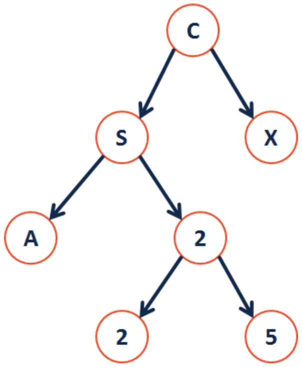
CS 225

#14: Tree Proof – NULLs in a Tree

February 16, 2018 · *Wade Fagen-Ulmschneider*

Definition: Binary Tree



A *binary tree* **T** is:

$T = \{ d, T_L, T_R \}$ *or* $T = \{\}$

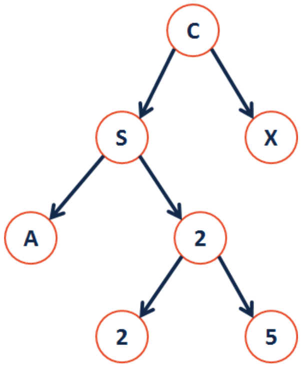
The height of a tree **T** is:

If T = {}, height(T) = -1

Otherwise:

height(T) = 1 + max(height(T_L), height(T_R))

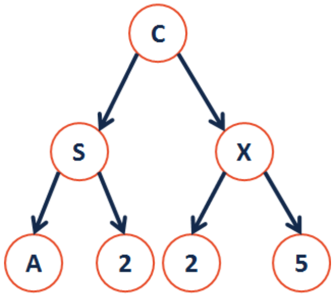
Tree Property: Full



1. F = { }

2. F = { R, F_l, F_r}

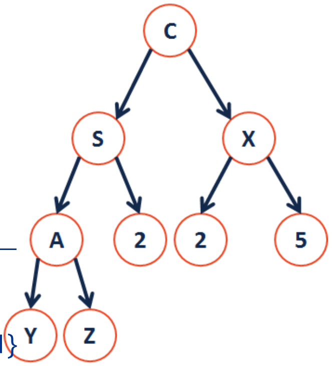
Tree Property: Perfect



Perfect tree is defined by the height of the tree

Conceptually: A perfect tree for every level except the last, where the last level if "pushed to the left".

Tree Property: Complete



A complete tree C of height h, C h :

1. C -1 = {}

2. C h (where h>0) = {r, T L , T R } and either:

T L is __C_{h-1}___ and T R is __P_{h-2}________

OR

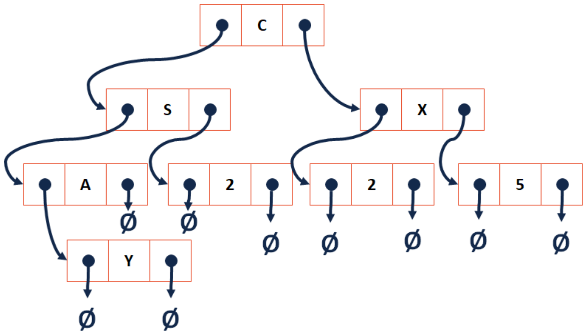
T L is ___P_{h-1}________ and T R is _____C_{h-1}_____

Tree Class

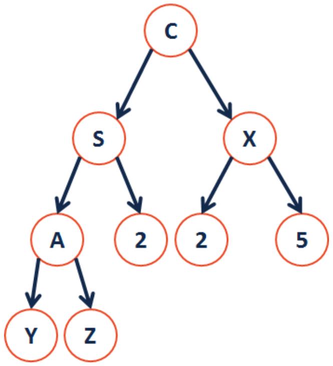
BinaryTree.h

```
#ifndef BINARYTREE_H_
#define BINARYTREE_H_

template <typename T>
class BinaryTree {
  public:

    /* ... */

  private:







};
#endif
```

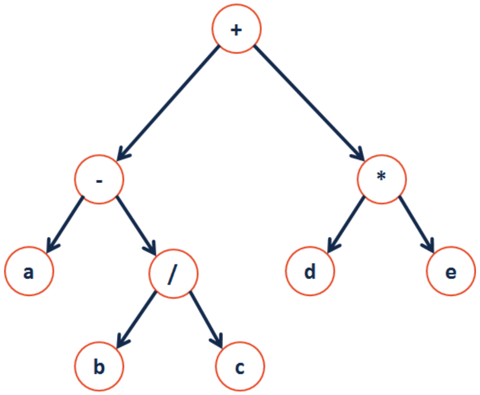
Trees are nothing new – they're fancy linked lists:



Theorem: If there are n data items in our representation of a binary tree, then there are ____________ NULL pointers.



Traversals:



CS 225 – Things To Be Doing:

1. Programming Exam A is on-going (final day is today!)
2. MP3 extra credit deadline is Monday!
3. lab_quacks due Sunday
4. Daily POTDs