CS171 Introduction to Computer Science II

Recursion

Li Xiong
What we have learned so far

• Basic data structure
  – Arrays
  – Linked list

• Abstract data types
  – Stacks
  – Queues
Linked List

- A Linked List is a sequence of nodes chained together.
- Each **node**, element, or link contains a **data item**, and a **reference** to next node.
Stacks and Queues

Stack. Examine the item most recently added. LIFO = "last in first out"

Queue. Examine the item least recently added. FIFO = "first in first out"

• Can be implemented by both (resizing) arrays and linked list
Today

• Quiz on stacks, queues, linked list
• Recursion
Recursion

• Recursion concept
• Examples
  – Factorial
  – Fibonacci
  – GCD
  – Recursive graph Htree
• Next lecture
  – Divide and conquer
  – Binary search
  – Tower of Hanoi
  – Cost analysis of recursive algorithms
Did you mean: recursion

**Recursion - Wikipedia, the free encyclopedia**

Recursion is the process of repeating items in a self-similar way. For instance, when the surfaces of two mirrors are exactly parallel with each other the nested ...

[Formal definitions of recursion](http://en.wikipedia.org/wiki/Recursion)

**Recursion (computer science) - Wikipedia, the free encyclopedia**

Recursion in computer science is a method where the solution to a problem depends on solutions to smaller instances of the same problem. The approach can ...

**Recursion -- from Wolfram MathWorld**
[mathworld.wolfram.com](http://mathworld.wolfram.com) > ... > Algorithms > Recursion

A recursive process is one in which objects are defined in terms of other objects of the same type. Using some sort of recurrence relation, the entire class of ...
What is recursion?
What is recursion?  When one function calls itself directly or indirectly.

Why learn recursion?
- New mode of thinking.
- Powerful programming paradigm.

Many computations are naturally self-referential.
- Mergesort, FFT, gcd, depth-first search.
- Linked data structures.
- A folder contains files and other folders.

Closely related to mathematical induction.
Factorial

N! = N*(N-1)*(N-2)*…..* 2 * 1

int fact (int N)
{
    if (N==0)
        return 1;
    else
        return (N * fact (N-1));
}
Recursive Method

• A method that calls itself (direct recursion)

```java
void recursiveMethod() {
    ... ...
    recursiveMethod();
}
... ...
```
Recursive Method

- A method that calls itself (direct recursion)
- Every recursive method must have a base case that is not recursive

```java
void recursiveMethod() {
    ...
    if (base case) {
        ...
    }
    else {
        ...
        recursiveMethod();
        ...
    }
}
```
Better version of recursion definition

Recursion

n. If you still don't get it, see Recursion.
Recursion

• A method calls itself
  – Calls a “clone” of itself to solve a smaller problem
  – Buck Passing

• Must have a base case
  – The buck stops here! (does not call the method)
Example: Fibonacci Numbers

- Recursive formula:
  \[ F(n) = F(n-1) + F(n-2) \]
  \[ F(0) = 0, \quad F(1) = 1 \]

- 0, 1, 1, 2, 3, 5, 8, 13, ....
Fibonacci Numbers: Java Code

```java
int F(int n) {
    if (n==0)
        return 0;
    else if (n==1)
        return 1;
    else
        return F(n-1)+F(n-2);
}
```
Greatest Common Divisor

\textbf{Gcd.} Find largest integer that evenly divides into p and q.

\textbf{Ex.} \ gcd(4032, 1272) = 24.

\begin{align*}
4032 & = 2^6 \times 3^2 \times 7^1 \\
1272 & = 2^3 \times 3^1 \times 53^1 \\
gcd & = 2^3 \times 3^1 = 24
\end{align*}

\textbf{Applications.}

- Simplify fractions: \(1272/4032 = 53/168\).
- RSA cryptosystem.
Greatest Common Divisor

**Gcd.** Find largest integer \( d \) that evenly divides into \( p \) and \( q \).

**Euclid's algorithm.** [Euclid 300 BCE]

\[
gcd(p, q) = \begin{cases} 
  p & \text{if } q = 0 \\
  gcd(q, p \mod q) & \text{otherwise}
\end{cases}
\]

- base case
- reduction step, converges to base case

\[
gcd(4032, 1272) = gcd(1272, 216) = gcd(216, 192) = gcd(192, 24) = gcd(24, 0) = 24.
\]

\[
4032 = 3 \times 1272 + 216
\]
Greatest Common Divisor

**Gcd.** Find largest integer \( d \) that evenly divides into \( p \) and \( q \).

\[
gcd(p, q) = \begin{cases} 
  p & \text{if } q = 0 \\
  gcd(q, p \mod q) & \text{otherwise}
\end{cases}
\]

- **base case**
- **reduction step, converges to base case**

<table>
<thead>
<tr>
<th>( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( q )</th>
<th>( q )</th>
<th>( p \mod q )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
p = 8x \\
q = 3x \\
gcd(p, q) = x
\]
Greatest Common Divisor

**Gcd.** Find largest integer $d$ that evenly divides into $p$ and $q$.

\[
gcd(p, q) = \begin{cases} 
  p & \text{if } q = 0 \\
  gcd(q, p \mod q) & \text{otherwise}
\end{cases}
\]

- base case
- reduction step, converges to base case

Java implementation.

```java
public static int gcd(int p, int q) {
    if (q == 0) return p;
    else return gcd(q, p % q);
}
```

- base case
- reduction step
Visual Recursion
Fractals
Htree

H-tree of order n.

- Draw an H.
- Recursively draw 4 H-trees of order n-1, one connected to each tip.
public class Htree {
    public static void draw(int n, double sz, double x, double y) {
        if (n == 0) return;
        double x0 = x - sz/2, x1 = x + sz/2;
        double y0 = y - sz/2, y1 = y + sz/2;
        StdDraw.line(x0, y, x1, y);
        StdDraw.line(x0, y0, x0, y1);
        StdDraw.line(x1, y0, x1, y1);
        draw(n-1, sz/2, x0, y0);
        draw(n-1, sz/2, x0, y1);
        draw(n-1, sz/2, x1, y0);
        draw(n-1, sz/2, x1, y1);
    }

    public static void main(String[] args) {
        int n = Integer.parseInt(args[0]);
        draw(n, .5, .5, .5);
    }
}
Recursion

• Recursive method
• Examples
  – Factorial
  – Fibonacci
  – GCD
  – Recursive graph Htree
• Next lecture
  – Divide and conquer
  – Binary search
  – Tower of Hanoi
  – Cost analysis of recursive algorithms