CS367 Lecture 13

Monday 7 July 2014

Announcements/Reminders:

- HW3 due 11 PM tonight
- HW4 assigned
- HW2, P1 graded. HW2 solutions link on Piazza.
- Midterm on Thursday

Last class:

- Stacks and Queues (finish)
    - Applications (cont'd)
    - `findMax` for stacks and queues
    - Sliding window and subsequence-sum problems

Today:

- (Last) Week in Review
- Recursion

(Last) Week in Review

- Linked Lists
    - `tail` reference and dummy header
    - Variations
    - Adding iterators
- Comparing Complexities with Array-Based List Implementations
- Shadow array improvement
- Stacks and Queues
    - Definitions, ADTs
    - Implementations and Complexities
    - Basic applications
    - Cooler applications

Recursion: Beginning Example

Write a code snippet for a method `print` called on a singly-linked chain of nodes such that `print(head)` prints the contents of the whole chain:

Iterative version:

```
void print(ListNode<String> ptr) {




}
```

Another way:

```
void print(ListNode<String> ptr) {




}
```

Recursion: What and Why

What is it?

Bad joke: “In order to understand recursion, you must first understand recursion.”

Why is it useful?

How does it work?

Rules:

1.

2.

Key Recursion Questions

Questions to keep in mind:

- How can you solve the problem in terms of smaller problems of the same kind?
- What instances of the problem can be used as base cases?
- How does the problem size decrease in each recursive call?
- As the problem size decreases will a base case be reached?

Similarities and Differences with Iteration

Writing Recursive Code

Computing Factorials: Write a method that computes the factorial of `n`:

```
n! = n * (n – 1) * (n – 2) * … * 2 * 1
```

Iterative version:

```
int factorial(int n) {



}
```

Recursive definition of factorial:

Recursive version:

```
int factorial(int n) {



}
```

`displayReverse()` for a chain of nodes:

Recursion Example: n choose k

Conventional Definition:

Recursive Definition:

Implementing the recursive definition:

Tracing an execution tree: