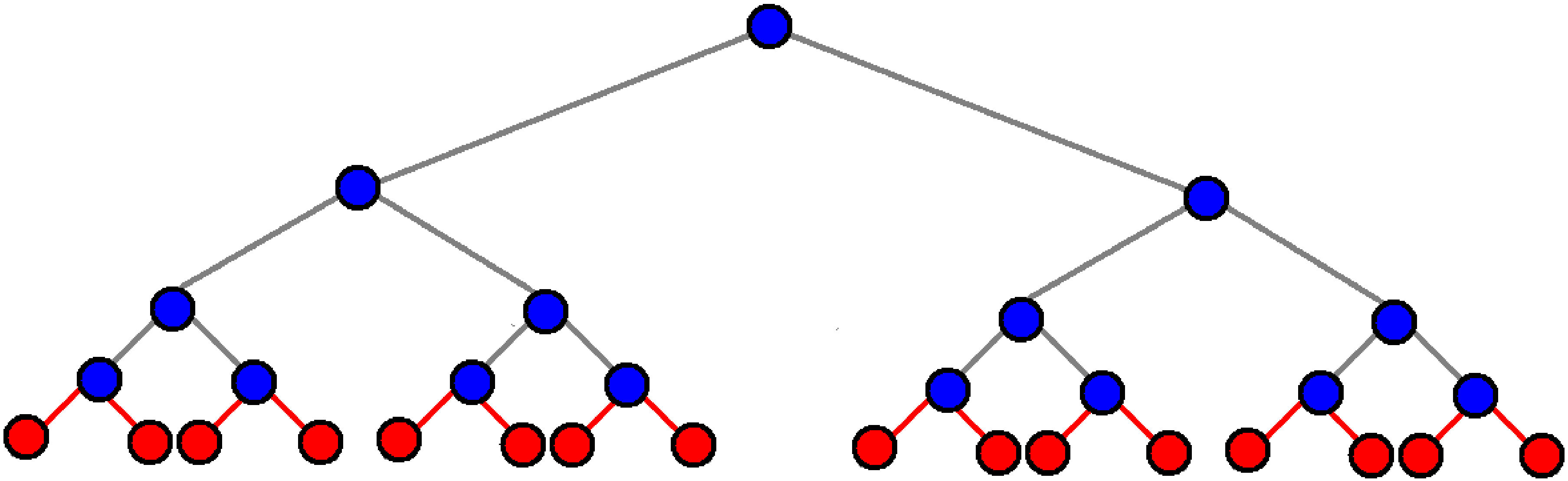
Height 3

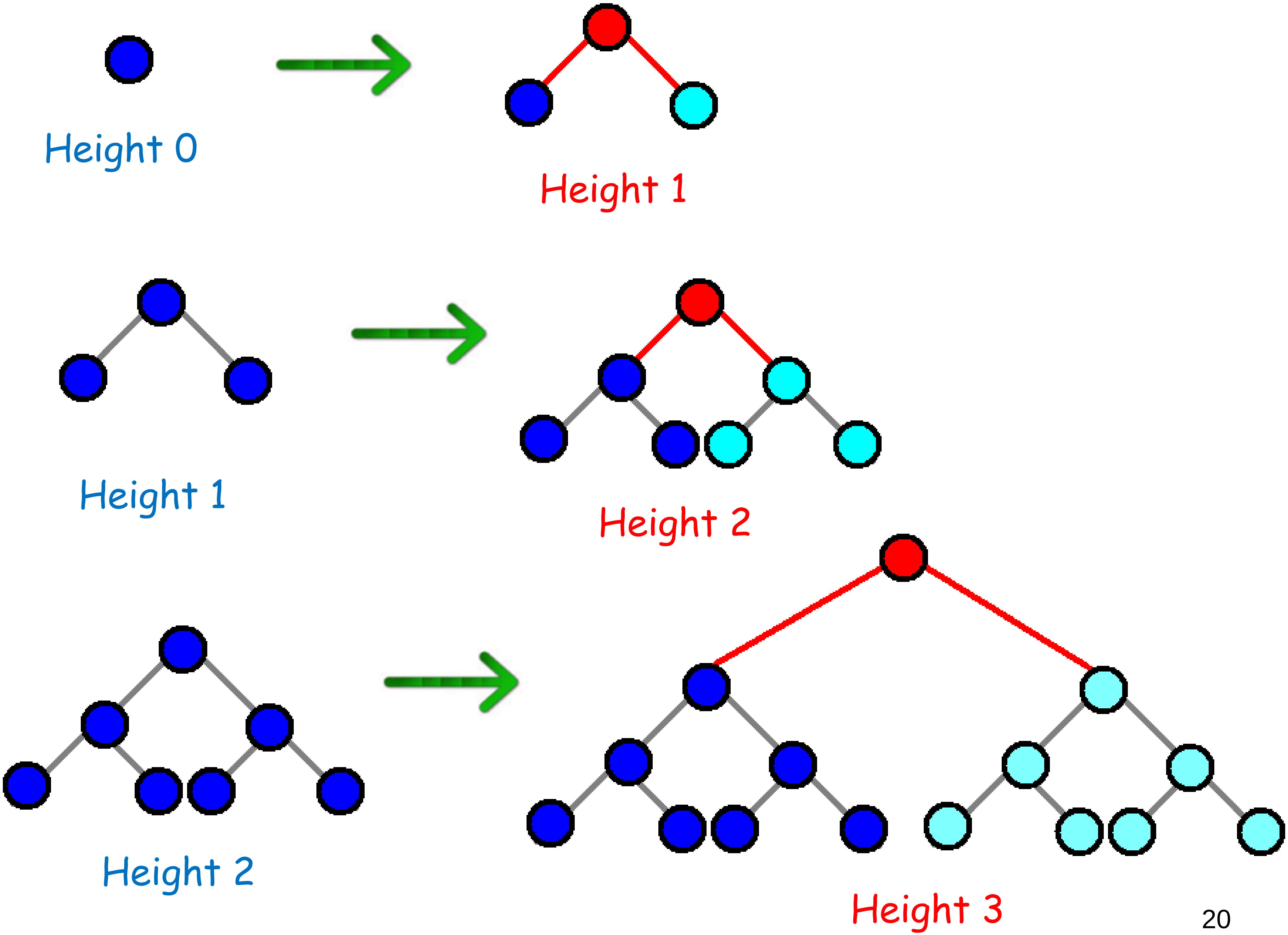


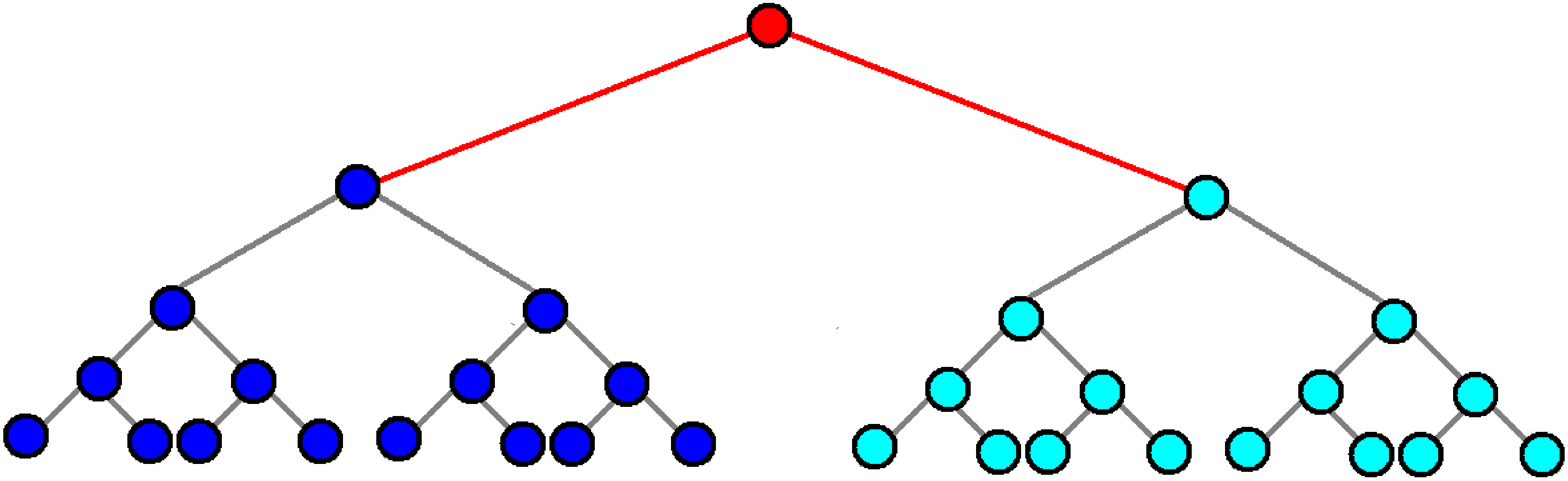
Leaves: vertices with one incident edge

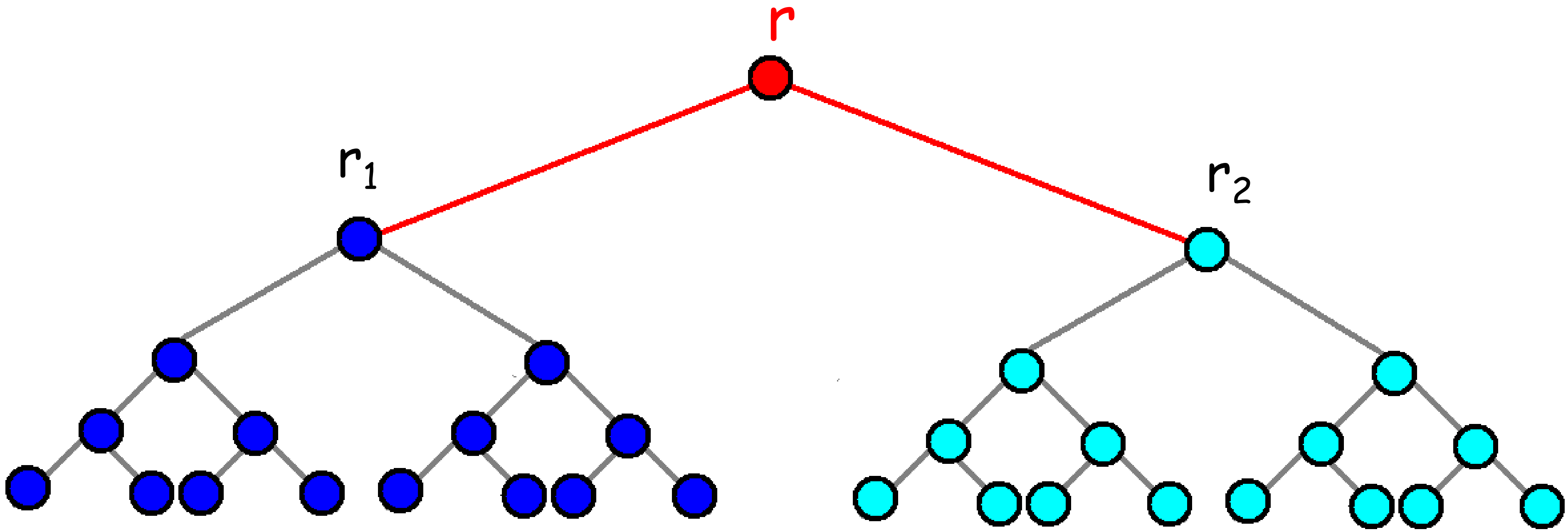
Height: maximum distance from root to a leaf measured by number of edges on the path.


Height 0
Height 1
Height 1
Height 2
Height 2
Height 3




Height 0
Height 1
Height 1
Height 2
Height 2
Height 3




r
r1
r2

How can we give an inductive definition of a complete binary tree of height h?

How many nodes does a complete binary tree of height h have?

Prove the answer is correct by induction.

Create a recurrence relation T(h) where T(h) gives the number of nodes of a complete binary tree of height h.

Solve your recurrence relation and prove the answer is correct by induction.