CSCE 221 Cover Page

Homework Assignment #2

Name: Josh Zschiesche UIN: 523000614

User Name: jzschiesche1 E-mail address: jzschiesche1@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 on Aggie Honor System Office website: `http://aggiehonor.tamu.edu/`

Type of sources		
People	Todd Christian	Peer TA's
Web pages (provide URL)	cpp reference stack overflow	http://goo.gl/AE0vGR
Printed material	Textbook	Lecture Slides
Other Sources	Ryan Yantz	

I certify that I have listed all the sources that I used to develop the solutions/codes in the submitted work. *On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work.*

Your Name Josh Zschieshce Date March 30, 2016

Homework 2

due March 29 at 11:59 pm.

1. (10 points) Describe (in pseudo code) how to implement the stack ADT using two queues. What is the running time of the push and pop functions in this implementation?

   (a) Implement the stack using ADT using two queues
   q1 and q2 = NULL, q1 elements and q2 as "holder"
   int size
   {
   q1.size()
   } // don't need info about q2
   void push (queue q, object obj)
   {
   q.insert(obj);
   }
   Object pop()
   {
   while (!isEmpty()) { push(q1);
   }
   object obj = q2.dequeue();
   while (!q2.isEmpty()) { q1.enqueue(q2.dequeue());
   }
   }

   (b) Time complexity of push: O(1)

   (c) Time complexity of pop: O(n)

2. (10 points) Solve C-5.8 on p. 224 (Describe non-recursive way of evaluating postfix notation)

   (a) Use two stacks, One stack is the "Operator" - holding the operators while the other is the "Operand" - holding the returned expression

   (b) Begin at the left of the original expression

      i. Place the operands into "Operand" stack
      ii. If an operand is found, push it to the "Operator" stack
      iii. Continue until another operand is found with lower precedence than others before it
      iv. When end of original expression is reached, empty Opeator into Operand

   (c) Example

      i. Infix: A*(B+C*D)+E
      ii. Postfix: ABCD*+*E+

3. (10 points) Linked list questions.

   (a) Write a recursive function in C++ that counts the number of nodes in a singly linked list.
   int countNodes()
   { //Not 100% sure this will compile
   struct node *temp = beginning;
   int count = 0;
   while(temp !=NULL)

{
length++; temp = temp->next;
}
return (length);

(b) Write a recurrence relation that represents the running time for your algorithm.

i. T(n) = T(n-1) + 1, for T(1) = 1

(c) Solve this relation and provide the classification of the algorithm using the Big-O asymptotic notation.

i. O(n)

4. (10 points) Find max value

(a) Write a recursive function that finds the maximum value in an array of integers without using any loops.

int findMax (int const &array[], int temp_max, int temp = 0)
{
if (array[temp] != NULL)
{
if (array[temp] > temp_max) { findMax(array, array[temp], temp++);
}
else { findMax(array, temp_max, temp++); }
}
else
{ return temp_max }
}

(b) Write a recurrence relation that represents running time of your algorithm.

i. T(n) = t(n-1) +1 , for T(1) = 1

(c) Solve this relation and classify the algorithm using the Big-O asymptotic notation.

i. O(n)

5. (10 points) Consider the quick sort algorithm.

(a) Provide an example of the inputs and the values of the pivot point for the best, worst and average cases for the quick sort.

BEST:
Input: | 1 | 3 | 9 | 2 | 6 | 5 | 4 | Pivot: 2, as 2 is in the middle
WORST:
Input: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Pivot: either 1 or 7, as those are on the ends
AVERAGE:
Input: | 1 | 2 | 9 | 4 | 5 | 3 | 7 | Pivot: Neither the middle nor the ends, for example 9

(b) Write a recursive relation for running time function and its solution for each case.

Base Case: T(1) = 1 for all of the cases.
BEST:
T(n) = 2*T(n/2)+n, O(nlog(n))
AVERAGE:
T(n) = n*T(n/2), O(nlog(n))
WORST:
T(n) = T(n-1) +n,$O(n^2)$

6. (10 points) Consider the merge sort algorithm.

   (a) Write a recurrence relation for running time function for the merge sort.

      i. T(n) = 2*T(n/2)+n Base Case: T(1) = 1

   (b) Use two methods to solve the recurrence relation.

      i. Master's Therom

         A. a = 2, b = 2, then $n^{log_2 2} = nlog(n)$, therefore O(nlog(n))

      ii. Recursion Therom

         A. T(n)
         B. Then, 2 * T(n/2) + f(n)
         C. Then, 4* T(n/4) + f(n) + f(n)
         D. Then, 8*T(n/8) + f(n) + f(n) + f(n)
         E. Then, 16*T(n/16) + f(n) + f(n) + f(n) + f(n)
         F. ...
         G. Then, $T(n) = 2^k T(n/2^k) + n$
         H. Then, $T(n) = 2^{log_2(n)} * T(n/2^{log_2(n)}) + n$
         I. Then, k = 1 and substituting the base case
         J. nlog(n)

   (c) What is the best, worst and average running time of the merge sort algorithm? Justify your answer.
   Best : O(nlog(n)), Worst : O(nlog(n)), Average : O(nlog(n))

   In every case, the algorithm continues to split data into smaller and smaller pieces, until it is only single values. Then, it merges them back together. Every time, no matter how well organized the data is, it will be O(nlog(n)), this is because the way that it merges back the algorithms will not

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.

      i. False - black trees cannot also be red trees

   (b) The sibling of an external node is either external or it is red.

      i. True - External nodes are always black and must have equal depth. If the root has one black internal sibling, then it would have to be at least lenght+1 length longer than the original. Therefore, siblings of an external node can't be black.

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

      i. True - each node having two red children must be a four-node tree, nodes with one red child must be a three-node tree, nodes with zero red children must be a two-node tree.

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

      i. False - there are up to three possible combinations of Red/Black trees.

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) AVL tree, log(100,000)

   (b) (2,4) tree, 1.44*log(100,000 + 1)

   (c) Red-black tree, 2*log(100,000)

(d) Binary search tree, 100,000 - 1

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.

   (a) See the pdf labled "Skip_List_Pic" in same directory as this document.

10. (10 points) R-9.7 p. 417

    Draw the 11-entry hash table that results from using the has function, $h(k) = (3k + 5) \bmod 11$, to hash the keys 12, 44, 13, 88, 23, 94, 11, 39, 20, 16, and 5, assuming collisions are handled by chaining.

HTVal	Key
0	13
1	94, 39
2	NULL
3	NULL
4	NULL
5	44, 88, 11
6	NULL
7	NULL
8	12, 23
9	16, 5
10	20

11. (10 points) R-9.8 p. 417

    What is the result of the previous exercise, assuming collisions are handled by linear probing?

HTVal	Key
0	13
1	94
2	39
3	16
4	5
5	44
6	88
7	11
8	12
9	23
10	20

12. (10 points) R-9.10 p. 417

    What is the result of Exercise R-9.7, when collisions are handled by double hashing using the secondary hash function $h_s(k) = 7 - (k \bmod 7)$?

HTVal	Key
0	13
1	94
2	23
3	88
4	39
5	44
6	11
7	5
8	12
9	16
10	20