Next program is out

Announcement

CS 180 - Mane Trees

- Test the week after next
- Post Order
- In Order
- Pre Order

Different ways depending on what is stored in a tree.

How to display or check in preorder?

I need traversal.
Ex: depth in a tree

Also

Ex: depth in a tree

For each child of V:
Perform "visit" at V

Preorder (T, V)?
Ex: Height in a tree

perform cousin(n, L)
postorder(T[n])
for each child w of n:
postorder(T[w])
Example: Size of a directory

- Need to know size of children before we can compute current directory's size.

10 + 3 + 2 = 15

10K (homework)  2K (CS 145)  3K (examples)

1K (CS 180)  1K (Student)

3K (examples)  1K (CreditCard)

150 total names - each 15K
\[-((3 + 1) \times 3)/(9 - 5) + 2)\]
Complete: ALL

Same depth
Leaves have

Binary Trees

Each internal node has exactly 2 children
Note: $\text{levels} = 2^{l} - 1$ nodes

Complete Binary Tree

Relevant equations on binary trees:

- $\text{If v is right}$
  - $\text{children of } v = 2 \times \text{children of } u$
  - $\text{If v is left}$
  - $\text{children of } v = \text{children of } u$

- $\text{If v is the root, } p(v) = 1$
- $\text{If v is the root, level numbering}$

$\text{Leaf number}$
What if not complete?
```
#include <iostream>

int main()
{
    int x = 5;
    std::cout << "The value of x is: " << x << std::endl;
    return 0;
}
```
Represent tree

Next line 1

Disadvantage: If the tree is complete this
- it is more space efficient
- fast

Advantage: For array based:
3

Node (C) : element (Object)

Struct Node

Alternative: Linked Structure (for binary)
Key = Column labeled Person
Values = Names of people

Ex: Standby list for a flight

Sorted based on those
date shared

Keys versus Values

Priority Queue ADT (Ch. 7)
\( R = \{ (x, y) \in A \times A \mid x \sim y \} \)

- Anti-symmetric: \( x \sim y \) and \( y \sim x \)

- Transitive: \( x \sim y \) and \( y \sim z \) imply \( x \sim z \)

- Reflexive: \( x \sim x \) for all \( x \in A \)

Then

Properties: Need to be able to compare

A note about ties:
Next time - code these w/ en any

- min_element(): returns the smallest element
- remove_min(): removes element with minimum key
- insert_first(): inserts the element

Methods:

- &A
- A