Name_____

CMPS 1053 Final Exam
Fall 2012

Question	Possible	Earned
1) Arrays		
2) Lists		
3) Stacks		
4) Queues		
5) Recursion		
6) Trees		
7) Heaps		
8) Avl Tree's		

Name_____

I started putting things in some kind of logical order, but I found that there was a lot of overlap. So, I'm going to try to put things in differing categories, but questions will seep into other topics.

List of Topics that you should have a handle on:

1) **Arrays**
    a. **2 dimensional arrays.**
    b. **Looping through arrays.**
    c. **Arrays Structs**
2) **Lists**
    a. **Head, Tail**
    b. **Adding to a list**
    c. **Deleting from a list**
    d. **Traversing a list**
3) **Stacks**
    a. **Array based**
    b. **List Base**
4) **Queues**
    a. **Array base (circular queue).**
    b. **List Based**
5) **Recursion**
    a. **3 Parts of recursion**
    b. **Writing recursive functions**
    c. **What is recursion good for (not for efficiency)**
6) **Trees**
    a. **Label parts of a tree**
    b. **Full, Complete Tree's**
    c. **Binary Search Tree (BST)**
    d. **List based implementation**
    e. **Array based implementation**
7) **Heap**
    a. **Array based implementation**
    b. **Used as a priority queue**
    c. **Used to sort items.**
8) **AVL Tree**
    a. **Balanced binary tree**
    b. **Inserting values into a tree**
    c. **Single Rotations and Double Rotations**

<u>Arrays</u>

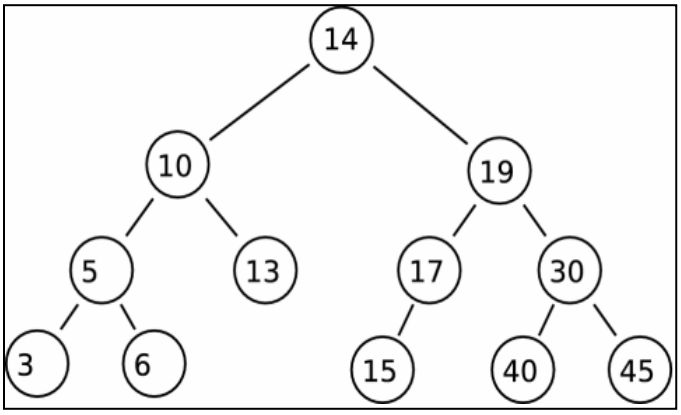
<u>Stacks, Queues and Lists</u>

Write a Stack Class using a linked list implementation.
Write a Stack Class using an array implementation.

Name_____

Write a Queue Class using a linked list implementation.
Write a Queue Class using an array implementation

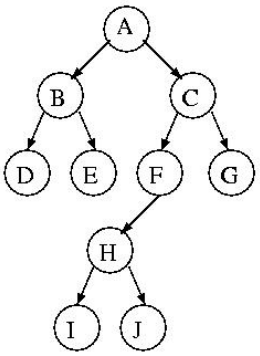
Binary Tree's



Here is a small binary tree:

A) Write the following directly on the tree:
-Put an "L" on all the leaves.
-Put an "R" on the root.
-Put an "A" on all the ancestors of the node that contains 40.
-Put an "D" on all the descendants of the node that contains 10.

B) Fill in the array representation of the tree



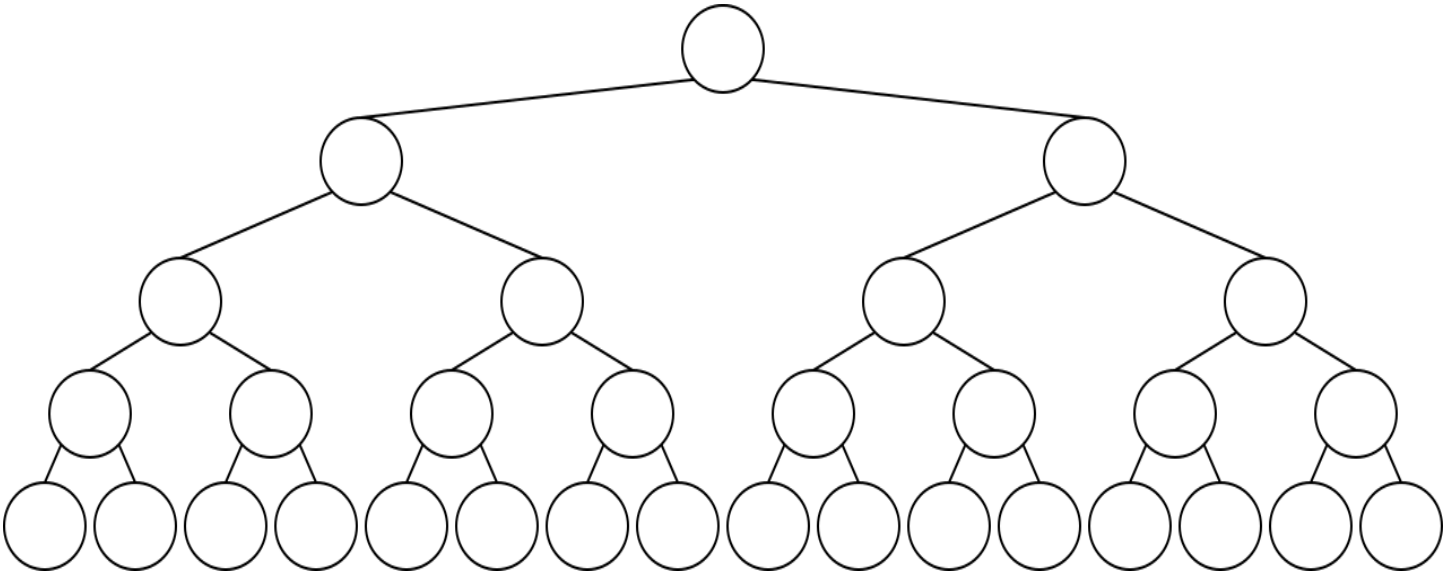
Array:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

C) Below is a binary tree illustration with all empty nodes. Place the following list of values into the tree in their proper order reading the values from left to right. If a node is left empty, then it is assumed null and not used.

Name_____

50, 20, 35, 14, 75, 66, 71, 74, 81, 7, 10, 18, 41, 45, 90, 1, 99, 25, 31



Sorting

Given an array of N elements, write a function called **BubbleSort** to sort the array using a bubble sort routine.

Searching

Use a standard binary search method to find the value **10** within the following array. Do anything necessary to the array to assist in your binary search. At each step, be sure to label the **First, Middle, and Last** (you can use F,M and L) locations. Also, you don't have to re-write every number in the array for each step, you can copy down only the necessary numbers still involved in the search. Circle the array location when your search is complete.

20	**10**	**12**	**22**	**2**	**16**	**18**	**24**	**4**	**14**	**6**	**28**	**26**	**30**	**8**
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
First							**Middle**							**Last**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Name________________________________

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Recursion

A) Write a recursive function to count the number of digits in an integer number N. Call your. function **CountDigits**.

B) Write a recursive function to return the Fibonacci value for some integer N. Call your function **Fibonacci**.

C) Given two lists, merge their nodes together to make one list, taking nodes alternately between the two lists. So ShuffleMerge() with {1, 2, 3} and {7, 13, 1} should yield {1, 7, 2, 13, 3, 1}. If either list runs out, all the nodes should be taken from the other list.

Lists & Pointers

Write a MaxValue () function that returns the largest integer value in the list.

```
List myList;
```

Name_____

```
int ArrayOfNums = {34,23,44,5,6,12,98,56,44,5,3,4,8};
myList.Load(ArrayOfNums);
cout<<"The largest value in the list is :"<<myList.MaxValue()<<endl;

/*
 Given a list return an integer containing the largest value in the list
*/

int MaxValue (struct node* head) {

// Your code
```

Binary Search Trees

Given the following list of values:

2,4,6,8,10,12,14,16,18…… N, where N = 2^{32}

Lets say you inserted those values into a Binary Search tree in the order you see them (from left to right). What problems would you run into? Would your Binary Search Tree be effective? If it is explain why, if not explain why and how you would fix it (what data structure would you use and why).

AVL Tree

Insert the following list of values into an AVL tree: 5, 6, 8, 3, 2, 4, 7. Show the final result.

Heaps

Consider the array theHeap = [-,3,5,6,7,20,8,2,9,12,15,30,17]

a. Draw the corresponding complete binary tree.
b. Insert elements 15, 20, and 45 (in that order) using the bubbling up process
c. Perform 4 remove min operations on the tree from part c.
d. Show the resulting tree.

Logic and Programming Problems

Using a stack that holds nodes (with the methods push, pop, empty) write a snippet of code that will reverse the nodes in a linked list. The variable "*Start*" is a node * that references the beginning of the list. You cannot

manipulate the values (by moving values between nodes) in the list, you can only change the order of the nodes.

Suppose a Stack and a Queue class have been defined, and implement the following public methods:

```
public class Stack
{
   public Stack();
   public void push(Object o);
   public Object pop(void);
   public int sizeOf(void);
   public boolean isEmpty();
}
```

```
public class Queue
{
   public Queue(void);
   public void insert(Object o);
   public Object retrieve(void);
   public int sizeOf(void);
   public boolean isEmpty();
}
```

Write a syntactically correct function **SumBoth** that adds the contents of both a Stack and a Queue, as in the prototype:

```
int SumBoth(Stack S, Queue Q)
{  /* your code here */ }
```

Note that you cannot assume that the Stack and the Queue initially have the same number of elements.

A doubly linked list is one where each node has two pointers – one points to the next node, and one points to the previous node. The "next" pointer for the last node points to NULL, as does the "previous" pointer for the first node. Suppose you are given a class List, which contains a linked structure of nodes declared as follows:

```
class Node {
  int data;
  Node *next;
  Node *prev;
}

class List {
  private:
    Node *head;
    Node *middle;
  public:
    // Lots of methods here
    bool deleteNum(int num);
}
```

Write the code for a method called deleteNum that takes an integer as a parameter and returns a bool. It should search through the linked list starting from the **middle**, find the first time that the integer appears, and remove it. If the integer is not present in the list, deleteNum should return false. The list is in ascending order, so the smallest value is pointed to by head.

Name_____

Write a function **Average** in C++ which will return the **Average** of all numbers stored in a two dimensional array of Integers. Below is a skeleton of what your function:

```
double Avg2D(int A[100][100],int width,height){

}
```

Draw the (1) stack and (2) queue and (3) circular queue data structures in array implementations for "each step" in the following sequence: **add(A), add(B), add(C), remove, add(D), remove , add(E), remove, add(F), add(G).** Assume an initial size of 5 for the array implementation. Remember to show TOP (top of the stack) for stack and both Front and Back for queue. In circular queue implementation, assume we sacrifice one storage space for telling the difference between FULL and EMPTY.

A) Draw the stack implementation
B) Draw the queue implementation
C) Draw the circular queue implementation

Below you see a stack ***S*** in its initial state. (Stack uses Push and Pop)
A queue ***Q***, in its initial state. (Queue uses Add and Remove)
And a list ***L*** that's in its FINAL state. (List uses Add)

1
6
3
9
2

Q

11	16					

L

9	11	2	8	3	6	1	16

Name_____

S | 8 |

You need to write a series of push, pops, adds, and removes to get the values from ***S*** and ***Q*** into ***L*** in the order displayed. You must use compound statements like **`Q.Add(S.Pop())`** to move values around. You cannot use temp variables like X=S.Pop() then Q.Add(X). ***`L.Add`*** adds the values to the <u>rear</u> of the list.

Use the table below to write your series of commands. You can write || (two vertical bars, if you're executing the same command as the previous.

1		16	
2		17	
3		18	
4		19	
5		20	
6		21	
7		22	
8		23	
9		24	
10		25	
11		26	
12		27	
13		28	
14		29	
15		30	

1)Which of the following are linked list operations?
A)traversing the list
B)appending a node
C)inserting or deleting a node
D)All of the above
E)None of the above

2)The successor pointer in the last node of a linked list should have its value set to

Name_____

A)the address of the previous node.
B)nothing: the last node has no successor pointer.
C)the address of the first node in the list.
D)NULL.
E)None of the above.

3)Variations of the linked list are:
A)circular linked list.
B)doubly-linked list.
C)triply linked list.
D)A and B.
E)None of the above.

4)Stacks are useful data structures for algorithms that work with lists of items in a
A)first in, first out order.
B)first in, last out order.
C)garbage in, garbage out order.
D)last in, last out order.
E)None of the above.

5)Static stacks have a _____ size, and are implemented using _____.
A)variable, linked lists
B)variable, arrays
C)fixed, linked lists
D)fixed, arrays
E)None of the above

6)A stack has two primary operations called
A)push and pop.
B)insert and delete.
C)append and delete.
D)push and pull.
E)None of the above.

7)A dynamic stack may be implemented as a(n) _____, and expand or shrink with each push or pop operation.
A)structure
B)linked list
C)array
D)A and B
E)None of the above

8)A queue is a data structure that stores and retrieves items in a _____ manner.
A)first in, last out
B)last in, first out
C)random
D)first in, first out
E)None of the above

9)When an element is added to a queue, it is added to the rear. When an element is removed, it is removed from the _____.
A)front

Name_____

B)rear
C)middle
D)All of the above
E)None of the above

10)In a dequeue operation, the element at the _____ of the queue is removed.
A)front
B)middle
C)declaration
D)mid-point
E)None of the above

11)A dynamic queue can be implemented as a(n)_____.
A)fixed-length array
B)dynamic linked list
C)fixed-length circular array
D)All of the above
E)None of the above

12)A strong reason to use a binary search tree is
A)to enhance code readability.
B)it is more flexible than the unary search tree.
C)aesthetics and program design.
D)to expedite the process of searching large sets of information.
E)None of the above.

13)A node that has no children is a _____.
A)pure binary node
B)leaf node
C)head node
D)root node
E)None of the above

14)When an application begins searching a binary search tree, it starts at:
A)the middle node, halfway between the root and the longest branch.
B)the root node.
C)the rightmost child of the root node.
D)the outermost leaf node.
E)None of the above.

15)The _____ in a binary tree is analogous to the head pointer in a linked list.
A)root pointer
B)null pointer
C)binary pointer
D)leaf pointer
E)None of the above

16)Visiting all nodes of a binary tree in some methodical fashion is known as
A)walking through tree.
B)branching out along the tree.
C)traversing the tree.

Name_____

D)climbing the tree.
E)None of the above.

17)Values are commonly stored in a binary search tree so that a node's _____ child holds data that is less than the _____ data.
A)right, left child's
B)left, node's
C)left, right child's
D)right, node's
E)None of the above

18)A linked list class using dynamically allocated memory should free its memory when the list is destroyed. This can be done by
A)the class destructor.
B)overloading the class removal function.
C)overriding the class removal function.
D)the system's memory deallocator.
E)None of the above.

19)Each node in a _____ list contains pointers to the nodes before and after it.
A)doubly-linked
B)singly-linked
C)circular-linked
D)both B and C
E)None of the above

20)A linked list can grow and shrink as a program runs.

21)Linked lists are less complex to code and manage than arrays.

22)Nodes in a linked list are stored in contiguous memory.

23)The programmer must declare in advance the size of a dynamic stack or queue.

24)The number of nodes in a binary tree is the number of nodes in its left subtree plus the number of nodes in its right subtree.

25) Output will be the same if you use inorder, postorder, or preorder traversals to print the values stored in a binary tree.

Explain the fundamental differences between a stack and a queue. What information needs to be remembered for each. Write the class definition for each (Stack.h, Queue.h).

Grive a brief explanation of the following operations. Include worst case running time vs average case running time if necessary. Give examples of possible improvements.

- Inserting into a linked list?

Name_____

- Searching a linked list?
- Deleting from a doubly linked list?
- Searching for a value in a Binary Tree?
- Searching for a value in an AVL Tree?
- Sorting an array using a swap method (no divide and conquer)?

Given an array of N elements, write a function bubble_sort to sort the array.

Given some array of N elements, show how a binary search method works.

Write recursive methods to do the following:

Reverse the numbers in some integer N.
Count the digits in some integer N.
Return the Fibonacci value for N.
Create your own "pow" function raising some number X, to the power N.

Assessment:

1) What is your expected grade in this course:? A B C D F

2) What did you earn in CS 1? A B C D F

3) On a scale of 1-10 (10 being the hardest), on average, how hard were the programs?

1 2 3 4 5 6 7 8 9 10

4) Did you turn in all your programs?