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	AVL Tree Info		
People			
Web pages (provide URL)	http://pages.cs.wisc.edu/ ealexand/cs367/NOTES/AVL-Trees/index.html		
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 Joseph Martinsen Date 03/27/2017

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) nodeCounter(0) = 0
   nodeCounter(n) = nodeCounter($n-1$) + 1

   (c)

   $$\begin{aligned} T(n) \\ T(n-1) \\ T(n-2) \\ \cdots \\ T(0) \\ 0 \end{aligned}$$

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

   (c) Solving the relation: maxArray($array$, 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 que would be the most well suited ADT to determine if a string was a palindrome. The first half of the string would be enqued. Due to the last in first out nature of a que, by dequing and comparing the dequed 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 queing, $2n$ for dequing 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?

   The two queues will be referred to as Q1 and Q2. Q1 is used to store values and Q2 is used to help Q1 be a stack. The $isEmpty()$ and $size()$ function would just call the same function on Q1. Pushing an item would just be enqueueing that said item into Q1. This takes $O(1)$ time. Popping an item would require all the items in Q1 to be dequeued and enqueued into Q2 until the last element in Q1 is reached. This would be the item that would be popped off. In order to return back to the regular state, all items in Q2 would be dequeued and enqueued back into Q1.

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.

   Worse case is $O(n^2)$

$$
\begin{aligned}
T(n) &= T(n-1) + n - 1 \\
T(n) &= T(n-2) + n - 1 + n - 2 \\
T(n) &= T(n-3) + 3n - 1 - 2 - 3 \\
T(n) &= T(1) + \sum_{i=0}^{n-1} (n-i) \\
T(n) &= \frac{n(n-1)}{2} \\
T(n) &= \frac{n^2 - n}{2} \\
&O(n)
\end{aligned}
$$

   Best: $O(n \log n)$

$$
\begin{aligned}
T(N) &= 2T(\frac{n}{2}) + n - 1 \\
T(N) &= 2(2T(\frac{n}{4}) + \frac{n}{2} - 1) + n - 1 \\
T(N) &= 4(2(\frac{n}{8}) + \frac{n}{4} - 1) + 2n - 3 \\
T(N) &= 2^k T(\frac{n}{2^k}) + kn - (2^n - 1) \\
T(1) &= 0 \\
2^k &= n \\
k &= \log_2 n \\
T(n) &= n \log_2 n - n + 1 \\
&O(n \log n)
\end{aligned}
$$

Average: $O(n \log n)$

$$
\begin{aligned}
T(n) &= 2T(\frac{n}{2}) + n - 1 \\
T(n) &= 2(2T(\frac{n}{4}) + \frac{n}{2} - 1) + n - 1 \\
T(n) &= 4(2(\frac{n}{8}) + \frac{n}{4} - 1) + 2n - 3 \\
T(n) &= 2^k T(\frac{n}{2^k}) + kn - (2^k - 1) \\
T(1) &= 0 \\
2^k &= n \\
k &= \log_2 n \\
T(n) &= n \log_2 n - n + 1 \\
&O(n \log n)
\end{aligned}
$$

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.

Divide and Conquer Method

$$
\begin{aligned}
a = 2 \quad b = 2 \\
T(1) &= 0 \\
C(n) &= \theta(n) \\
D(n) &= \theta(1) \\
C(n) + D(n) &= \theta(n) \\
T(n) &= 2T\left(\frac{n}{2}\right) + \theta(n) \\
&O(\log_2(n)
\end{aligned}
$$

Tree Method

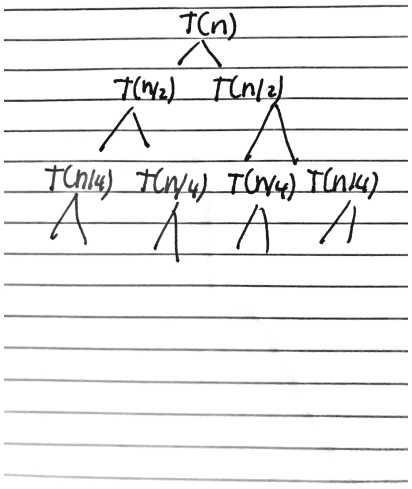

Figure 1:

$O(\log_2(n)$

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. If an external node had a black internal brother or sister, the black depth of the external nodes below the black brother or sister would be $h + 1$ which is incorrect because all external nodes must have the same black height.

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

   A node in the red-black tree that contains two red children is represented as a 4-node in 2-4 tree. A node with one red child is represented as a 3-node in the 2-4 tree. A node with 0 red children is represented as a 2-node in the 2-4 tree.

   (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.

   $$h \leq 2\log_2(n+2)$$
   $$h \approx 33$$

   (b) T is a (2,4) tree.

   $$h < \log_2(n+1)$$
   $$h \approx 17$$

   (c) T is a red-black tree.

   $$h < 2\log_2(n+1)$$
   $$h \approx 33$$

   (d) T is a binary search tree.
   100,000

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.

   Coin flips for x : 4
   Coin flips for 5 : 5

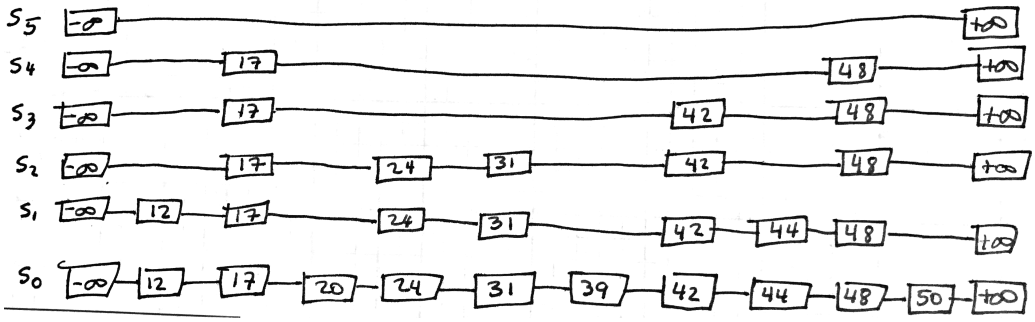

Figure 2: Skip List