

CISC 3115

Tail Recursion

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Discussed
 - Problem Solving using Recursion
 - Recursive math functions
 - Design solutions to recursive math functions using recursion
 - Recursions and Strings
 - Recursive helper method/function
 - Example problems (sorting, searching, directory size, Tower of Hanoi)
- To discuss
 - Concept of tail recursion

Tail Recursion

- A recursive method is said to be *tail recursive* if there are no pending operations to be performed on return from a recursive call.
- Tail recursions can be realized by compiler efficiently.

Tail and Non-tail Recursion: Compute Factorial

Non-tail recursion

```
public static int factorial(int n) {  
    if (n == 0) { // base case  
        return 1;  
    } else { // recursive call or method invocation  
        // non-tail recursion, because we have to  
        // multiple factorial(n-1) by n, a pending  
        // operation  
        return n * factorial(n - 1);  
    }  
}
```

Tail recursion

```
public static int factorial(int n) {  
    return factorial(n, 1);  
}  
  
private static int factorial(int n, int result) {  
    if (n == 0) { // base case  
        return result;  
    } else { // recursive call  
        // tail recursion, no pending operation after  
        // returning from the recursive call  
        return factorial(n - 1, n * result);  
    }  
}
```

Questions

- Concept of tail and non-tail recursions
- Can you identify non-tail/tail-recursive methods in preceding examples?
- Write tail-recursive methods