Questions...

1. What is a call stack?

2. Could you write a compiler that did not use one?

3. What data structure does a printer queue use?

Sparse Matrices

- Sparse matrices

- What does this remind us of?
- How could we represent it?

Lists of Lists

- LISP
  - Programming used in AI, math, functional programming
  - Lists (of lists)

\[
\text{add} \quad \text{sqrt} \quad \text{16} \quad \text{d} \quad \varnothing
\]

\[
\text{add} \quad \text{sqrt} \quad \text{16} \quad \varnothing
\]

Other Data Structures for Lists

- Doubly Linked List - when useful?

- Circular List - when useful?

Why Do We Need Trees?

- Lists, Stacks, and Queues represent linear sequences
- Data often contain hierarchical relationships that cannot be expressed as a linear ordering
  - File directories or folders on your computer
  - Moves in a game
  - Employee hierarchies in organizations and companies
  - Family trees
  - Classification hierarchies (e.g. phylum, family, genus, species)
Tree Jargon

- Basic terminology:
  - nodes and edges
  - root
  - subtrees
  - parent
  - children, siblings
  - leaves
  - path
  - ancestors
  - descendants
  - path length

Note: Arrows denote directed edges
Trees always contain directed edges but arrows are often omitted.

More Tree Jargon

- Length of a path = number of edges
- Depth of a node N = length of path from root to N
- Height of node N = length of longest path from N to a leaf
- Depth and height of tree = ?

Definition and Tree Trivia

Recursive Definition of a Tree:
A tree is a set of nodes that is
a. an empty set of nodes, or
b. has one node called the root from which zero or more
trees (subtrees) descend.
- A tree with N nodes always has ___ edges
- Two nodes in a tree have at most how many paths between
  them?
- Can a non-zero path from node N reach node N again?
- Does depth of nodes in a non-zero path increase or
decrease?

Definition and Tree Trivia

Recursive Definition of a Tree:
A tree is a set of nodes that is
a. an empty set of nodes, or
b. has one node called the root from which zero or more
trees (subtrees) descend.
- A tree with N nodes always has N-1 edges
- Two nodes in a tree have at most one path between them
- Can a non-zero path from node N reach node N again?
  - No! Trees can never have cycles.
- Does depth of nodes in a non-zero path increase or
decrease?
  - Depth always increases in a non-zero path

Implementation of Trees

- Obvious Pointer-Based Implementation: Node with value and
  pointers to children
  - Problem: Do not usually know number of children for a node in advance.
  - How many pointers should we allocate space for?
- Better Implementation: 1st Child/Next Sibling Representation
  - Each node has 2 pointers: one to its first child and one to next sibling
  - Can handle arbitrary number of children
  - Exercise: Draw the representation for this tree…

Application: Arithmetic Expression Trees

Example Arithmetic Expression:
A + (B * (C / D ))

How would you express this as a tree?
**Application: Arithmetic Expression Trees**

Example Arithmetic Expression:

\[ A + (B \times (C / D)) \]

Tree for the above expression:

- Used in most compilers
- No parenthesis needed – use the tree structure
- Can speed up calculations e.g. replace / node with C/D if C and D are known
- Calculate by traversing the tree (how?)

---

**Traversing Trees**

- **Preorder**: Root, then Children
  - \( + A \times B / C D \)
- **Postorder**: Children, then Root
  - \( A B C D / + \)
- **Inorder**: Left child, Root, Right child
  - \( A + B \times C / D \)

---

**Example Code for Recursive Preorder**

```c
void print_preorder (TreeNode * T)
{ 
    if (T == NULL) return;
    else { print_element(T->Element);
             P = T->FirstChild;
             while (P != NULL) {
                print_preorder (P);
                P = P->NextSibling;
             }
    }
}
```

What is the running time for a tree with \( N \) nodes?

---

**Preorder Traversal with a Stack**

```c
void Stack_Preorder (TreeNode * T, Stack S)
{ 
    if (T == NULL) return; else push(T,S);
    while (!isempty(S)) {
        T = pop(S);
        print_element(T->Element);
        if (T->Right != NULL) push(T->Right, S);
        if (T->Left != NULL) push(T->Left, S);
    }
}
```

What is the running time for a tree with \( N \) nodes?

---

**Alternative: Nested List Implementation of a Tree**

```c
data next

```

---

**How To Represent?**
How To Represent?

Recursive Preorder for Nested List Implementation

Determining Type of a Node

class node {
    public: enum Tag { I, P };  
    private:
        union { int i; node * p; }; 
    Tag tag;  
    void check(Tag t){ if (tag!=t) error();}; 
    public:
        Tag get_tag() { return tag; }    
        int & ival() { check(I); return i; }    
        node * & pval() { check(P); return p; }
}

Creating and Setting Nodes

class node {  
    ... 
    public:
        // Creating a new node  
        node(int ii) { i=ii; tag=I; }  
        node(node * pp) { p=pp; tag=P; }  
        // Changing the value in a node  
        void set(int ii) { i=ii; tag=I; }  
        void set(node * pp) { p=pp; tag=P; }  
    };

Binary Trees

- Every node has at most two children
  - Most popular tree in computer science
- Given N nodes, what is the minimum depth of a binary tree?
- What is the maximum depth of a binary tree?

Binary Trees

- Every node has at most two children
  - Most popular tree in computer science
- Given N nodes, what is the minimum depth of a binary tree?
  - At depth d, you can have N = 2^d so 2^d-1 nodes (a full tree)
  - So, minimum depth d is: log N ≤ d ≤ log(N+1)-1 or #log N
- What is the maximum depth of a binary tree?
  - Degenerate case: Tree is a linked list!
  - Maximum depth = N-1
- Goal: Would like to keep depth at around log N to get better performance than linked list for operations like Find.
Coming Up

- Read Chapter 4
- Analysis of Binary Search Tree Operations
- AVL, Splay, and Balanced Trees