Model Lab Exercises

Question No.	Problem
1.	Implement a C**ontact book application** using Doubly Linked List.
2.	Make a **Dictionary** using Binary trees
3.	Implement Huffman coding: this requires tinkering with binary trees and priority queues.
4.	Recursive selection sort Write a program to determine the median of the array given below: (9, 4, 5, 1, 7, 78, 22, 15, 96, 45,25)
5.	Write a Modified Insertion Sort Algorithm for integer key values. However, the moderation is: The input is a stack (not an array), and the only variables that your algorithm may use are a fixed number of integers and a fixed number of stacks. The algorithm should return a stack containing the records in sorted order (with the least value being at the top of the stack), with example. Implement optimized bubble sort
6.	Implement a radix sorting machine. A radix sort for base 10 integers is a mechanical sorting technique that utilizes a collection of bins, one main bin and 10 digit bins. Each bin acts like a queue and maintains its values in the order that they arrive. The algorithm begins by placing each number in the main bin. Then it considers each value digit by digit. The first value is removed and placed in a digit bin corresponding to the digit being considered. For example, if the ones digit is being considered, 534 is placed in digit bin 4 and 667 is placed in digit bin 7. Once all the values are placed in the corresponding digit bins, the values are collected from bin 0 to bin 9 and placed back in the main bin. The process continues with the tens digit, the hundreds, and so on. After the last digit is processed, the main bin contains the values in order.
7.	Use general trees to implement a simple algorithm enumerating all the possible words from a fixed alphabet at a given length. You can use that algorithm to implement a naive dictionary attack and crack a set of passwords.
8.	Consider any five elements randomly and design an array based data structure for two stacks called a DualStack The two stacks should share the same array in an efficient manner. If there are MaxSize entries in the array then the IsFull function should only return true if all the entries in the array are occupied. Your operations should all be constant time. a. Implement Push, Pop, IsEmpty and IsFull functions. b. Explain why such a nice data structure would not be possible for 3 stacks.
9.	It is often handy to store a binary tree in a file. Assume each node in the binary tree contains a character string. Assume also that all operations you need on strings are provided. For

	example, you do not need to design algorithms to test if a string equals ".", to write a string into a file, or to read a string from a file. To create the file in preorder file format do a preorder traversal of the tree, when a node is visited put the character string in the file followed by a newline and when a null is visited put a dot, ".", followed by a newline. For example, the tree a / \ b c / / \ d e f is stored as the file "a b d . . . c e . . f . ." where spaces indicate newlines. o Design an algorithm with example which outputs the preorder file format of a binary tree given a pointer to the root of a binary tree. Assume the binary tree has nodes with fields "data", "left_child", and "right_child". o Design an algorithm with example which takes a preorder file format of a binary tree and produces the binary tree. Hint: an effective approach is to design a recursive function that processes a sequence of lines in a file and returns a binary tree.
10.	Suppose we have an unsorted array A[1..n] of integers with possible duplicates. Design a version of Quicksort that instead of partioning into two sets, one whose elements are less than or equal to the pivot and a second whose elements are greater than or equal to the pivot, the new algorithm partitions into three sets, one whose elements are strictly less than the pivot, a second whose elements are stricktly more than the pivot, and a third whose elements are equal to the pivot. Your algorithm should be in-place. One idea is that in the partitioning phase that as we move the two pointers i and j toward each other we maintain the invariant that the array looks like; **i j** **[elements equal to pivot] [elements less than pivot] [unknown elements] [elements greater than pivot] [elements equal to pivot]** When there are no unknown elements left then the elements can be rearranged to be of this form: **[elements less than pivot][elements equal to pivot][elements greater than pivot]** Design the Quicksort and Partition algorithms with example that implements this idea.
11.	Some project planning applications use a labeled acyclic directed graphs to represent the jobs and job times on a project. A vertex in the graph represents a job and its label

	represents the time the job will take. A directed edge from one vertex to another represents the fact the job represented by the first vertex must be completed before the job represented by the second vertex. Assume we have a directed acyclic graph G =({1,2,...,n},E) with vertices labeled by non-negative integers c_1, c_2, ... ,c_n . The label c_i represents the time job i will take. Assume futher that every vertex is reachable by some path from vertex 1, vertex 1 has in-degree 0, vertex n is reachable by some path from every vertex, and n has out degree 0. Vertex 1 represent the beginning of the project and vertex n represent the end of the project. The length of a path from 1 to n is the sum of the labels on the vertices along the path. Design an algorithm based on the topological sort algorithm to find the length of a longest path from 1 to n in the graph. The length of the longest path represents how long the entire project will take. Sometimes a longest path is called a critical path. Your algorithm should use the adjacency list representation of a graph. The labels can be stored in an additional array. Your algorithm should run in linear time. Hint: ultimately you will need to compute the length of the longest path from 1 to every other vertex. In the topological sort, when a vertex achieves in-degree 0, the length of the longest path from 1 to it should be known.
12.	Draw a picture of a directed graph with vertices a, b,..., f and edges ab, ad, ae, bc, be, bf, ca, db, df, ed, ef, fa and fc with edge weights 8, 3, 6, 1, 4, –2, 5, 4, 2, 10, 9, 2, 3, (in that order). • Find a shortest path tree rooted at vertex f and highlight the edges by drawing them extra heavy. • Select three edges that are not in your shortest path tree from problem 1. Show that all three edges satisfy the shortest path tree condition. • Select some edge from your shortest path tree. If you increase the length of this edge by a large enough amount, the tree will cease to be a shortest path tree. Determine the smallest integer edge weight for which this is true. Show that there is at least one edge in the graph that now violates the shortest path tree condition.
13.	Given two square matrices A and B of size n x n each, find their multiplication matrix Using Strassen's algorithm.
14.	A recursive version of Insertionsort works as follows. We have a list to sort. If the list is empty then there is nothing to do. Otherwise, sort all but the first element of the list recursively, then insert the first element of the list into its proper place in the the list by comparing it to members of the returned list which is already sorted. o Assume a node in the list has two fields: value which holds an integer, and next which is a pointer to a node. Design the recursive list version of Insertionsort node pointer in pseudocode. o Analyze the worst case time of your algorithm. Because it is recursive your time bound should first be defined by a recurrence, then the recurrence should be solved.
15.	Imagine that you have been asked to to implement a new IRS database. There are 100,000,000 records, each of which take an average of 2,000 bytes. The records are keyed with a Social

	Security Number which is 4 bytes. The computer that you will be using has 2,048 byte pages and pages are addressed with 4 byte integers. Assume that a B-tree node completely as possible fills a page. This may means that leaves will hold may hold more or less keys than internal nodes depending on what data is stored on the leaves. Assume that the computer has 16 MB of memory that is usable for storing all or part of a search structure. For the B-tree, disk addresses are used for pointers, and a 2 byte integer is used to keep track of the length of the active area in the B-tree node. The leaves of the B-tree contain pointers to the actual records. We assume that the nodes, other than the root, of the B-tree are about 3/4-th full on average. a. Calculate the maximum number of children (M) that an internal node of the B-tree can have. Calculate the maximum number of entries (L) a leaf of the B-tree can have. b. Calculate the height of the B-tree with that order so that all the 100,000,000 records can be accessed. About how many children does the root have. c. Calculate how many levels of the B-tree can fit into the memory of the computer. Several full levels and the portion of one level will fit into memory. Calculate how many disk accesses in the worst case are necessary to find a specific record using this search structure. How many on average?
16.	Implement modified quick sort
17.	You are given an array containing ***n*** integers. You are asked to determine if the list contains two numbers that sum to some arbitrary integer, ***k***. For example, if the list is **8, 4, 1** and **6** and ***k* = 10**, the answer is "yes", because **4+6 = 10**. a. Find a way to solve this problem that is $O(n^2)$. b. Can you do better? What is the time complexity of the better method?
18.	Convert the Heapify function to an iterative form. Implement and show the result.
19.	There are ***k*** sorted lists containing a total of ***n*** elements. Give an $O(n \log k)$ algorithm to merge these lists into a single sorted list. *Hint:* Use a heap for the merge.
20.	Convert a recursive quicksort to an iterative one. Implement and show the result. Suggest a simple strategy that can be used to turn **any** sort into a stable sort.
21.	Implement deque. A double-ended queue or deque is one that has both LIFO and FIFO behaviour, ie you can add an item to the head or the tail of a list and extract an item from the head or the tail.