Recursion in a Nutshell

Clare So

`socm@mcmaster.ca`

SE2AA4 Tutorial. McMaster University. March 21, 2007.

Recursion: It’s all about me!

- A function that calls **itself** as part of its definition

- “To iterate is human; to recurse divine” – L. Peter Deutch

- Mathematical examples
  - Fibonacci numbers

$$F(n) = \begin{cases} 0 & n = 0 \\ 1 & n = 1 \\ F(n-1) + F(n-2) & n > 1 \end{cases}$$

  - Factorial

$$n! = \begin{cases} 1 & n = 1 \\ n \times (n-1)! & n > 1 \end{cases}$$

Fun Example: Open Russian Nested Dolls

More About Recursion

- A recursive function usually consists of
  - Base case
  - A function call of itself
- Why recursion?
  - Elegant solutions
  - May be easier to understand
- Drawbacks of recursion
  - Usually slower (becuase of stacking up call frames)
  - Tail recursion can help

Recursion in ADTs: Binary Tree

- A node in the binary tree contains data, left child and right child
- The childrens of a node are nodes as well
  (recursive definition)
- The beginning of a tree node class may look like

```
public class TreeNode implements Comparable {

    private Comparable contents;
    private TreeNode myLeft;
    private TreeNode myRight;
```

Recursion in ADTs: Binary Tree

- We can find out the height/depth of a binary tree by recursion

```
public int depth() {
   return depthHelper(0, root);
}

private int depthHelper(int depth, TreeNode node) {
   int currentDepth, leftDepth, rightDepth;
   if (node!=null) {
      currentDepth = depth+1;
      leftDepth = depthHelper(currentDepth, node.getLeft());
      rightDepth = depthHelper(currentDepth, node.getRight());
      return Math.max(leftDepth, rightDepth);
   }
   else
      return depth;
}
```

Recursion in ADTs: Binary Tree

- Other recursive algorithms for binary tree include
  - Count the number of nodes
  - Pre-order/In-order/Post-order traversal
  - Insert a new node

In-Class Activity: Binary Tree

- Let’s insert the names of the students into a binary tree
- We can calculate the height etc