LAB MANUAL

ALGORITHMS LABORATORY[CS-2098]

School of Computer Engineering

KIIT UNIVERSITY, BHUBANESWAR

INSTRUCTIONS TO STUDENTS

- Students should be regular and come prepared for the lab practice.
- In case a student misses a class, it is his/her responsibility to complete that missed experiment(s)
- Students should bring the observation book, lab journal and lab manual. Prescribed textbook and class notes can be kept ready for reference if required.
- Once the experiment(s) get executed, they should show the results to the instructors and copy the same in their observation book.
- The algorithms have to be implemented in C/C++.

PROCEDURE FOR EVALUATION

The entire lab course consists of 100 marks. The marking scheme is as follows

Continuous Evaluation marks	60
End Sem. Lab Examination	40
Total	**100**

Scheme for continuous evaluation

Students will be evaluated bi-weekly. Minimum 6 evaluations should be conducted for each student. Each evaluation carries 10 marks. The scheme is as follows:

Program & Execution	5
Observation	3
Viva-Voce	2
Total	**10**

Scheme for end sem lab examination

End sem. lab exam will be conducted after the completion of all the weekly exercises. The student will not be allowed for exam if he/she is found short of attendance and has not completed all the experiments. The marking scheme for end sem lab exam is as follows:

Program Written & execution, Checking Results for all inputs	15
Final Viva-Voce	25
Total	**40**

CONTENTS

Lab-1: Review of Fundamental Data Structures

Lab. Exercises (LE)

LE1.1 Given an array. Let us assume that there can be duplicates in the list. Write a program to search for an element in the list in such a way that we get the highest index if there are duplicates.

LE1.2 Write a program for finding i and j in an array A for any key such that $A[j]^2 + A[i]^2 == key$

LE1.3 Given key in a sorted array A with distinct values. Write a program to find i, j, k such that A[i] + A[j] + A[k] == key.

LE1.4 Suppose an array A has n distinct integers. Increasing sequence is given as A[1].....A[k] and decreasing sequence is given as A[k+1].....A[n]. Write a program for finding k.

LE1.5 Write a program to display an array of n integers (n>1) in O(n) time, where at every index of the array should contain the product of all elements in the array except the element at the given index. No additional array declaration is allowed.

Example:

Input : 10, 4, 1, 6, 2

Output : 48,120,480,80,240

Home Exercises (HE)

HE1.1 Write an algorithm for finding counting inversions in an array. Inversion is a pair such that for an array A = {a1, a2, a3,. , an}, and ai > aj and i < j.

HE1.2 You have 1 to N1 array and 1 to N numbers and one number is missing. Write a program for finding that missing number.

HE1.3 Write a program to find out the largest difference between two elements A[i] and A[j] (A[j]-A[i]) of the array of integers A in O(n) time such that j > i. For example: Let A is an array of integers:

int[] a = { 10, 3, 6, 8, 9, 4, 3 };

if i=1, j=3, then diff = a[j] – a[i] = 8 – 3 = 5

if i=4, j=6, then diff = a[j] – a[i] = 3 – 9 = -6

…….

…….

if i=1, j=4, then diff = a[j] – a[i] = 9 – 3 = 6

……..

……..

6 is the largest number between all the differences, that is the answer.

Lab-2: Fundamentals of Algorithmic Problem Solving

Lab. Exercises (LE)

LE2.1 Write a program to implement GCD (greatest common divisor) using the following three algorithms.

a) Euclid's algorithm
b) Consecutive integer checking algorithm.
c) Middle school procedure which makes use of common prime factors. For finding list of primes implement sieve of Eratosthenes.

Find gcd(31415, 14142) by applying each of the above algorithms.

LE2.2 Write a program to find out which algorithm is faster for the above data. Estimate how many times it will be faster than the other two.

LE2.3 Write a program to solve the problems LE1.5 in O(n) time.

Home Exercises (HE)

HE2.1 Write a program using a function for computing $\lfloor\sqrt{n}\rfloor$ for any positive integer.Besides assignment and comparison, your algorithm may only use the four basic arithmetic operations.

HE2.2 Write a program to implement recursive solution to the Tower of Hanoi puzzle.

HE2.3 Write a program to compute the n^{th} Fibonacci number recursively and find its time complexity.

HE2.4 Write a program to solve the problems HE1.3 in O(n) time.

Lab-3: Brute force Techniques

Lab. Exercises (LE)

LE3.1 Write a program to perform linear search operation in an array of n integers. Determine the time required to search an element. Repeat the experiment for different values of n, the number of elements in the list to be searched and plot a graph of the time taken versus n. The n integers can be generated randomly.

LE3.2 Write a program to sort a given set of elements using the insertion sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

LE3.3 Write a program to sort a given set of elements using the heap sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

Home Exercises (HE)

HE3.1 Write a program to find the k^{th} minimum and maximum element in Heap.

HE3.2 Write a program to delete k^{th} indexed element in Min heap and Max heap.

HE3.3 Given a string called TEXT with ‘n’ characters and another string called PATTERN with ‘m’ characters(m<=n) .Write a program which implements brute force string matching to search for a given pattern in the text. If thepattern is present then find the position of first occurrences of Pattern in that Text.

Lab-4: Divide and Conquer

Lab. Exercises (LE)

LE4.1 Write a program to sort a given set of elements using the Merge sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

LE4.2 Write a program to recursively implement Binary Search using divide and conquer method. Determine the time required to search an element in an array of n integers. Repeat the experiment for different values of n, the number of elements in the list to be searched and plot a graph of the time taken versus n. The n integers can be generated randomly.

Home Exercises (HE)

HE4.1 Write a program to use divide and conquer method to recursively implement and to find the maximum and minimum in a given list of n elements.

HE4.2 Write a program to sort a given set of elements using the Quick sort method and determine the time required to sort the elements. Repeat the experiment for different values of n, the number of elements in the list to be sorted and plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator.

Lab-5: Greedy Technique-I

Lab. Exercises (LE)

LE5.1 Write a program to implement the file or code compression using Huffman‟s algorithm.

LE5.2 Write a program to implementation of Fractional Knapsack algorithm.

LE5.3 You are given n events where each takes one unit of time. Event i will provide a profit of gi dollars (gi > 0) if started at or before time ti where ti is an arbitrary real number. (Note: If an event is not started by ti then there is no benefit in scheduling it at all. All events can start as early as time 0). Write a program to implement to implement this Job Scheduling with dead line problem.

LE5.4 Write a program to implement the activity-selection problem stated as follows:
You are given n activities with their start and finish times. Select the maximum number of activities that can be performed by a single person, assuming that a person can only work on a single activity at a time. Example: Consider the following 6 activities. start[] = {1, 3, 0, 5, 8, 5}; finish[] = {2, 4, 6, 7, 9, 9}; The maximum set of activities that can be executed by a single person is {0, 1, 3, 4}.

Home Exercises (HE)

HE5.1 Write a program to implement Optical Storage on tapes.

HE5.2 Consider the following scheduling problem. You are given n jobs. Job i is specified by an earliest start time si, and a processing time pi. We consider a preemptive version of the problem where a job's execution can be suspended at any time and then completed later. For example if n = 2 and the input is s1 = 2, p1 = 5 and s2 = 0, p2 = 3, then a legal preemptive schedule is one in which job 2 runs from time 0 to 2 and is then suspended. hen job 1 runs from time 2 to 7 and secondly, job 2 is completed from time 7 to 8. The goal is to output a schedule that minimizes ΣCj = 1, where Cj is the time when job j is completed and j runs from 1 to n. In the example schedule given above, C1 =7 and C2 =8.

Lab-6: Greedy Technique-II

Lab. Exercises (LE)

LE6.1 From a given vertex in a weighted connected graph, Write a program to find shortest paths to other vertices using Bellman-Ford algorithm. Draw simple, connected weighted graph with 10 vertices and 18 edges, such that graph contains minimum weight cycle with at least 4 edges. Show that the Bellman-Ford algorithm find this cycle.

LE6.2 From a given vertex in a weighted connected graph, Write a program to find shortest paths to other vertices using Dijkstra's algorithm. Draw simple, connected weighted graph with 8 vertices and 16 edges, each with unique edge weight. Identify one vertex as a start vertex and obtain shortest path using Dijkstra's algorithm.

LE6.3 Write a program to find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm.

Home Exercises (HE)

HE6.1 Write a program to find to determine whether a directed graph with positive and negative cost edges has negative cost cycle.

HE6.2 Write a program to find the strongly connected components in a digraph.

HE6.3 Write a program to find the kth shortest path between two given nodes of a graph.

HE6.4 Write a program to find Minimum Cost Spanning Tree of a given undirected graph using Prims algorithm.

Lab-7: Dynamic Programming

Lab. Exercises (LE)

LE7.1 Write a program to implement All Pair Shortest paths problem using Floyed Warshall's Algorithm.

LE7.2 Write a C program to implement the Longest Common Subsequence.
Sample Inputs & outputs
The LCS of HUMAN and CHIMPANZEE is HMAN

LE7.3 Write a program to implement the matrix chain multiplication problem using M-table & S-table or to find optimal ordering of matrix multiplication.

Home Exercises (HE)

HE7.1 Write a program to find the Longest palindromic subsequence (Hint: Subsequence is obtained by deleting some of the characters from a string without reordering the remaining characters, which is also a palindrome).

HE7.2 Write a program to implement 0/1 Knapsack problem using dynamic programming.

HE7.3 Find the Binomial Coefficient using Dynamic Programming.

Lab-8: Graph Traversals

Lab. Exercises (LE)

LE8.1 Write a C program to implement Breadth First Search.
LE8.2 Write a C program to implement Depth First Search.
LE8.3 Write a C program to check whether a given graph is connected or not using DFS method.
LE8.4 Write a C program to check whether a given graph is connected or not using DFS method.

Home Exercises (HE)

HE8.1 Write a C program to print all the nodes reachable from a given starting node in a given digraph using Depth First Search method.
HE8.2 Write an algorithm to print all the nodes reachable from a given starting node in a digraph using BFS method

Lab-9: Computational Complexity

Lab. Exercise (LE)

LE10.1 Write a program to implement subset-sum problem in polynomial time if the target value t is expected in unary.

Home Exercises (HE)

HE10.1 Write a program to implement any scheme to find the optimal solution for the Travelling Salesperson problem and then solve the same problem instance using any approximation algorithm and determine the error in the approximation. TSP Approximation Algorithm (Known as Christofides Heuristics)