

CSCE 221 Cover Page  
Homework Assignment #2  
Due March 27 at 23:59 pm to eCampus

First Name Joseph Last Name Martinsen UIN 323006691

User Name josephmart E-mail address josephmart@tamu.edu

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero, read more Aggie Honor System Office <http://aggiehonor.tamu.edu/>

|                         |  |  |  |
|-------------------------|--|--|--|
| Type of sources         |  |  |  |
| People                  |  |  |  |
| Web pages (provide URL) |  |  |  |
| Printed material        |  |  |  |
| Other Sources           |  |  |  |

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.

“On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work.”

Your Name

Date

## Homework 2

due March 27 at 11:59 pm to eCampus.

1. (15 points) Write a recursive function that counts the number of nodes in a singly linked list. Write a recurrence relation that represents your algorithm. Solve the relation and obtain the running time of the algorithm in Big-O.

(a) 

```
int nodeCounter(Node* n)
{
    if(temp == nullptr) { return 0; }

    return nodeCounter(n->next()) + 1;
}
```

(b)  $\text{nodeCounter}(0) = 0$   
 $\text{nodeCounter}(n) = \text{nodeCounter}(n-1) + 1$

(c) Solving the relation:  $\text{nodeCounter}(n) = n$   
The Big-O :  $O(n)$

2. (15 points) Write a recursive function that finds the maximum value in an array of int values without using any loops. Write a recurrence relation that represents your algorithm. Solve the relation and obtain the running time of the algorithm in Big-O.

(a) 

```
int maxArray(int a[], int length)
{
    if (length == 1)
        return a[0];

    int maxNext = maxArray(a, length-1);

    if (a[length-1] > maxNext)
        return a[length-1];
    return maxNext;
}
```

(b)  $\text{maxArray}(\text{array}, \text{size}) = \text{maxArray}(\text{array}, \text{size}-1)$   
 $\text{maxArray}(\text{array}, 1) = a[0]$

(c) Solving the relation:  $\text{maxArray}(\text{array}, 1) = a[n]$   
The Big-O :  $O(n)$

3. (10 points) What data structure is most suitable to determine if a string  $s$  is a palindrome, that is, it is equal to its reverse. For example, “racecar” and “gohan gasalamiimalasagnahog” are palindromes. Justify your answer. Use Big-O notation to represent the efficiency of your algorithm.

A queue would be the most well suited ADT to determine if a string was a palindrome. The first half of the string would be enqueued. Due to the last in first out nature of a queue, by dequeuing and comparing the dequeued item to the next item in the string, each comparison should return true if the string is in fact a palindrome.

It would be  $n$  for queuing,  $2n$  for dequeuing and comparing. This results in  $3n$  which is  $O(n)$

4. (10 points) Describe how to implement the stack ADT using two queues. What is the running time of the push and pop functions in this case?

5. (10 points) What is the best, worst and average running time of quick sort algorithm? Provide arrangement of the input and the selection of the pivot point at every case. Provide a recursive relation and solution for each case.

Best and average case of quick sort is  $O(\log_2(n))$ . Worse case is  $O(n^2)$

6. (10 points) What is the best, worst and average running time of merge sort algorithm? Use two methods for solving a recurrence relation for the average case to justify your answer.

7. (10 points) R-10.17 p. 493

For the following statements about red-black trees, provide a justification for each true statement and a counterexample for each false one.

- (a) A subtree of a red-black tree is itself a red-black tree.  
False. A subtree with a red root must have a black root node
- (b) The sibling of an external node is either external or it is red.

True. Suppose an external node had a black, internal sibling and let  $h$  be the black depth of the external node. Then the black depth of the external nodes below the black, internal sibling would have black depth at least  $h+1$  thereby violating the depth property of a red-black tree. The depth property says that all external nodes must have the same black depth, that is, the same number of black ancestors.

- (c) There is a unique (2,4) tree associated with a given red-black tree.  
True.

The given one means that Given a red-black tree  $T$ , there is a unique 2-3-4 tree  $T_0$  associated with  $T$ . Every node in the red-black tree with two red children is uniquely represented as a 4-node in  $T_0$ , each node with one red child is uniquely represented as a 3-node in  $T_0$ , and a node with no red children is uniquely represented as a 2-node in  $T$

- (d) There is a unique red-black tree associated with a given (2,4) tree.  
False. A 3-node has two different possible ways of being shown in a red-black tree.

8. (10 points) R-10.19 p. 493

Consider a tree  $T$  storing 100,000 entries. What is the worst-case height of  $T$  in the following cases?

- (a)  $T$  is an AVL tree.
- (b)  $T$  is a (2,4) tree.
- (c)  $T$  is a red-black tree.  
99,999
- (d)  $T$  is a binary search tree.  
99,999

9. (10 points) R-9.16 p. 418

Draw an example skip list that results from performing the following series of operations on the skip list shown in Figure 9.12: `erase(38)`, `insert(48,x)`, `insert(24,y)`, `erase(55)`. Record your coin flips, as well.