Review for 311 Midterm:
Recursion
Queue
Stack
BT
BST
AVL
Big Oh and counting

Counting

(1) What is T(n) for insertion sort (Wikipedia):

```
i ← 1
while i < length(A)
     j ← i
     while j > 0 and A[j-1] > A[j]
         swap A[j] and A[j-1]
         j ← j − 1
     end while
     i ← i + 1
 end while
```

(2) What is T(n) for selection sort (Wikipedia):

```
/* a[0] to a[n-1] is the array to sort */
int i,j;
int n; // initialise to a's length

/* advance the position through the entire array */
/*   (could do j < n-1 because single element is also min element) */
for (j = 0; j < n-1; j++)
{
    /* find the min element in the unsorted a[j .. n-1] */

    /* assume the min is the first element */
    int iMin = j;
    /* test against elements after j to find the smallest */
    for (i = j+1; i < n; i++)
    {
        /* if this element is less, then it is the new minimum */
        if (a[i] < a[iMin])
        {
            /* found new minimum; remember its index */
            iMin = i;
        }
    }

    if (iMin != j)
    {
        swap(a[j], a[iMin]);
    }
}
```

(3) Here is pseudo-code to multiply two n X n arrays together . What is T(n)?

```
(A * B = C,
c[i,j] = a[i,0]*b[0,j] + a[i,1]*b[1,j] + … a[i,(n-1)]*b[(n-1),j])

for (i = 0; i < n; i++)
   for (j = 0; j < n; j++)
      C[i, j] = 0;

for (i = 0; i < n; j++)
   for (j = 0; j < n; j++)
      for (k = 0; k < n; k++)
         C[i,j] += (A[i, k] * B[k,j])
```

(4) For each of the following T(n) formulas, give Big Oh

a. $5\ n^2 + 3000\ n \log n + 25$

b. $3000\ n^2 + 5\ n \log n + 25$

c. $1 + n + n^2 + n^3 + n^4$

d. $10 + n + n^2 + n^2 \log n$

e. $10^n + n^5$

f. $15\ n^2 \log n + 15\ n (\log n)^2$

(5) Using a queue,

`insert(x)` means append x to tail
`delete()` means return value from head

Consider an initially empty queue, and these operations in order:

```
insert(1)
insert(2)
insert(3)
delete()
insert(4)
insert(5)
delete()
delete()
insert(6)
```

What is content of queue from head to tail.

(6) Consider a circular queue implemented on a 1D array.

head points to the element at the head of the queue
tail points to the (empty) element where the next incoming element will be placed.
one element in the array is always kept "empty"

The following pseudo-code for delete() and insert() is incorrect. Fix the code.

```
/* test for full */
boolean isFull() {
   return head == (tail + 1) % size;
}

/* test for empty */
boolean isEmpty() {
   return head == tail;
}

/* insert x at tail */
void insert(x) {
   if ( !isFull() )
      tail = tail + 1;
      q[tail] = x;
   else error();
}

/* delete from head */
type delete() {
   if ( !isEmpty() ) {
    return q[head];
    head = head + 1 % size;
   return error();
}
```

(7) Consider this string "a b c c b d e". Use a stack to determine if the string is a palindrome. Give the content of the stack after the second 'c' has been read and processed.

(7*) Something similar for any of the stack algorithms (evaluate postfix, infix → postfix, delimiter matching, …)

(8) Consider this infix expression: ((1 + 2) * 3) / (4 – 5).

Give (a) postfix, (b) infix, (c) expression tree.

(9) Consider this binary tree.

```
a
       b
              c
                     d
                            NULL
                            NULL
                     NULL

              e
                     NULL
                     NULL


       f
              NULL
              g
                     h
                            i
                                   NULL
                                   NULL

                            NULL

                     NULL
```

Give (a) inorder, (b) postorder, (c) preorder, and (d) breadth traversals.

(10) For the binary tree given above, give height and depth at every non-NULL node.

(11) What is the binary search tree obtained by inserting the following data in order shown:
50, 20, 10, 30, 80, 100, 110, 90, 85. Call this tree TREE

(12) From TREE, show the resulting tree from deleting 80.

(13) From TREE, show the resulting tree from deleting 50.

(14) From TREE, show the OTHER resulting tree from deleting 50.

(15) What is the AVL tree obtained by inserting the following data in order shown:
50, 20, 10, 30, 80, 100, 110, 90, 85.

(16) Consider the quad.

```
int quad (int n) {
   if  (n <= 0) return 1;
   if (n == 1) return 1;
   if (n == 2) return 2;
   if (n == 3) return 3;
   return quad (n – 3) * quad (n – 2);
}
```

(a) What is the result for quad(8)?

(b) How many times is quad() invoked when called with quad(8) (include the quad(8) invocation)?

(c) Suppose you are memo-izing. Give the memo table after you return from a call to quad(6).