COMS 1003: Introduction to Computer Programming in C

Recursion

October 4th 2005
Announcements

• Use the blog
• HW3 and 4 released
• mid-semster reviews
  – me, the class, TAs
Outline

• Review

• Recursion
Recursion

- Doing the same work on smaller instances of a problem
- Circular Definitions
Recursive Procedure

• Base cases
  − condition specifying when to stop recursion

• Recursive Step
  − do some small part of the problem
  − invoke same function on reduced problem
Self-Calling Functions

• Entirely legal, but must handle with care

```c
int a(int x) {
    return a(x);
}
```
Self-Calling Functions

//what do I do?
int a(int x)
{
    return a(x);
}

//what do I do?
int a(int x)
{
    return a(x);
}
Self-Calling Functions

```c
int a(int x) {
    return a(x);
}
```
Self-Calling Functions

```c
int a(int x)
{
    return a(x); //what do I do?
}
```

```c
int a(int x)
{
    return a(x); //what do I do?
}
```

```c
int a(int x)
{
    return a(x); //what do I do?
}
```

```c
int a(int x)
{
    return a(x); //what do I do?
}
```
Self-Calling Functions

```c
int a(int x)
{
    return a(x);
}
```
Self-Calling Functions

//what do I do?
int a(int x)
{
  return a(x);
}

//what do I do?
int a(int x)
{
  return a(x);
}

//what do I do?
int a(int x)
{
  return a(x);
}

//what do I do?
int a(int x)
{
  return a(x);
}

//what do I do?
int a(int x)
{
  return a(x);
}
int a(int x) {
    return a(x);
}

Call Graph
Recursive Example: Factorial

- \( N! = N \times (N-1)! \)
- What if \( N \leq 0 \)?
- E.g.,
  
  4! = 4 \times 3!
  
  3! = 3 \times 2!
  
  2! = 2 \times 1!
  
  1! = 1 \times 0!
  
  0! = 1
Recursive Example: Factorial

/* Function prototype */
long fact(long);
Recursive Example: Factorial

/* Function prototype */
long fact(long);

/* Function definition */
long fact(long n)
{
    //calculate factorial
    //return answer
}
Recursive Example: Factorial

/* Function prototype */
long fact(long);

/* Function definition */
long fact(long n)
{
    /* recursive step */
    return n*fact(n-1);
}
Recursive Example: Factorial

/* Function prototype */
long fact(long);

/* Function definition */
long fact(long n)
{
    /* base cases */
    if(0==n || n==1)
        return 1;
    else
        /* recursive step */
        return n*fact(n-1);
}