Name: ___________________________

CSE332 Summer 2010: Midterm Exam Sample Solutions

Closed notes, closed book; calculator ok.

Read the instructions for each problem carefully before answering. Problems vary in point-values, difficulty and length, so you may want to work through the easier ones first in order to better budget your time. You have the full hour to complete this exam.

Good luck!

Problem	Max points	Score
1	9	
2	11	
3	11	
4	10	
5	11	
6	9	
7	14	
8	11	
9	14	
Total	100	

1. Big Oh Notation (9 points)

For parts (a) – (e), state whether the statement is true or false; part (f) provides its own instructions. You do not need to show your work for parts (a) – (e).

a. $3n\log_3 n$ is in $O(n^2)$
True

b. n is in $O(\log_2 n)$
False

c. n^3 is in $\Omega(n^4)$
False

d. $2n^3+n^2$ is in $\theta(n^3)$
True

e. 2^n is in $\Omega(n^2)$
True

f. Demonstrate that '$10n^3$ is in $O(n^3)$' by finding positive integers c & n_0 such that the definition of Big Oh is satisfied. You do not need to prove that these constants satisfy the definition; just specify the constants.

There are numerous (actually infinite) pairs of values that will work; a simple one is $c=10$, $n_0=1$.

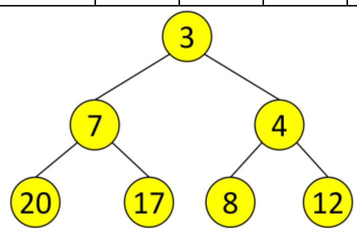
Rubric: 1 pt for (a)-(e), 4 for (f).

Name: ____________________________

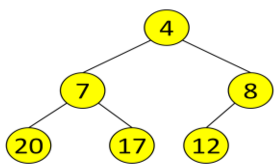
2. Binary Heap Operations (11 points)

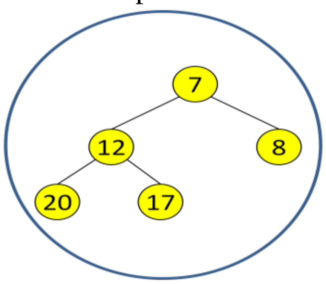
a. Draw the binary min heap represented by the following array:

Index	0	1	2	3	4	5	6	7
Value	7	3	7	4	20	17	8	12

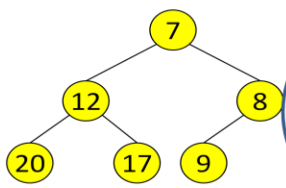


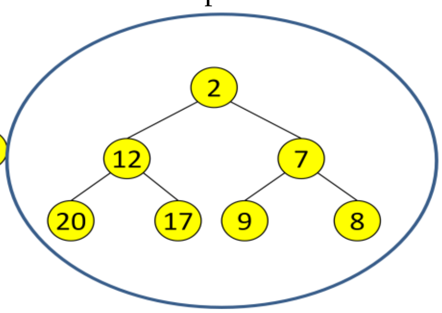
b. Show the result of calling deleteMin twice on the heap you drew in part (a). Show the heap after each deleteMin, and circle the final heap.





c. Starting with the heap you ended up with in part (b), insert values 9 & 2 in that order. Draw the heap after each insertion, and circle the final heap.





3. Binary Search Tree (11 points)

Write pseudo-code for a method in the BinarySearchTree class to count the number of node imbalances (according to the AVL definition of imbalance) in the Binary Search Tree. Assume that the tree is a valid Binary Search Tree (possibly empty). Essentially you need to check for imbalances at each node, and increment a counter if one has occurred here. You may declare class variables (such as an imbalance counter) and helper methods, if you wish. Clearly indicate which of your methods can be called to get the number of imbalances; it should return an 'int' and take no arguments.

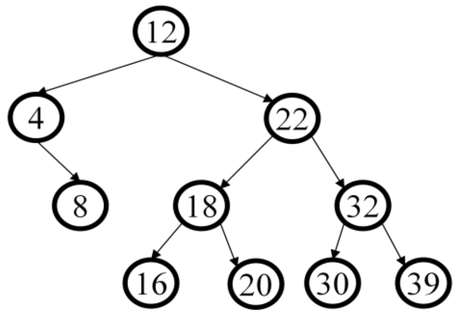
```
public class BinarySearchTree<E>
{
        … //other BST methods defined up here
        class BSTNode
        {
                E data;
                BSTNode left,right;
        }
        BSTNode overallRoot;

        int imbalances;

        //the method that returns the tree's imbalances
        int getImbalances()
        {
                imbalances=0;
                getImbalanceRec(overallRoot);
                return imbalances;
        }
        int getImbalanceRec(BSTNode node) //returns height of node upon which it was called
        {
                if(node==null) return -1;
                int lh=getImbalanceRec(node.left);
                int rh=getImbalanceRec(node.right);
                if(Math.abs(lh-rh) > 1) imbalances++;
                return Math.max(lh,rh)+1;
        }
```

Rubric: No points lost for 'Java' errors. ~3 lost for not defining how to compute height() for node, or for doing the computation wrong. More lost if algorithm unclear or unfinished. Many people defined height(BSTNode n) recursively, which would take O(n) time per node, so O(n^2) overall. The directions did not specify that it needed to be done in O(n) though, so no points were lost.

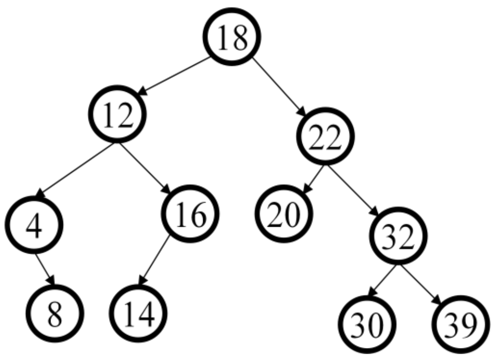
Name: ___________________________

4. AVL Tree Operations (10 points)

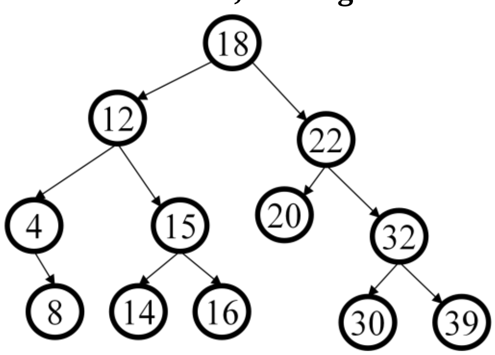
Starting with the following AVL tree, insert the following values in order: 14, 15 and 42. Draw the AVL tree resulting from each insertion. Label which tree is a result of which operation.



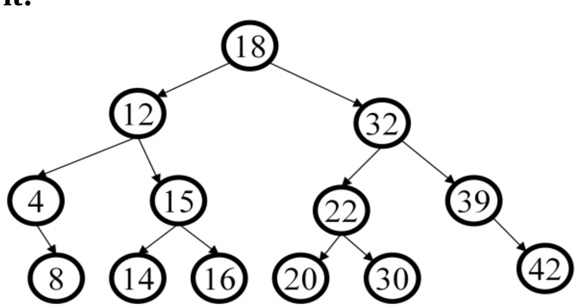
Upon inserting 14 we have an imbalance detected at the root. We perform a case 3 rotation to fix it:



Now we insert 15, causing an imbalance to be detected at 16. We perform a case 2 rotation to fix it:



Finally, insert 42, which, again, causes an imbalance; here detected at 22. Perform a case 4 rotation to fix it:



Rubric: ~1 point off for small errors, ~3 off for major errors.

5. Hash-Table Operations (11 points)

For each of the following versions of hash-tables, insert the following elements in order: 21, 20, 30, 29, 49 & 9. For each table, the TableSize is 10, and you should use the primary hash function h(k) = k%10. For each table, the first column holds the indices, and the second can hold values or pointers, depending on the hash-table type.

a. Separate chaining hash-table; use a standard linked list for each bucket, and insert at the front.

0	**→30→20**
1	**→21**
2	/
3	/
4	/
5	/
6	/
7	/
8	/
9	**→9→49→29**

Contents of the linked lists are shown in the second column for (a).

b. Linear probing

0	**20**
1	**21**
2	**30**
3	**49**
4	**9**
5	
6	
7	
8	
9	**29**

c. Quadratic probing

0	**20**
1	**21**
2	
3	**49**
4	**30**
5	
6	
7	
8	**9**
9	**29**

6. More Hash-Tables (9 points)

a. The following hash-table has 2 values already inserted, using a primary hash function of h(k) = k % 10, and a double hash function of g(k) = 7 – (k % 7). List a single integer that, when we attempt to insert it in the table below using the double-hashing mentioned, results in an infinite loop.

0	
1	21
2	
3	
4	
5	
6	36
7	
8	
9	

One possible answer is 16.

The key idea is that we'll hit an infinite loop if h(k) maps to 1 or 6, and g(k)=5; in that case, we'll just keep hitting indices 1 & 6 forever.

b. In a hash-table, what are the worst-case consequences of a poorly chosen primary hash function h( )? Assume that the hash function always returns a value in a timely manner.

The worst case occurs when all keys map to the same value; the hash-table will still work, but we'll get poor performance for most types of hash tables.

c. In a hash-table using double hashing, what are the worst-case consequences of a poorly chosen secondary hash function g( )? Assume that the secondary hash function always returns a value in a timely manner.

If the secondary hash function maps to 0, we'll have an infinite loop.

7. Tree Bounds (14 points)

The maximum number of nodes a binary tree of height h can have is
$\sum_{i=0}^{h} 2^i = 2^{h+1}-1$
As there is one at the root, 2 on the next level down, then 4, etc. So a binary tree of height 2 can have at most 7 nodes.

The maximum number of nodes in a d-ary tree of height h is:
$\sum_{i=0}^{h} d^i = (d^{h+1}-1)/(d-1)$

Provide an inductive proof that the bound stated above is the maximum number of nodes a d-ary tree of height h can have.

Base case: h=0
If h=0, then
$\sum_{i=0}^{0} d^i = 1 = (d^1-1)/(d-1)$

Inductive Case:
Assume it's true for k; that is,
$\sum_{i=0}^{k} d^i = (d^{k+1}-1)/(d-1)$

Now show it's true for k+1:
$\sum_{i=0}^{k+1} d^i = (d^{k+2}-1)/(d-1)$

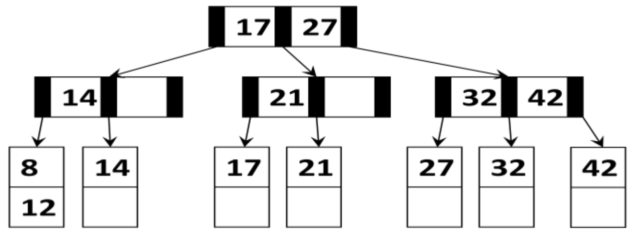
Start by writing it out:
$\sum_{i=0}^{k+1} d^i = 0+1+2+\ldots+d^{k+1} = \sum_{i=0}^{k} d^i + d^{k+1}$
$= (d^{k+1}-1)/(d-1) + d^{k+1}$
$=(d^{k+1}-1 + d^{k+2} - d^{k+1})/(d-1)$
$=(d^{k+2} - 1)/(d-1)$

Rubric: Some points were lost for lack of clarity and errors.

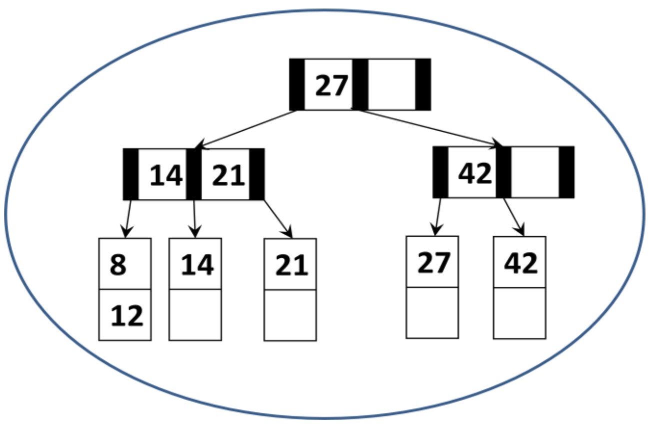
Name: ______________________________

8. B-Tree Operations (11 points)

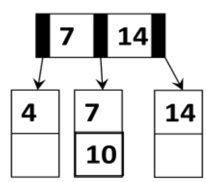
a. Starting with the B-Tree shown below, delete 32 and 17 in that order; draw and circle the resulting tree. When adopting, try to adopt first from the left sibling, then from the right. M=3 & L=2.

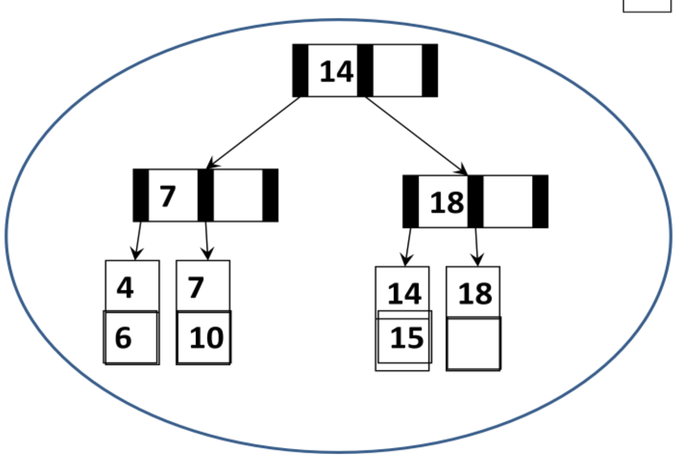


Result:



b. Starting with the B-Tree shown below, insert values 18, 15 and 6 in that order; draw and circle the resulting tree. Do not use adoption. M=3 & L=2.





9. Recurrence Relations (14 points)

Given the following recurrence relation:
T(1) = 3
T(n) = n + T(n-1)
Find the closed form for T(n), and show your work step-by-step. Your answer should *not* be in Big Oh notation; show the relevant constants in your final answer.

Start by expanding T(n)
T(n)=n+(n-1)+T(n-2)
= n+(n-1)+(n-2)+(n-3)+…+(n-(k-1))+T(n-k)
Separate the n's from '-1', '-2', etc. There are then k n's
=kn -1-2-3…+T(n-k)
=kn - $\sum_{i=1}^{k-1} i$ **+ T(n-k)**
We reduce to the base case by having T(n-k)=T(1); so n-k=1, so k=n-1.
Plug in that value for k.
=n(n-1) - $\sum_{i=1}^{n-2} i$ **+ T(1)**
We know $\sum_{i=1}^{c} i$ **= c(c+1)/2**
So
=n(n-1) – (n-1)(n-2)/2 + T(1)
=(n-1)(n+2)/2 + T(1)
=(n^2 **+ n -2)/2 + 3**

Rubric: Partial credit was given if the final answer was ~ n^2 **and if work was shown.**

Completely Non-CS Extra Credit

Each of the following is worth ½ a point of extra credit. If your final score is not an integer, it will be rounded up to the next integer.

a . From what movie is this quote taken? "Do or do not; there is no try"

Star Wars: The Empire Strikes Back

b. To whom is this quote attributed? "We are slaves of the law so that we are able to be free"

Cicero

c. To whom is this quote attributed? "I know not with what weapons World War III will be fought, but World War IV will be fought with sticks and stones"

Albert Einstein

d. From what movie is this quote taken? "There can be only one!"

The movie 'The Highlander'