Format No. QSP/7.1/01.F01 (B) Issue No. 05, Rev. No. 5, Dated: Jan 1, 2017

UNIVERSITY OF PETROLEUM & ENERGY STUDIES

School of Computer Science

Dehradun

LAB MANUAL

Programme : B. Tech. CSE spl. in IoT and Smart Cities

Course : Advanced Data Structures Lab

Subject Code : CSEG 1104

No. of credits : 1

Semester : II

Session : January 2019 – June 2019

Batch : 2018 - 2022

Prepared by : Mr. S. Christalin Nelson

Email : cnelson@ddn.upes.ac.in

Approved By

_____ _____

Faculty HOD

UPES Campus
"Energy Acres"
P.O. Bidholi, Via Prem Nagar, Dehradun

Tel: +91-135-2770137
Fax: +91 135- 27760904
Website: www.upes.ac.in

UPES
UNIVERSITY WITH A PURPOSE

LAB MANUAL

A. PRE-REQUISITES

Basic knowledge of C programming and Data Structures.

B. PROGRAM OUTCOMES (POs) and PROGRAM SPECIFIC OUTCOMES (PSOs) for B.Tech. CSE spl. in IoT and Smart Cities

B1. PROGRAM OUTCOMES (POs)

PO1. *Engineering knowledge:* Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. *Problem analysis:* Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. *Design/development of solutions:* Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4. *Conduct investigations of complex problems:* Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5. *Modern tool usage:* Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6. *The engineer and society:* Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7. *Environment and sustainability:* Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8. *Ethics:* Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. *Individual and team-work:* Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. *Communication:* Communicate effectively on complex engineering activities with the engineering community and with society at-large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11. *Project management and finance:* Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12. *Life-long learning:* Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

B2. PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1. Perform system and application programming using computer system concepts, concepts of Data Structures, algorithm development, problem solving and optimizing techniques.

PSO2. Apply software development and project management methodologies using concepts of front-end and back-end development and emerging technologies and platforms.

PSO3. Design and develop smart city and IoT applications using the principles of IoT and knowledge of cloud architectures and data analytics.

C. COURSE OBJECTIVES

1. To help the students to learn the basics of C++ programming language and enable them to contrast procedural and object-oriented approaches of problem solving.
2. To make students aware about the object-oriented concepts.
3. To enable students to develop programming skills using C++.
4. To provide students with the knowledge about advanced data structure concepts and enable students to implement data structures using C++ programming language.

D. COURSE OUTCOMES (COs), Mapping with POs and PSOs

Upon completion of this course the learners will be able to:

CO1. Demonstrate a clear understanding of the C++ programming environment and basic concepts including encapsulation, access control and data hiding, constructors, destructors and dynamic memory allocation.

CO2. Design solutions using the concepts of Inheritance and Polymorphism in object-oriented programming.

CO3. Exhibit a clear understanding of Generic Programming, Exception Handling ad File Handling.

CO4. Demonstrate and develop skills to work with Hash Tables, Trees and Graph data structures.

Table: Correlation of COs with POs and PSOs

PO/PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	1	1	2	1								1	3		
CO2	1	1	2	1								1	3		
CO3	1	1	2	1								1	3		
CO4	1	1	2	1								1	3		

1=weakly mapped 2= moderately mapped 3=strongly mapped

Table: Mapping of COs with POs and PSOs

Course Outcomes (COs)		Mapped PO & PSO
CO1	Demonstrate a clear understanding of the C++ programming environment and basic concepts including encapsulation, access control, constructors, destructors and dynamic memory allocation.	PO1, PO2, PO3, PO4, PO12, PSO1
CO2	Design solutions using the concepts of Inheritance and Polymorphism in object-oriented programming.	PO1, PO2, PO3, PO4, PO12, PSO1
CO3	Exhibit a clear understanding of Generic Programming, Exception Handling and File Handling.	PO1, PO2, PO3, PO4, PO12, PSO1
CO4	Demonstrate and develop skills to work with Hash Tables, Trees and Graph data structures.	PO1, PO2, PO3, PO4, PO12, PSO1

Table: Correlation of the Course with the POs and PSOs

Course Code	Course Title	PO1 Engineering Knowledge	PO2 Problem analysis	PO3 Design/development of solutions	PO4 Conduct investigations of complex problems	PO5 Modern tool usage	PO6 The engineer and society	PO7 Environment and sustainability	PO8 Ethics	PO9 Individual or team work	PO10 Communication	PO11 Project management and finance	PO12 Life-long Learning	PSO1 Perform system and application programming using computer system concepts, concepts of Data Structures, algorithm development, problem solving and optimizing techniques.	PSO2 Apply software development and project management methodologies using concepts of front-end and back-end development and emerging technologies and platforms.	PSO3 Design and develop smart city and IoT applications using the principles of IoT and knowledge of cloud architectures and data analytics.
CSEG 1004	Advanced Data Structure	1	1	2	1	0	0	0	0	0	0	0	1	3	0	0

1=weakly mapped 2= moderately mapped 3=strongly mapped

E. COURSE OUTLINE

Expt. No.	Big Ideas/ Topics	CO	Modality
1	Classes and Objects	CO1	F2F/e-lab
2	Constructor and Destructor	CO1	F2F/e-lab
3	Dynamic Memory Allocation	CO1	F2F/e-lab
4	Friends to a Class, Inheritance	CO1, CO2	F2F/e-lab
5	Redefinition and Overloading	CO2	F2F/e-lab
6	Virtual Functions and Abstract Classes	CO2	F2F/e-lab
7	Generic Programming	CO3	F2F/e-lab
8	File Handling	CO3	F2F/e-lab
9	Exception Handling	CO3	F2F/e-lab
10	Hash Tables	CO4	F2F/e-lab
11	Trees	CO4	F2F/e-lab
12	Graphs	CO4	F2F/e-lab

F. PEDAGOGY

Solution to the problems should be designed (algorithm/flow-chart/pseudocode) and tested. After obtaining a successful design, the solution to the problem is implemented using C++ language. Students are evaluated based on Performance (via efficient design, implementation) and record keeping, and preparation of students (via viva-voce and quiz).

G. COURSE COMPLETION PLAN

No. of Labs Sessions planned	No. of Viva planned	No. of Quiz planned
12	2	1

One Session = 120 minutes

H. EVALUATION & GRADING

Students are continuously evaluated through the proceedings of the course based on the following:

1) Performance & Record (50%)
2) Viva Voce & Quiz (50%)

H.1. Performance & Record: WEIGHTAGE - 50%

A minimum of 12 experiments should be handled as F2F in the lab or online through e-labs/virtual labs. This component is obtained through continuous evaluation after performing the experiments. Each experiment requires several activities to be performed for better understanding of C++ programming and data structures concepts by the students. More details of these lab activities are available in Section I.

Sample Scheme of Experiment Evaluation (10 marks per experiment)	
Algorithm Design	3 Marks
Coding Syntax	2 Marks
Execution / Bug Finding	2 Marks
Records (submitted **before** the very next turn.)	3 Marks

H.2. Viva-Voce and Quiz: WEIGHTAGE - 50%

The preparation of the students would be evaluated based on two viva-voce (with a weightage of 20% each conducted by mid-term and end-term) and one Quiz (of 10% weightage conducted by end-term).

It is mandatory for the students to attend the above said evaluations. Students who do not attend will lose their marks. Continuous Internal Assessment Record Sheet will be displayed at the end of the semester.

GRADING:

At course completion, the student is awarded with a grade (on a 10-point scale) based on the overall marks obtained comprising the above two mentioned out of 100 marks. Students scoring less than 35 absolute marks in individual course shall be awarded a 'F' grade. Students scoring 85 marks and above as composite score shall be awarded a 'O' grade. The minimum individual course grade is 'C'.

The student who is debarred due to shortage of attendance for a lab course will need to repeat the continuous evaluation during summer vacation (June-July) after the registration by payment of the prescribed fee per subject as notified by the University. The grades are awarded based on the performance of the students and capping as per the University guidelines. All other rules and regulations such as requirement of passing, etc. will remain same.

I. DETAILED SESSION PLAN

Activities listed under 'Lab activities' should be performed F2F (face-to-face) in the allocated laboratories (either compiled in lab machines or compiled online using e-labs).

UPES
UNIVERSITY WITH A PURPOSE

EXPERIMENT – 1

Title: **Classes and Objects**

Objective: **To understand the concept of classes and objects, data hiding and encapsulation**

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Consider rectangle, square, triangle and circle as 4 given shapes with a common area of 314 sq.cm. The length of a rectangle is 2cm and height of right-angled triangle is 4cm. Find the shape which has the smallest perimeter and largest perimeter.
2. Use for loop to print the truth table for the expression XY+Z.
3. Convert Decimal number into a Binary Number using 1D array.

X	Y	Z	XY+Z
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
.	.	.	.
.	.	.	.
1	1	1	1

EXPERIMENT – 2

Title: **Constructor and Destructor**

Objective: **To understand the concept of Constructor and Destructor**

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Design a class with a recursive member function with prototype “int fact(int &no)” to find the factorial of a number (n) obtained using a parameterized constructor. The class also has a destructor. Illustrate dynamic initialization of the variable ‘int result’ with a suitable function call to fact() from main().
2. Implement the operations of a Queue Data Structure using 1D array and constructor initialization list.

EXPERIMENT – 3

Title: **Dynamic Memory Allocation**

Objective: **To understand the concept of Dynamic Memory Allocation**

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Implement the operations of a dynamically allocated Stack data structure using a self-referential structure and single Linked List.
2. Create a C++ class with member functions that use *this pointer* to perform complex number arithmetic. The objects of this class should only be dynamically allocated [Hint: use private destructor].

EXPERIMENT – 4

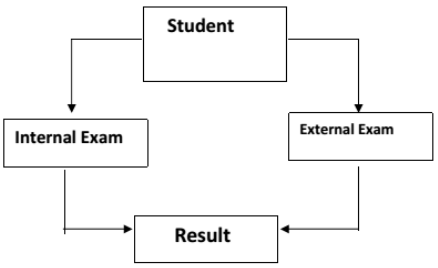
Title: Friends to a Class and Inheritance

Objective: To understand the concepts of friend function, friend class and Inheritance

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Implement Inheritance as shown in the diagram. Class Student has members functions student_Details() and display_Student_Details() and name, roll no, and gender as data members. Internal_Exam and External_Exam classes have separate functions to get and display Internal and External marks for 6 subjects. Define Result class to evaluate and display the results suitably (40% Internal + 60% External).



2. Read molecular values for Oxygen, Hydrogen and Sulphur in three different classes named Oxygen, Hydrogen and Sulphur. These classes have a common non-member friend function which is used to find out the chemical compounds (Sulphuric acid, Sulphurous acid, Thiosulphuric acid) that can be formed based on the given input molecular values.
3. Create a student class with student sapid, name, marks of 6 subjects as its members and initializes the details. Use friend class that access the details of student, calculates total marks, average marks and prints the result.

EXPERIMENT – 5

Title: Redefinition and Overloading

Objective: To understand the concepts of Redefinition and Overloading

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Base class 'Temperature' obtains temperature value in Celsius (c) through its parameterized constructor with default arguments and convert it into Fahrenheit value (f) and prints it using the function calculate(). Class 'Temp' inherits class 'Temperature' to obtain the temperature value in Fahrenheit and converts it into Kelvin value (k) and prints it using the redefined function calculate().
2. Define a class with following member functions
    a. constructor to obtain side1, side2.
    b. void choice() to have menu driven control to find and print the area of square, rectangle, or triangle
    c. int calculate(int) to compute the Area of Square
    d. int calculate(int, int) to compute the Area of Rectangle
    e. float calculate(int, int) to compute the Area of Triangle
3. Add two matrices by overloading binary operator +.

EXPERIMENT – 6

Title: Virtual Functions and Abstract Classes

Objective: To understand the concepts of Virtual Functions and Abstract Classes

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Base class 'Temperature' obtains temperature value in Kelvin (k) through its parameterized constructor, convert it into Celsius value (c) and prints it using the virtual function calculate(). Class 'Temp' inherits class 'Temperature' to obtain the temperature value in Celsius (c), converts it into Fahrenheit value (f) and prints it using the overridden function calculate().
2. Abstract Base class 'Temperature' obtains temperature value in Celsius (c) through its constructor and has a pure virtual function calculate(). Class 'Fahrenheit' inherits class 'Temperature' to obtain the temperature value in Celsius (c), converts it into Fahrenheit value (f) and prints it using calculate(). Class 'Kelvin' inherits class 'Temperature' to obtain the temperature value in Celsius (c), converts it into Kelvin value (k) and prints it using calculate().

EXPERIMENT – 7

Title: Generic Programming

Objective: To understand the concepts of Generic Functions and Generic Classes

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Write a generic function to sort the given elements in ascending order using bubble sort algorithm. This generic function must support integer and float types.
2. Write a generic class to implement the operations of a stack data structure using arrays. This generic class must support integer and float types.

EXPERIMENT – 8

Title: File Handling

Objective: To understand the concept of File Handling

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Write contents into a Test file. Get a word from the user as input, compute the number of occurrences of this word in the file and print the count.
2. Define class 'Residence' with data member house_no, income, house_name, and house_type. The class has the following member functions:
    a. void input() – obtains house_no, house_name and income. Calls assign()
    b. void assign() – allocates house type based on income (A/B/C/etc.)
    c. void output() – prints house details
    d. int house() – returns house_no

    main() is menu driven to (1) add multiple records as required into a binary file in append mode (2) Enter house_no to locate the record in the binary file and modify it using random access (3) display records from the binary file.

EXPERIMENT – 9

Title: Exception Handling

Objective: To understand the concepts of Exception Handling.

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Write an interactive program to compute different roots of a Quadratic Equation. While computing square root of a number the input value must be tested for validity. If it is negative, the user defined function my_sqrt() should be invoked by set_terminate() to handle the exception promptly.
2. Write an interactive program to compute different roots of a Quadratic Equation. While computing square root of a number the input value must be tested for validity. If it is negative,

the catch block of the user-defined exception class object should be made to handle the exception promptly apart from the other catch blocks.

3. Illustrate Stack unwinding while working with Exception Handling.

EXPERIMENT – 10

Title: Hash tables

Objective: To understand the concepts of Hash tables

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Implement a hash function on student SAP-ID and categorize them in to their 10 families based on the last three digits. (E.g. Student with SAP-ID 5000423 belongs to family 9 and student with SAP-ID 5000425 belongs to family 2 based on last three digits)
2. Implement a Hash table using arrays. Perform Insert, Delete and Search operations on the hash table using the above Hash function. Adopt a suitable user-defined exception handling strategy if collision occurs while inserting data.
3. Repeat S.No.2 with Linear probing as Collision avoidance strategy.

EXPERIMENT – 11

Title: Trees

Objective: To understand the concepts of Trees Data Structure

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Create a binary tree using an array/linked List.
2. Construct a Binary Tree and perform Inorder, Preorder and Postorder Traversal.
3. Implement Heap Sort.

EXPERIMENT – 12

Title: Graphs

Objective: To understand the concepts of Graph Data Structure

List of Lab Activities:

Write algorithm, prepare test cases and dry run the algorithm to test the design. Code using C++ language and evaluate the code using the test cases so as to obtain the desired results.

1. Accept the vertices and edges for a graph and stores it as an adjacency matrix. Implement functions to print indegree, outdegree and to display the adjacency matrix.
2. Accept the vertices and edges for a graph and stores it as an adjacency list. Implement functions to print outdegree of any vertex i.
3. Accept the graph as an adjacency matrix and checks if the graph is undirected. The matrix for undirected graph is symmetric

J. SUGGESTED READING

J.1 Text Books

1. Herbert Schildt, "C++: The Complete Reference", McGraw Hill Education, 2003.
2. John R. Hubbard, "Data Structures with C++", Schaum's Outlines, Tata McGraw Hill Education, 2000.

J.2 Reference Books

1. Michael T. Goodrich, Roberto Tamassia, David Mount, "Data Structures and Algorithms in C++", Wiley India Pvt. Ltd., 2004.
2. Seymour Lipschutz, "Data Structures", Schaum's Outlines, Tata McGraw Hill Education, 2006.

K. GUIDELINES

Cell Phones and other Electronic Communication Devices: Cell phones and other electronic communication devices (such as Blackberries/Laptops) MUST be turned off during the lab session.

E-Mail and online learning tool: Each student in the class should have UPES e-mail id and a password to access the Blackboard regularly. The best way to arrange meetings with faculty or is by email and prior appointment. Various research papers/reference material will be mailed/uploaded on online learning platform time to time.

Attendance: Students are required to have **minimum attendance of 75%** in the subject.