Stacks and Queues

1. Suppose that items 10, 20, 30, 40, 50 are enqueued, in that order, onto an initially empty queue $Q$. Then a sequence of four dequeue operations are performed on $Q$; as each item is dequeued, it is inserted into an initially empty stack $S$. If two items are then popped off the stack $S$, what is the item to be returned by the next pop operation?

2. Suppose you have two objects $P$ and $Q$, from a class `Queue` with standard member functions: `enqueue`, `dequeue`, `isempty`, and `size`. How would you implement the two stack operations `pop` and `push`? Give the running time of each operation.

3. What is the output of the following code?

   ```
   int values[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
   std::stack<int> mystack;
   for (int i = 0 ; i < 10 ; i ++)
       mystack.push(values[i]);
   int n = 20;
   for (int i = 0 ; i < 5 ; i ++)
       n += mystack.pop();
   for (int i = 8 ; i >= 1 ; i /= 2)
       n -= mystack.pop();
   std::cout << n;
   ```

4. Suppose you are given an array $A$ containing $n$ numbers in order. Describe in pseudocode an efficient algorithm for reversing the order of the numbers in $A$ using a single for-loop that indexes through the cells of $A$, to insert each element into a stack, and then another for-loop that removes the elements from the stack and puts them back into $A$ in reverse order. What is the running time of this algorithm?

5. Solve the previous exercise using a queue instead of stack.
Linked Lists

1. Describe, in pseudocode, a link-hopping method for finding the middle node of a doubly linked list with head and tail pointers, and an odd number of nodes. Note that you should only use link-hopping (cannot use a counter).

2. Describe an implementation of void insertBefore(Node *p, int val), for a doubly linked list. The method should insert a new node with the element val before the node pointed by p.

3. Assuming efficient implementation of linked-lists (LL), indicate the worst-case performance of each of the following operations (in terms of n):

<table>
<thead>
<tr>
<th>Operation</th>
<th>Singly LL</th>
<th>Doubly LL</th>
<th>Circular SLL</th>
<th>Circular DLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>getFirst()</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>getLast()</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>prepend(val)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>append(val)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>insertAt(index, val)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>removeFirst()</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>removeLast()</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>removeAt(index)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>contains(val)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

OBS: Consider that you can only have the pointer head in your implementation.

Trees

1. Let T be a tree with more than one node.

(a) Is it possible that a preorder traversal visits the nodes of T in the same order as a postorder traversal? If so, provide an example, otherwise argue why this cannot occur.

(b) Is it possible that a preorder traversal visits the nodes of T in the reverse order as a postorder traversal? If so, provide an example, otherwise argue why this cannot occur.
2. Let $T$ be a full $k$-ary tree, where $k = 2$ (a.k.a. binary tree), with $n$ nodes. Let $h$ denote the height of $T$.

   (a) What is the minimum number of leaves for $T$? Justify your answer.
   
   (b) What is the maximum number of leaves for $T$? Justify your answer.
   
   (c) What is the minimum number of internal nodes for $T$? Justify your answer.
   
   (d) What is the maximum number of internal nodes for $T$? Justify your answer.

3. Give a $O(n)$ time algorithm for computing the depth of all the nodes of a tree, where $n$ is the number of nodes.

4. Show that the maximum number of nodes in a binary tree of height $h$ is $2^{h+1} - 1$.

5. Considering the definition of height of a binary tree, give a formula for:

   (a) the maximum height of a binary tree with $n$ nodes?
   (b) the minimum height of a binary tree with $n$ nodes?
   (c) the maximum number of nodes in a binary tree of height $h$?
   (d) the minimum number of nodes in a binary tree of height $h$?