DESIGN AND ANALYSIS OF ALGORITHMS LABORATORY

(Effective from the academic year 2018 -2019)

SEMESTER – IV

Course Code	**18CSL47**	CIE Marks	40
Number of Contact Hours/Week	0:2:2	SEE Marks	60
Total Number of Lab Contact Hours	36	Exam Hours	03

Credits – 2

Course Learning Objectives: This course (18CSL47) will enable students to:

- Design and implement various algorithms in JAVA
- Employ various design strategies for problem solving.
- Measure and compare the performance of different algorithms.

Descriptions (if any):

- Design, develop, and implement the specified algorithms for the following problems using Java language under LINUX /Windows environment. Netbeans / Eclipse or IntellijIdea Community Edition IDE tool can be used for development and demonstration.
- **Installation procedure of the required software must be demonstrated, carried out in groups and documented in the journal.**

Programs List:

1.	
a.	Create a Java class called ***Student*** with the following details as variables within it. (i) USN (ii) Name (iii) Programme (iv) Phone Write a Java program to create *nStudent* objects and print the USN, Name, Programme, and Phoneof these objects with suitable headings.
b.	Write a Java program to implement the Stack using arrays. Write Push(), Pop(), and Display() methods to demonstrate its working.
2.	
a.	Design a superclass called ***Staff*** with details as StaffId, Name, Phone, Salary. Extend this class by writing three subclasses namely ***Teaching*** (domain, publications), ***Technical*** (skills), and ***Contract*** (period). Write a Java program to read and display at least 3 *staff* objects of all three categories.
b.	Write a Java class called ***Customer*** to store their name and date_of_birth. The date_of_birth format should be dd/mm/yyyy. Write methods to read customer data as <name, dd/mm/yyyy> and display as <name, dd, mm, yyyy> using StringTokenizer class considering the delimiter character as "/".
3.	
a.	Write a Java program to read two integers *a* and*b*. Compute *a*/*b* and print, when *b* is not zero. Raise an exception when *b* is equal to zero.
b.	Write a Java program that implements a multi-thread application that has three threads. First thread generates a random integer for every 1 second; second thread computes the square of the number andprints; third thread will print the value of cube of the number.
4.	Sort a given set of ***n*** integer elements using **Quick Sort** method and compute its time complexity. Run the program for varied values of *n*> 5000 and record the time taken to sort. Plot a graph of the time taken versus ***n***on graph sheet. The elements can be read from a file or can be generated using the random number generator. Demonstrate using Java how the divide-and-conquer method works along with its time complexity analysis: worst case, average case and best case.
5.	Sort a given set of ***n*** integer elements using **Merge Sort** method and compute its time

	complexity. Run the program for varied values of *n*> 5000, and record the time taken to sort. Plot a graph of the time taken versus ***n***on graph sheet. The elements can be read from a file or can be generated using the random number generator. Demonstrate using Java how the divide-and-conquer method works along with its time complexity analysis: worst case, average case and best case.
6.	Implement in Java, the **0/1 Knapsack** problem using (a) Dynamic Programming method (b) Greedy method.
7.	From a given vertex in a weighted connected graph, find shortest paths to other vertices using **Dijkstra's algorithm**. Write the program in Java.
8.	Find Minimum Cost Spanning Tree of a given connected undirected graph using **Kruskal'salgorithm.** Use Union-Find algorithms in your program
9.	Find Minimum Cost Spanning Tree of a given connected undirected graph using **Prim's algorithm**.
10.	Write Java programs to (a) Implement All-Pairs Shortest Paths problem using **Floyd's algorithm**. (b) Implement **Travelling Sales Person problem** using Dynamic programming.
11.	Design and implement in Java to find a **subset** of a given set **S** = {Sl, S2,.....,Sn} of ***n*** positive integers whose SUM is equal to a given positive integer ***d***. For example, if S ={1, 2, 5, 6, 8} and ***d***= 9, there are two solutions {1,2,6}and {1,8}. Display a suitable message, if the given problem instance doesn't have a solution.
12.	Design and implement in Java to find all **Hamiltonian Cycles** in a connected undirected Graph G of *n* vertices using backtracking principle.

Laboratory Outcomes: The student should be able to:

- Design algorithms using appropriate design techniques (brute-force, greedy, dynamic programming, etc.)
- Implement a variety of algorithms such assorting, graph related, combinatorial, etc., in a high level language.
- Analyze and compare the performance of algorithms using language features.
- Apply and implement learned algorithm design techniques and data structuresto solve real-world problems.

Conduct of Practical Examination:

- Experiment distribution
  - o For laboratories having only one part: Students are allowed to pick one experiment from the lot with equal opportunity.
  - o For laboratories having PART A and PART B: Students are allowed to pick one experiment from PART A and one experiment from PART B, with equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure to be made zero of the changed part only.
- Marks Distribution *(Courseed to change in accoradance with university regulations)*
  - e) For laboratories having only one part – Procedure + Execution + Viva-Voce: 15+70+15 = 100 Marks
  - f) For laboratories having PART A and PART B
    - i. Part A – Procedure + Execution + Viva = 6 + 28 + 6 = 40 Marks
    - ii. Part B – Procedure + Execution + Viva = 9 + 42 + 9 = 60 Marks