**LAB MANUAL**

On

Algorithms Laboratory

For BE 4th Semester Students.

**Instructions to the Readers/Students:**

1. This document is a useful material for the 4th Semester students (2013 and 2015 Scheme P13ISL47 and P15ISL47) of our Syllabus.
2. Though these programs are compiled and verified on Turbo Compiler, they can be executed on different C compilers with little/no modifications.
3. Additional explanation for the program is provided, wherever necessary. However, it is advised to know the theory concepts by referring to the respective subject.
4. Clarifications/suggestions are welcome!!
5. Students should be regular and come prepared for the lab practice.
6. In case a student misses a class, it is his/her responsibility to complete that missed experiment(s).
7. Students should bring the observation book, and lab manual. Prescribed textbook and class notes can be kept ready for reference if required.
8. Once the experiment(s) get executed, they should show the results to the instructors and copy the same in their observation book.
9. The algorithms have to be implemented in C/C++ and OOPs Concepts.
10. Assume integer data, if explicitly not mentioned.

**PROCEDURE FOR EVALUATION:**

The entire lab course consists of 50 Continuous Internal Evaluation (CIE) Marks. The marking scheme is as follows.

|  |  |
| --- | --- |
| Continuous Evaluations. | 10 Marks |
| Semester End Lab Test. | 40 Marks |
| **Total** | **50 Marks** |

**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 and Execution | 5 marks |
| Observation | 3 marks |
| Viva-voce | 2 Marks |
| **Total** | **10 Marks** |

**Scheme for End Sem Lab Test:**

End SEM lab Test will be conducted after the completion of all the Programs. The student will be not allowed for Test if He/She is found shortage of attendance and has not completed all the experiments. The marking scheme for end sem lab exam is as follows:

|  |  |
| --- | --- |
| Algorithm Design & Write-up of program | 08 Marks |
| Program execution | 15 Marks |
| Viva -voce | 07 Marks |
| **Total** | **30 Marks** |
| Record Writing | 10 Marks |
| **Grand Total** | **40 Marks** |

**Lab Instructions**

**Contents**

|  |  |  |  |
| --- | --- | --- | --- |
| **Q. No.** | **Description** | | **Page No.** |
| **1** | Implement the ascending and descending order of elements using Selection Sort.  Implement the ascending and descending order of elements using Bubble Sort. | |  |
| **2** | Implement Quick Sort using first/last/any random element as pivot.  Implement the ascending and descending order using Quick Sort. | |  |
| **3** | Implement the ascending and descending order of elements using Merge Sort.  Implement the stressen’s matrix multiplication problem. | |  |
| **4** | Write an algorithm to print all the nodes reachable from a given starting node in a digraph using BFS method.  Obtain the Topological ordering of vertices in a given digraph using Source Removal Method. | |  |
| **5** | Check whether a given graph is connected or not using DFS method.  Obtain the Topological ordering of vertices in a given digraph using DFS Method. | |  |
| **6** | Compute the transitive closure of a given directed graph using Warshall’s algorithm.  Implement Floyd’s algorithm to find the All-Pairs-Shortest-Paths problem. | |  |
|  | | | |
| **7.** | Solve Binomial Co-efficient problem Using Dynamic Programming.  Implement 0/1 Knapsack problem using dynamic programming. |  | |
| **8.** | Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal’s algorithm.  Find Minimum Cost Spanning Tree of a given undirected graph using Prim’s algorithm. |  | |
| **9.** | From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.  Implement Horspool algorithm for String Matching. |  | |
| **Compulsory Lab Program for each Student in the Group** | | | |
| **1** | Implement the algorithm for the Hash Table to check both arrays have same set of numbers/Characters (Input: Given two arrays of unordered numbers/Characters). |  | |
| **2.** | Implement the algorithm for the Hash Table to removing the duplicate characters/ numbers (Input: an array of characters/numbers). |  | |
| **3.** | Give an algorithm for finding non repeated character in a string/number using Hash table. Example the first non repeated character in the string “abbcc” is “a”. |  | |
| **4.** | Implement the file or code compression using Huffman’s algorithm. |  | |

**Note:**

1. Implement the following using C/C++ programming Language on LINUX or Windows like platform.
2. Each Program Input elements can be read from a file or can be generated using random number generator.
3. Determine the time required to sort 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.