

# CAS CS 112 – Spring 2008

## Practice Midterm Questions

### Problem 1

Write a function `int findMin(Queue q)`, which takes a queue of integers as input, and returns the minimum element in the queue, without changing the order of the elements. You can assume that the queue is non-empty and implements the usual operations (`enqueue()`, `dequeue()`, `isEmpty()`).

### Problem 2

Write a function `void reverseList(Node head)`, which takes the head of a singly-linked list as an argument and reverses the list, such that `head` now points to the beginning of the reversed list. Your answer should run in linear time.

### Problem 3

Write a function `void removeLast(Node head)`, which takes the head of a doubly-linked list as an argument and removes the last element of the list, such that `head` now points to the list with the last element removed.

### Problem 4

State whether each of the following statements is true or false and briefly explain why:

1. The `get(index)` and `set(index,value)` operations of the List ADT can be implemented faster using an array-based list rather than a linked list.
2. The `add(value)` operation of a linked list, which adds the value to the back of the list, can be implemented in constant time if we keep a reference to the back of the list.
3. An array list implementation is preferred if we are doing a lot of inserting into the front and the back of the list, while a linked list implementation is preferred if we are performing a lot of get and set operations.
4. Quicksort always runs in  $O(n \log n)$  while mergesort runs in  $O(n \log n)$  in the average case.
5. Bubble sort and insertion sort are faster than mergesort and quicksort on some inputs.

## Problem 5

Consider an implementation of quicksort which picks the pivot as the first element of the array. Draw a diagram to illustrate how this implementation performs on the following array of numbers: {1, 8, 4, 3, 5, 9, 2, 6, 10, 3, 5, 8, 7}. At each iteration, show how the array is partitioned and the location where the pivot is fixed. You can assume that the recursion bottoms out when the size of the array is 3 or less (at which point another sort (in this case your head) is used), and that elements equal to the pivot go to the left.

## Problem 6

A level-order traversal of a tree visits the nodes top to bottom, left to right. Draw a tree with 5 elements such that its inorder and level-order traversals are the same.

## Problem 7

Identify three bugs in the `ArrayBasedList` method below that inserts an item into an array-based list in sorted order (3 pts per bug). As in class, assume the “a” member variable is the array that stores the items and the “size” member variable stores the number of elements. The fact that array overflow is not tested does not count as a bug. Non-bugs that you identify as bugs will count against you. Line numbers are indicated for your convenience.

```
(1) void BuggyInsert(Item i) {
(2)     int j, k;
(3)     // Find the location of the first item larger
(4)     // than i and store it in variable j. If all
(5)     // items are smaller than i, j gets value "size"
(6)     for (j = 0; j < size || a[j] < i; j++)
(7)         ;
(8)     // Slide items from position j through position
(9)     // size-1 one to the right to make room.
(10)    for (k = size-1; k > j; k--) {
(11)        a[k] = a[k-1];
(12)    }
(13)    // now put i into its correct place
(14)    a[j] = i;
(15) }
```